Journal of Machine Learning Research 21 (2020) 1-31 Submitted 8/18; Revised 5/20; Published 6/20

Optimal Algorithms for Continuous Non-monotone Submodular and DR-Submodular Maximization

Rad Niazadeh RAD.NIAZADEH@CHICAGOBOOTH.EDU
Chicago Booth School of Business, University of Chicago,
5807 S Woodlawn Ave,
Chicago, IL 60637, USA.

Tim Roughgarden TIM.ROUGHGARDEN@GMAIL.COM
Department of Computer Science, Columbia University,
500 West 120th Street, Room 450,
New York, NY 10027, USA.

Joshua R. Wang JOSHUAWANG@GOOGLE.COM
Google Research,
1600 Amphitheatre Pkwy,
Mountain View, CA 94043, USA.

Editor: Prof. Andreas Krause

Abstract

In this paper we study the fundamental problems of maximizing a continuous non-monotone submodular function over the hypercube, both with and without coordinate-wise concavity. This family of optimization problems has several applications in machine learning, economics, and communication systems. Our main result is the first $\frac{1}{2}$-approximation algorithm for continuous submodular function maximization; this approximation factor of $\frac{1}{2}$ is the best possible for algorithms that only query the objective function at polynomially many points. For the special case of DR-submodular maximization, i.e. when the submodular function is also coordinate-wise concave along all coordinates, we provide a different $\frac{1}{2}$-approximation algorithm that runs in quasi-linear time. Both these results improve upon prior work (Bian et al., 2017a,b; Soma and Yoshida, 2017).

Our first algorithm uses novel ideas such as reducing the guaranteed approximation problem to analyzing a zero-sum game for each coordinate, and incorporates the geometry of this zero-sum game to fix the value at this coordinate. Our second algorithm exploits coordinate-wise concavity to identify a monotone equilibrium condition sufficient for getting the required approximation guarantee, and hunts for the equilibrium point using binary search. We further run experiments to verify the performance of our proposed algorithms in related machine learning applications.

Keywords: Continuous submodularity, non-monotone submodular maximization, approximation algorithms

1. Introduction

Submodular optimization is a sweet spot between tractability and expressiveness, with numerous applications in machine learning (e.g., Krause and Golovin (2014), and see below) while permitting many algorithms that are both practical and backed by rigorous guarantees

(e.g., Buchbinder et al. (2015b); Feige et al. (2011); Calinescu et al. (2011)). In general, a real-valued function $\mathcal{F}$ defined on a lattice $\mathcal{L}$ is *submodular* if and only if

$$\mathcal{F}(x \vee y) + \mathcal{F}(x \wedge y) \leq \mathcal{F}(x) + \mathcal{F}(y) \quad ,$$

for all $x, y \in \mathcal{L}$, where $x \vee y$ and $x \wedge y$ denote the join and meet, respectively, of x and y in the lattice $\mathcal{L}$. Such functions are generally neither convex nor concave. In one of the most commonly studied examples, $\mathcal{L}$ is the lattice of subsets of a fixed ground set (or a sublattice thereof), with union and intersection playing the roles of join and meet, respectively.

This paper concerns a different well-studied setting, where $\mathcal{L}$ is a hypercube (i.e., $[0,1]^n$), with componentwise maximum and minimum serving as the join and meet, respectively. We consider the fundamental problem of (approximately) maximizing a continuous and submodular function over the hypercube. We further assume function $\mathcal{F} : [0,1]^n \to [0,1]$ is non-negative, bounded (by 1), coordinate-wise Lipschitz continuous and differentiable.[1] The function $\mathcal{F}$ is given as a "black box", which means it is accessible only via querying its value at a point. In later parts of the paper (Section 3), we further consider access to first-order partials of $\mathcal{F}$.[2] We are interested in algorithms that use at most a polynomial (in n) number of queries. We do not assume $\mathcal{F}$ is monotone, otherwise the problem is trivial.

Our results. Maximizing a submodular function over the hypercube is at least as difficult as over the subsets of a ground set. An instance of the latter problem is a discrete problem, but can be converted to one of the former by extending the given set function f (with domain viewed as $\{0,1\}^n$) to its multilinear extension $\mathcal{F}$ defined on the hypercube (where $\mathcal{F}(\mathbf{x}) = \sum_{S \subseteq [n]} \prod_{i \in S} x_i \prod_{i \notin S} (1 - x_i) f(S)$). Sampling based on an α-approximate solution for the multilinear extension yields an equally good approximate solution to the original discrete problem. For this discrete problem, considering algorithms that make polynomial number of (set or integral) queries to the discrete function, the best approximation ratio achievable is $\frac{1}{2}$; the (information-theoretic) lower bound is originally due to Feige et al. (2011), a new proof based on symmetry gap is due to Vondrák (2013), and the optimal algorithm due to Buchbinder et al. (2015b). Moreover, by employing standard techniques based on symmetry gap of submodular functions as in Vondrák (2013), we can show that integral queries to the discrete function are as good as fractional queries to the multi-linear extension in this lower bound (Remark 5). Thus, the best-case scenario for maximizing a submodular function over the hypercube (using polynomially many queries, possibly fractional) is a $\frac{1}{2}$-approximation. The main result of this paper achieves this best-case scenario:

> *There is an algorithm for maximizing a continuous submodular function over the hypercube that guarantees a $\frac{1}{2}$-approximation (up to additive $\epsilon > 0$ error) while using only a polynomial number of queries in n and $\frac{1}{\epsilon}$ to the function under mild (Lipschitz) continuity assumptions.*

1. More generally, for reasons that will be clear later, the function only has to be nonnegative at the points $\vec{0}$ and $\vec{1}$, and bounded. Also, in most parts of the paper, differentiablity of $\mathcal{F}$ is not fundamentally necessary; however, for the sake of a clean presentation of main ideas, we use all these assumptions.
2. This assumption is only a technical assumption for the ease of presentation, lack of which will only cause an extra small additive error due to Lipschitz-ness. We will elaborate on this later.

Note that both additive error ϵ and Lipschitz continuity are required to obtain any meaningful positive results in this problem, as the special case of 1-dimensional continuous submodular function is an arbitrary single variable function.

Our algorithm is inspired by the *bi-greedy* algorithm of Buchbinder et al. (2015b), which maximizes a submodular set function; it maintains two solutions initialized at $\vec{0}$ and $\vec{1}$, go over coordinates sequentially, and make the two solutions agree on each coordinate. The algorithmic question here is how to choose the new coordinate value for the two solutions, so that the algorithm gains enough value relative to the optimum in each iteration. Prior to our work, the best-known result was a $\frac{1}{3}$-approximation (Bian et al., 2017b), which is also inspired by the bi-greedy. Our algorithm requires a number of new ideas, including a reduction to the analysis of a zero-sum game for each coordinate, and the use of the special geometry of this game to bound the value of the game.

We further consider a well-studied special class of submodular functions that are concave in each coordinate. This class is called *DR-submodular* in Soma and Yoshida (2015) (inspired by diminishing returns defined in Kapralov et al. (2013)). Here, an optimal $\frac{1}{2}$-approximation algorithm was already known on integer lattices (Soma and Yoshida, 2017), that can easily be generalized to our continuous setting as well; our contribution is a significantly faster such bi-greedy algorithm. The main idea here is to identify a monotone equilibrium condition sufficient for getting the required approximation guarantee, which enables a binary search-type solution.

We should also point out that both of our theoretical results extend naturally to arbitrary axis-aligned boxes (i.e., "box constraints"). We summarize our results and how they are compared with previous work in Table 1.

Table 1: Summary of results for non-monotone continuous submodular maximization.

Reference	**Setting**	**Constraint**	**Objective**	**Complexity**
Buchbinder et al. (2015b)	discrete/submodular	unconstrained	$\frac{1}{2}\cdot\texttt{OPT}$	$O(n)$
Bian et al. (2017a)	smooth/submodular	box	$\frac{1}{3}\cdot\texttt{OPT}-\epsilon$	$O(\frac{n}{\epsilon})$
Bian et al. (2017b) Feldman et al. (2011)	smooth/DR-submodular	convex/down-closed	$\frac{1}{e}\cdot\texttt{OPT}-\epsilon$	$O(\frac{1}{\epsilon})^*$
Mokhtari et al. (2018)	stochastic†/smooth/DR-submodular	convex	$\frac{1}{e}\cdot\texttt{OPT}-\epsilon$	$O(\frac{1}{\epsilon^3})^*$
Soma and Yoshida (2017)	integer lattice/DR-submodular	integer box	$\frac{1}{2}\cdot\texttt{OPT}$	$O(n^2)$
[This paper]	smooth/submodular	box	$\frac{1}{2}\cdot\texttt{OPT}-\epsilon$	$O(\frac{n^2}{\epsilon})$
[This paper]	smooth/DR-submodular	box	$\frac{1}{2}\cdot\texttt{OPT}-\epsilon$	$O(\frac{n}{\epsilon})$

* Query complexity has dependency on Lipschitz constant and diameter of the constraint set.
† For stochastic functions, algorithms only have access to unbiased samples of the gradient.

Applications. We next briefly mention four applications of maximizing a non-monotone submodular function over a hypercube that are germane to machine learning and other

related application domains. See Appendix A for more details on these applications.

Non-concave quadratic programming. In this problem, the goal is to maximize $\mathcal{F}(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x} + c$, where the off-diagonal entries of $\mathbf{H}$ are non-positive. One application of this problem is large-scale price optimization on the basis of demand forecasting models (Ito and Fujimaki, 2016).

Map inference for Determinantal Point Processes (DPP). DPPs are elegant probabilistic models that arise in statistical physics and random matrix theory. DPPs can be used as generative models in applications such as text summarization, human pose estimation, and news threading tasks (Kulesza et al., 2012). The approach in Gillenwater et al. (2012) to the problem boils down to maximize a suitable submodular function over the hypercube, accompanied with an appropriate rounding (see also Bian et al. (2017a)). We should point out this vanilla version of this problem is indeed an instance of DR-submodular maximization; however, one can also think of regularizing this objective function with ℓ_2-norm regularizer, in order to avoid overfitting. Even with a regularizer, the function remains submodular, but it does not necessarily remain DR-submodular.

Log-submodularity and mean-field inference. Another probabilistic model that generalizes DPPs and all other strong Rayleigh measures (Li et al., 2016; Zhang et al., 2015) is the class of *log-submodular* distributions over sets, i.e., $p(S) \sim \exp(\mathcal{F}(S))$ where $\mathcal{F}(\cdot)$ is a set submodular function. MAP inference over this distribution has applications in machine learning (Djolonga and Krause, 2014). One variational approach towards this MAP inference task is to use *mean-field inference* to approximate the distribution p with a product distribution $\mathbf{x} \in [0,1]^n$, which again boils down to submodular function maximization over the hypercube (see Bian et al. (2017a)). We should point out this problem is indeed an instance of DR-submodular maximization.

Revenue maximization over social networks. In this problem, there is a seller who wants to sell a product over a social network of buyers. To do so, the seller gives away trial products and fractions thereof to the buyers in the network (Bian et al., 2017b; Hartline et al., 2008). In Bian et al. (2017b), there is an objective function that takes into account two parts: the revenue gain from those who did not get a free product, where the revenue function for any such buyer is a non-negative non-decreasing and submodular function $R_i(\mathbf{x})$; and the revenue loss from those who received the free product, where the revenue function for any such buyer is a non-positive non-increasing and submodular function $\bar{R}_i(\mathbf{x})$. The combination for all buyers is a non-monotone submodular function. It is also non-negative at $\vec{0}$ and $\vec{1}$, by extending the model and accounting for extra revenue gains from buyers with free trials.

In order to verify the performance of our proposed algorithms in some of these practical machine learning applications, we further run experiments on synthetic data. We observe that our algorithms match the performance of the prior work in these experiments, while providing either a better guaranteed approximation or a better running time.

Further related work. Buchbinder and Feldman (2016) derandomize the bi-greedy algorithm. Staib and Jegelka (2017) apply continuous submodular optimization to budget allocation, and develop a new submodular optimization algorithm to this end. Hassani et al. (2017) give a $\frac{1}{2}$-approximation for *monotone* continuous submodular functions under

convex constraints. Gotovos et al. (2015) consider (adaptive) submodular maximization when feedback is given after an element is chosen. Chen et al. (2018); Roughgarden and Wang (2018) consider submodular maximization in the context of online no-regret learning. Mirzasoleiman et al. (2013) show how to perform submodular maximization with distributed computation. Buchbinder et al. (2015a) studies the competitive ratio that can be obtained for same problem in the online competitive setting. Submodular minimization has been studied in Schrijver (2000); Iwata et al. (2001). See Bach et al. (2013) for a survey on more applications in machine learning.

Equivalent definitions of continuous submodularity. Two related properties to (continuous) submodularity studied in the literature are *weak Diminishing Returns Submodularity* (`weak DR-SM`) and *strong Diminishing Returns Submodularity* (`strong DR-SM`) (Bian et al., 2017b), formally defined below.

Definition 1 (`Weak/Strong DR-SM`) *Consider a continuous function $\mathcal{F} : [0,1]^n \to [0,1]$. For any $\mathbf{x} \in \mathbb{R}^n, i \in [n]$, let $\mathbf{x}_{-i} \triangleq [x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n]$. We define the following two properties:*

- *`Weak DR-SM` (continuous submodular): $\forall i \in [n],\ \forall \boldsymbol{x}_{-i} \leq \boldsymbol{y}_{-i} \in [0,1]^n$, and $\forall \delta \geq 0, \forall z$*

$$\mathcal{F}(z+\delta, \boldsymbol{x}_{-i}) - \mathcal{F}(z, \boldsymbol{x}_{-i}) \geq \mathcal{F}(z+\delta, \boldsymbol{y}_{-i}) - \mathcal{F}(z, \boldsymbol{y}_{-i}) \ .$$

- *`Strong DR-SM` (*DR-submodular *): $\forall i \in [n],\ \forall \boldsymbol{x} \leq \boldsymbol{y} \in [0,1]^n$, and $\forall \delta \in [0, 1 - y_i]$:*

$$\mathcal{F}(x_i+\delta, \boldsymbol{x}_{-i}) - \mathcal{F}(\boldsymbol{x}) \geq \mathcal{F}(y_i+\delta, \boldsymbol{y}_{-i}) - \mathcal{F}(\boldsymbol{y}) \ .$$

As simple corollaries, a twice-differentiable $\mathcal{F}$ is `strong DR-SM` if and only if all the entries of its Hessian are non-positive, and `weak DR-SM` if and only if all of the *off-diagonal* entries of its Hessian are non-positive. Also, `weak DR-SM` together with concavity along each coordinate is equivalent to `strong DR-SM` (see Proposition 19 in Appendix B for the proof).

As an important remark, it can be shown that `weak DR-SM` is equivalent to submodularity and `strong DR-SM` is equivalent to DR-submodularity. Therefore, we use these terms interchangeably in the rest of the paper. See Proposition 19 in Appendix B for more details and a formal treatment of these connections.

Coordinate-wise Lipschitz continuity. Consider univariate functions generated by fixing all but one of the coordinates of the original function $\mathcal{F}(\cdot)$. In future sections, we sometimes require mild technical assumptions on the Lipschitz continuity of these single dimensional functions.

Definition 2 (Coordinate-wise Lipschitz) *A function $\mathcal{F} : [0,1]^n \to [0,1]$ is* coordinate-wise Lipschitz continuous *if there exists a constant $C > 0$ such that $\forall i \in [n]$, $\forall \boldsymbol{x}_{-i} \in [0,1]^n$, the single variate function $\mathcal{F}(\cdot, \boldsymbol{x}_{-i})$ is C-Lipschitz continuous, i.e.,*

$$\forall z_1, z_2 \in [0,1]: \quad |\mathcal{F}(z_1, \boldsymbol{x}_{-i}) - \mathcal{F}(z_2, \boldsymbol{x}_{-i})| \leq C|z_1 - z_2| \ .$$

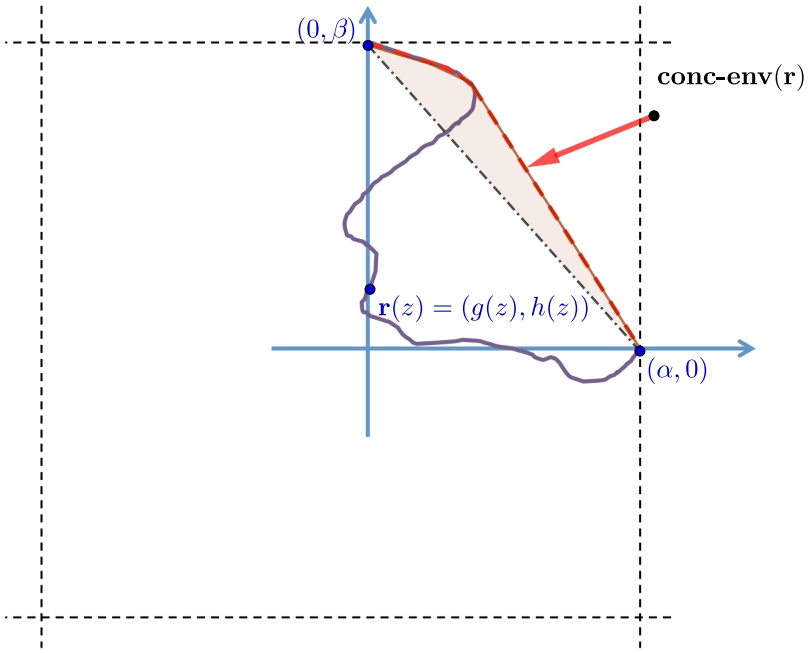


Figure 1: Continuous curve $\mathbf{r}(z)$ in $\mathbb{R}^2$ (dark blue), positive-orthant concave envelope (red).

2. Weak DR-SM Maximization: Continuous Randomized Bi-Greedy

Our first main result is a $\frac{1}{2}$-approximation algorithm (up to additive error δ) for maximizing a continuous submodular function $\mathcal{F}$, a.k.a., `weak DR-SM`, which is information-theoretically optimal (Feige et al., 2011). This result assumes that $\mathcal{F}$ is coordinate-wise Lipschitz continuous.[3] Before describing our algorithm, we introduce the notion of the *positive-orthant concave envelope* of a two-dimensional curve, which is useful for understanding our algorithm.

Definition 3 *Consider a curve $\boldsymbol{r}(z) = (g(z), h(z)) \in \mathbb{R}^2$ over the interval $z \in [Z_l, Z_u]$ such that for $\alpha, \beta \in [0,1]$:*

1. *$g : [Z_l, Z_u] \to [-1, \alpha]$ and $h : [Z_l, Z_u] \to [-1, \beta]$ are both continuous,*
2. *$g(Z_l) = h(Z_u) = 0$, and $h(Z_l) = \beta \in [0,1]$, $g(Z_u) = \alpha \in [0,1]$.*

Then the positive-orthant concave envelope *of $\boldsymbol{r}(\cdot)$, denoted by `conc-env`$(\boldsymbol{r})$, is the smallest concave curve in the positive-orthant upper-bounding all the points $\{\mathbf{r}(z) : z \in [Z_l, Z_u]\}$ (see Figure 1), i.e.,*

$$\texttt{conc-env}(\boldsymbol{r}) \triangleq \textit{upper-face}\left(\textit{conv}\left(\{\mathbf{r}(z) : z \in [Z_l, Z_u]\}\right)\right) \ .$$

3. Such an assumption is necessary, since otherwise the single-dimensional problem amounts to optimizing an arbitrary function, and is hence intractable. Prior work, e.g., Bian et al. (2017b) and Bian et al. (2017a), implicitly requires such an assumption to perform single-dimensional optimization.

Here is how the rest of Section 2 is organized. We first consider a vanilla version of our algorithm for maximizing $\mathcal{F}$ over the unit hypercube, termed as *continuous randomized bi-greedy* (Algorithm 1). This version assumes blackbox oracle access to algorithms for a few computations involving univariate functions of the form $\mathcal{F}(., \mathbf{x}_{-i})$ (e.g., maximization over $[0,1]$, computing `conc-env`(.), etc.). In Section 2.1, we prove that the vanilla algorithm finds a solution with an objective value of at least $\frac{1}{2}$ of the optimum. In Section 2.2, we then show how to approximately implement these oracles in polynomial time when $\mathcal{F}$ is coordinate-wise Lipschitz.

Algorithm 1: (Vanilla) Continuous Randomized Bi-Greedy

input: function $\mathcal{F} : [0,1]^n \to [0,1]$;
output: vector $\hat{\mathbf{z}} = (\hat{z}_1, \ldots, \hat{z}_n) \in [0,1]^n$;
Initialize $\mathbf{X} \leftarrow (0, \ldots, 0)$ and $\mathbf{Y} \leftarrow (1, \ldots, 1)$;
for $i = 1$ *to* n **do**
  Find $Z_u, Z_l \in [0,1]$ such that $\begin{cases} Z_l \in \underset{z \in [0,1]}{\operatorname{argmax}} \mathcal{F}(z, \mathbf{Y}_{-i}) \\ Z_u \in \underset{z \in [0,1]}{\operatorname{argmax}} \mathcal{F}(z, \mathbf{X}_{-i}) \end{cases}$;
  if $Z_u \leq Z_l$ **then**
    $\hat{z}_i \leftarrow Z_l$;
  else
    $\forall z \in [Z_l, Z_u]$, let $\begin{cases} g(z) \triangleq \mathcal{F}(z, \mathbf{X}_{-i}) - \mathcal{F}(Z_l, \mathbf{X}_{-i}), \\ h(z) \triangleq \mathcal{F}(z, \mathbf{Y}_{-i}) - \mathcal{F}(Z_u, \mathbf{Y}_{-i}), \end{cases}$;
    Let $\alpha \triangleq g(Z_u)$ and $\beta \triangleq h(Z_l)$; `// note that` $\alpha, \beta \geq 0$
    Let $\mathbf{r}(z) \triangleq (g(z), h(z))$ be a continuous two-dimensional curve in $[-1, \alpha] \times [-1, \beta]$;
    Compute `conc-env`$(\mathbf{r})$ (i.e., positive-orthant concave envelope of $\mathbf{r}(t)$ as in Definition 3) ;
    Find point $\mathcal{P} \triangleq$ intersection of `conc-env`$(\mathbf{r})$ and the line $h' - \beta = g' - \alpha$ on g-h plane ;
    Suppose $\mathcal{P} = \lambda \mathcal{P}_1 + (1-\lambda)\mathcal{P}_2$, where $\lambda \in [0,1]$ and $\mathcal{P}_j = \mathbf{r}(z^{(j)}), z^{(j)} \in [Z_l, Z_u]$ for $j = 1, 2$, and both points are also on the `conc-env`$(\mathbf{r})$; `// see Figure 2`
    Randomly pick $\hat{z}_i$ such that $\begin{cases} \hat{z}_i \leftarrow z^{(1)} & \text{with probablity } \lambda \\ \hat{z}_i \leftarrow z^{(2)} & \text{o.w.} \end{cases}$;
  Let $X_i \leftarrow \hat{z}_i$ and $Y_i \leftarrow \hat{z}_i$; `// after this,` $\mathbf{X}$ `and` $\mathbf{Y}$ `will agree on coordinate` i
return $\hat{\mathbf{z}} = (\hat{z}_1, \ldots, \hat{z}_n)$

Theorem 4 *If $\mathcal{F}(\cdot)$ is non-negative and submodular (or equivalently is* `weak DR-SM`*), then Algorithm 1 is a randomized $\frac{1}{2}$-approximation algorithm, i.e., returns $\hat{\mathbf{z}} \in [0,1]^n$ s.t.*

$$2\mathbf{E}\left[\mathcal{F}(\hat{\mathbf{z}})\right] \geq \mathcal{F}(\mathbf{x}^*), \qquad \text{where } \mathbf{x}^* \in \underset{\boldsymbol{x} \in [0,1]^n}{\operatorname{argmax}} \mathcal{F}(\boldsymbol{x}) \text{ is the optimal solution.}$$

Running time. The algorithm has n iterations. Moreover, as our implementation in Section 2.2 shows, we need to make $O(n/\epsilon)$ function computations at each iteration, assuming mild Lipschtiz continuity. These function computations will result in a running time of $O(n^2/\epsilon)$. See Theorem 13 for more details.

Remark 5 *There exists a (family of)* `strong DR-SM` *continuous function(s) $\mathcal{F}(\cdot)$ so that no $(\frac{1}{2}+\epsilon)$-approximation is possible with polynomial in n number of value queries for any $\epsilon > 0$. This statement is true for maximizing discrete non-monotone submodular set functions (Feige et al., 2011). At the first glance, since* multi-linear extension *of a submodular set function $f(.)$, i.e.,*

$$\mathcal{F}(\mathbf{x}) \triangleq \sum_{S \subseteq [n]} f(S) \prod_{i \in S} x_i \prod_{i \notin S} (1 - x_i) \ ,$$

is a special case of our setting, one might think Feige et al. (2011) implies the same hardness result for continuous submodular functions. However, when querying a multi-linear extension, we might be able to benefit from querying fractional points (and not only querying integral points that correspond to sets). Interestingly, fractional queries are as helpful as intergral queries by employing a standard symmetry gap argument à la Vondrák (2013). In a nutshell, by looking at Lemma 3.3 in Vondrák (2013), the crux of the argument for a reduction from symmetry gap to inapproximibility is that for any fixed query Q to the objective function $\mathcal{F}(.)$, the associated vector $q = \xi(Q)$ in Lemma 3.3 is very likely to be close to its symmetrized version $\bar{q}$ due to a simple concentration bound. This is only "more true" for interior points, as the same concentration bound (even a slightly stronger version) still holds (cf. Vondrák (2013) for more details).[4]

2.1. Analysis of the Continuous Randomized Bi-Greedy (proof of Theorem 4)

We start by defining these vectors, used in our analysis in the same spirit as Buchbinder et al. (2015b):

$$\begin{aligned}
&i \in [n]: \ \mathbf{X}^{(i)} \triangleq (\hat{z}_1, \ldots, \hat{z}_i, 0, 0, \ldots, 0), && \mathbf{X}^{(0)} \triangleq (0, \ldots, 0) \\
&i \in [n]: \ \mathbf{Y}^{(i)} \triangleq (\hat{z}_1, \ldots, \hat{z}_i, 1, 1, \ldots, 1), && \mathbf{Y}^{(0)} \triangleq (1, \ldots, 1) \\
&i \in [n]: \ \mathbf{O}^{(i)} \triangleq (\hat{z}_1, \ldots, \hat{z}_i, x^*_{i+1}, \ldots, x^*_n), && \mathbf{O}^{(0)} \triangleq (x^*_1, \ldots, x^*_n)
\end{aligned}$$

Note that $\mathbf{X}^{(i)}$ and $\mathbf{Y}^{(i)}$ (or $\mathbf{X}^{(i-1)}$ and $\mathbf{Y}^{(i-1)}$) are the values of $\mathbf{X}$ and $\mathbf{Y}$ at the end of (or at the beginning of) the i^{th} iteration of Algorithm 1. A nice geometric way of thinking about this set of vectors is thinking of them as three paths inside the unit hypercube, denoted as *lower path*, *upper path*, and the *optimal path*. Lower-path (or upper-path) is defined by $\left(\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(n)}\right)$ (or $\left(\mathbf{Y}^{(1)}, \ldots, \mathbf{Y}^{(n)}\right)$), which starts from $[0, 0, \ldots, 0]$ (or $[1, 1, \ldots, 1]$) and ends at $\hat{\mathbf{z}}$. Similarly, optimal path is defined by $\left(\mathbf{O}^{(1)}, \ldots, \mathbf{O}^{(n)}\right)$, which starts from the optimal point $\mathbf{x}^*$ and ends at $\hat{\mathbf{z}}$. Importantly, these three paths all collide at the final point returned by the algorithm which is $\hat{\mathbf{z}}$.

Algorithm 1 is a single-pass algorithm that goes over coordinates one by one in an arbitrary order, and searches for the right coordinate value $\hat{z}_i$ for each coordinate i. Once it decides a coordinate value, it fixes this decision and never changes $\hat{z}_i$ in future iterations. At the core of our algorithm there is a single dimensional search sub-problem per each coordinate to find $\hat{z}_i$. Fix a coordinate $i \in [n]$. In order to prove the correctness of the particular coordinate-wise search of Algorithm 1, we need to define a few mathematical

4. We confirmed the correctness of this statement with the author through a personal communication. We omit further details here as it is beyond the scope of our paper and requires defining several new notations as in Vondrák (2013).

components related to marginal changes of the function on lower and upper paths when processing coordinate i.

Definition 6 *Fix coordinate i and values $\hat{z}_1, \ldots, \hat{z}_{i-1}$. Define $Z_l \in [0,1]$ and $Z_u \in [0,1]$ to be (one of the) maximizers of the function $\mathcal{F}(.)$ on upper and lower paths respectively, i.e.,*

$$Z_l \in \operatorname*{argmax}_{z \in [0,1]} \mathcal{F}(z, \mathbf{Y}_{-i}) \quad , \quad Z_u \in \operatorname*{argmax}_{z \in [0,1]} \mathcal{F}(z, \mathbf{X}_{-i}) \ .$$

Moreover, lower path marginal function $g(.) : [0,1] \to [0,1]$ and upper path marginal function $h(.) : [0,1] \to [0,1]$ are defined as follows:

$$g(z) \triangleq \mathcal{F}(z, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(Z_l, \mathbf{X}_{-i}^{(i-1)}) \quad , \quad h(z) \triangleq \mathcal{F}(z, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(Z_u, \mathbf{Y}_{-i}^{(i-1)}).$$

Lemma 7 *Upper/lower path marginal functions in Definition 6 satisfy these properties:*

1. $\forall z \in [0,1] : -1 \leq g(z) \leq g(Z_u) \triangleq \alpha$ *and* $-1 \leq h(z) \leq h(Z_l) \triangleq \beta$,
2. $\alpha \in [0,1], \beta \in [0,1]$,
3. *Single crossing property: The univariate function $d(z) = h(z) - g(z)$ is monotone non-increasing.*

Proof The first two items hold because $\mathcal{F}(\mathbf{x}) \in [0,1]$, and by definition of Z_l and Z_u (see Definition 6). By using the `weak DR-SM` property of $\mathcal{F}(.)$ the proof of last item is immediate, as for any $\delta \geq 0$,

$$d(z+\delta) - d(z) = \Big(\mathcal{F}(z+\delta, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(z, \mathbf{Y}_{-i}^{(i-1)})\Big) - (F(z+\delta, \mathbf{X}_{-i}^{(i-1)}) - F(z+\delta, \mathbf{X}_{-i}^{(i-1)})) \leq 0,$$

where the inequality holds due to the fact that $\mathbf{Y}_{-i}^{(i-1)} \geq \mathbf{X}_{-i}^{(i-1)}$ and $\delta \geq 0$. ■

In the remainder of this section, we show how to use the above ingredients to analyze Algorithm 1; we sketch our proofs, give the high-level ideas, and finally provide formal proofs for different components of our analysis.

2.1.1. Reduction to coordinate-wise zero-sum games.

For each coordinate $i \in [n]$, we consider a sub-problem. In particular, define a two-player *zero-sum game* played between the *algorithm player* (denoted by `ALG`) and the *adversary player* (denoted by `ADV`). `ALG` selects a (randomized) strategy $\hat{z}_i \in [0,1]$, and `ADV` selects a (randomized) strategy $x_i^* \in [0,1]$. Recall the descriptions of $g(z)$ and $h(z)$ at iteration i of Algorithm 1.

Remark 8 *For a `weak DR` submodular function $\mathcal{F}(\cdot)$, it is not hard to see that the two-dimensional curve $(g(z), h(z))$ crosses every line with slope 1 at most once, e.g., as in Figure 1.This fact is an immediate consequence of monotonictiy of $g(z) - h(z)$ (Lemma 7).*

We now define the utility of ALG (negative of the utility of ADV) in our zero-sum game as follows:

$$\mathcal{V}^{(i)}(\hat{z}_i, x_i^*) \triangleq \frac{1}{2}g(\hat{z}_i) + \frac{1}{2}h(\hat{z}_i) - \max\left(g(x_i^*) - g(\hat{z}_i), h(x_i^*) - h(\hat{z}_i)\right). \tag{1}$$

Suppose the expected utility of ALG is non-negative at the equilibrium of this game. In particular, suppose ALG's randomized strategy $\hat{z}_i$ (in Algorithm 1) guarantees that for every strategy x_i^* of ADV the expected utility of ALG is non-negative. If this statement holds for all of the zero-sum games corresponding to different iterations $i \in [n]$, then Algorithm 1 is a $\frac{1}{2}$-approximation of the optimum.

Lemma 9 *If* $\forall i \in [n] : \mathbf{E}\left[\mathcal{V}^{(i)}(\hat{z}_i, x_i^*)\right] \geq -\delta/n$ *for constant* $\delta > 0$, *then* $2\mathbf{E}\left[\mathcal{F}(\hat{\mathbf{z}})\right] \geq \mathcal{F}(\mathbf{x}^*) - \delta$.

Proof sketch. Our bi-greedy approach, á la Buchbinder et al. (2015b), revolves around analyzing the evolving values of three points: $\mathbf{X}^{(i)}$, $\mathbf{Y}^{(i)}$, and $\mathbf{O}^{(i)}$. These three points begin at all-zeroes, all-ones, and the optimum solution, respectively, and converge to the algorithm's final point. In each iteration, we aim to relate the total increase in value of the first two points with the decrease in value of the third point. If we can show that the former quantity is at least twice the latter quantity, then a telescoping sum proves that the algorithm's final choice of point scores at least half that of optimum.

The utility of our game is specifically engineered to compare the total increase in value of the first two points with the decrease in value of the third point. The positive term of the utility is half of this increase in value, and the negative term is a bound on how large in magnitude the decrease in value may be. As a result, an overall nonnegative utility implies that the increase beats the decrease by a factor of two, exactly the requirement for our bi-greedy approach to work. Finally, an additive slack of δ/n in the utility of each game sums over n iterations for a total slack of δ.

Proof [*formal proof of Lemma 9*] Consider a realization of $\hat{z}_i$ where $\hat{z}_i \geq x_i^*$. We have:

$$\begin{aligned}
\mathcal{F}(\mathbf{O}^{(i-1)}) - \mathcal{F}(\mathbf{O}^{(i)}) &= \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, x_i^*, x_{i+1}^*, \ldots, x_n^*) - \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, \hat{z}_i, x_{i+1}^*, \ldots, x_n^*) \\
&= -\left(\mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, \hat{z}_i, x_{i+1}^*, \ldots, x_n^*) - \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, x_i^*, x_{i+1}^*, \ldots, x_n^*)\right) \\
&\leq -\left(\mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, \hat{z}_i, 1, \ldots, 1) - \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, x_i^*, 1, \ldots, 1)\right) \\
&= \left(\mathcal{F}(x_i^*, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(Z_u, \mathbf{Y}_{-i}^{(i-1)})\right) - \left(\mathcal{F}(\hat{z}_i, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(Z_u, \mathbf{Y}_{-i}^{(i-1)})\right) \\
&= h(x_i^*) - h(\hat{z}_i)\ , \qquad (2)
\end{aligned}$$

where the inequality holds due to the weak DR-SM. Similarly, for a a realization of $\hat{z}_i$ where $\hat{z}_i \leq x_i^*$:

$$\begin{aligned}
\mathcal{F}(\mathbf{O}^{(i-1)}) - \mathcal{F}(\mathbf{O}^{(i)}) &= \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, x_i^*, x_{i+1}^*, \ldots, x_n^*) - \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, \hat{z}_i, x_{i+1}^*, \ldots, x_n^*) \\
&\leq \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, x_i^*, 0, \ldots, 0) - \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, \hat{z}_i, 0, \ldots, 0) \\
&= \left(\mathcal{F}(x_i^*, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(Z_l, \mathbf{X}_{-i}^{(i-1)})\right) - \left(\mathcal{F}(\hat{z}_i, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(Z_l, \mathbf{X}_{-i}^{(i-1)})\right) \\
&= g(x_i^*) - g(\hat{z}_i)\ . \qquad (3)
\end{aligned}$$

Putting eq. (2) and eq. (3) together, for every realization $\hat{z}_i$ we have:

$$F(\mathbf{O}^{(i-1)}) - \mathcal{F}(\mathbf{O}^{(i)}) \leq \max\left(g(x_i^*) - g(\hat{z}_i), h(x_i^*) - h(\hat{z}_i)\right) \ . \tag{4}$$

Moreover, consider the term $\mathcal{F}(\mathbf{X}^{(i)}) - \mathcal{F}(\mathbf{X}^{(i-1)})$. We have:

$$\begin{aligned}
\mathcal{F}(\mathbf{X}^i) - \mathcal{F}(\mathbf{X}^{(i-1)}) &= \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, \hat{z}_i, 0, \ldots, 0) - \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, 0, 0, \ldots, 0) \\
&= g(\hat{z}_i) - g(0) = g(\hat{z}_i) + \mathcal{F}(Z_l, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(\mathbf{X}^{(i-1)}) \\
&\geq g(\hat{z}_i) + \mathcal{F}(Z_l, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(0, \mathbf{Y}_{-i}^{(i-1)}) \geq g(\hat{z}_i) \ ,
\end{aligned} \tag{5}$$

where the first inequality holds due to `weak DR-SM` property and the second inequity holds as $Z_l \in \operatorname{argmax}_{z \in [0,1]} \mathcal{F}(z, \mathbf{Y}_{-i}^{(i-1)})$. Similarly, consider the term $\mathcal{F}(\mathbf{Y}^{(i)}) - \mathcal{F}(\mathbf{Y}^{(i-1)})$. We have:

$$\begin{aligned}
\mathcal{F}(\mathbf{Y}^{(i)}) - \mathcal{F}(\mathbf{Y}^{(i-1)}) &= \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, \hat{z}_i, 1, \ldots, 1) - \mathcal{F}(\hat{z}_1, \ldots, \hat{z}_{i-1}, 1, 1, \ldots, 1) \\
&= h(\hat{z}_i) - h(1) = h(\hat{z}_i) + \mathcal{F}(Z_u, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(\mathbf{Y}^{(i-1)}) \\
&\geq h(\hat{z}_i) + \mathcal{F}(Z_u, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(1, \mathbf{X}_{-i}^{(i-1)}) \geq h(\hat{z}_i) \ ,
\end{aligned} \tag{6}$$

where the first inequality holds due to `weak DR-SM` and the second inequity holds as $Z_u \in \operatorname{argmax}_{z \in [0,1]} \mathcal{F}(z, \mathbf{X}_{-i}^{(i-1)})$. By eq. (4), eq. (5), eq. (6), and the fact that $\mathcal{F}(\mathbf{0}) + \mathcal{F}(\mathbf{1}) \geq 0$, we have:

$$\begin{aligned}
-\delta &\leq \sum_{i=1}^{n} \mathbf{E}\left[\mathcal{V}^{(i)}(\hat{z}_i, x_i^*)\right] = \sum_{i=1}^{n} \left(\frac{1}{2}\mathbf{E}\left[g(\hat{z}_i)\right] + \frac{1}{2}\mathbf{E}\left[h(\hat{z}_i)\right] - \mathbf{E}\left[\max\left(g(x_i^*) - g(\hat{z}_i), h(x_i^*) - h(\hat{z}_i)\right)\right]\right) \\
&\leq \mathbf{E}\left[\frac{1}{2}\sum_{i=1}^{n}\left(\mathcal{F}(\mathbf{X}^{(i)}) - \mathcal{F}(\mathbf{X}^{(i-1)})\right) + \frac{1}{2}\sum_{i=1}^{n}\left(\mathcal{F}(\mathbf{Y}^{(i)}) - \mathcal{F}(\mathbf{Y}^{(i-1)})\right) - \sum_{i=1}^{n}\left(\mathcal{F}(\mathbf{O}^{(i-1)}) - \mathcal{F}(\mathbf{O}^{(i)})\right)\right] \\
&= \mathbf{E}\left[\frac{\mathcal{F}(\mathbf{X}^{(n)}) - \mathcal{F}(\mathbf{X}^{(0)})}{2} + \frac{\mathcal{F}(\mathbf{Y}^{(n)}) - \mathcal{F}(\mathbf{Y}^{(0)})}{2} - \mathcal{F}(\mathbf{O}^{(0)}) + \mathcal{F}(\mathbf{O}^{(n)})\right] \\
&\leq \mathbf{E}\left[\frac{\mathcal{F}(\hat{\mathbf{z}})}{2} + \frac{\mathcal{F}(\hat{\mathbf{z}})}{2} - \mathcal{F}(\mathbf{x}^*) + \mathcal{F}(\hat{\mathbf{z}})\right] = 2\mathbf{E}\left[\mathcal{F}(\hat{\mathbf{z}})\right] - \mathcal{F}(\mathbf{x}^*) \ .
\end{aligned}$$

■

We next show how to analyze the coordinate-wise zero-sum games introduced in this section. The analysis of this zero-sum game is based on a novel geometric intuition related to the `weak-DR` submodular functions (which will be described shortly) and is basically the technical heart of our entire analysis. We formally prove the following proposition.

Proposition 10 *If `ALG` plays the (randomized) strategy $\hat{z}_i$ as described in Algorithm 1, then we have $\mathbf{E}\left[\mathcal{V}^{(i)}(\hat{z}_i, x_i^*)\right] \geq 0$ against any strategy x_i^* of `ADV`.*

2.1.2. Analyzing the zero-sum games (proof of Proposition 10)

Fix an iteration $i \in [n]$ of Algorithm 1. We now lower-bound the optimal value of the game played between the adversary player and the algorithm player. To do so, we consider a

particular strategy of the algorithm player, which is following Algorithm 1, and show that no matter what strategy the adversary plays the expected utility of the algorithm player is non-negative. Formally, we prove Proposition 10. To this end, we consider two cases based on the values of Z_l and Z_u (defined in Algorithm 1) and prove each case separately:

☐ **Case $\mathbf{Z_l \geq Z_u}$ (*easy*):** In this case, the algorithm plays a deterministic strategy $\hat{z}_i = Z_l$. We therefore have:

$$\mathcal{V}^{(i)}(\hat{z}_i, x_i^*) = \frac{1}{2}g(\hat{z}_i) + \frac{1}{2}h(\hat{z}_i) - \max\left(g(x_i^*) - g(\hat{z}_i), h(x_i^*) - h(\hat{z}_i)\right) \geq \min(g(\hat{z}_i) - g(x_i^*), 0) \ ,$$

where the inequality holds because $g(\hat{z}_i) = g(Z_l) = 0$, and also $Z_l \in \operatorname*{argmax}_{z\in[0,1]} \mathcal{F}(z, \mathbf{Y}_{-i}^{(i)})$, so:

- $h(\hat{z}_i) = h(Z_l) = \mathcal{F}(Z_l, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(Z_u, \mathbf{Y}_{-i}^{(i-1)}) \geq 0$,
- $h(x_i^*) - h(\hat{z}_i) = \mathcal{F}(x_i^*, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(Z_l, \mathbf{Y}_{-i}^{(i-1)}) \leq 0$.

To complete the proof for this case, it only remains to show $g(\hat{z}_i) - g(x_i^*) \geq 0$ for $\hat{z}_i = Z_l$. As $Z_l \geq Z_u$, we have:

$$\begin{aligned} g(\hat{z}_i) - g(x_i^*) &= \mathcal{F}(Z_l, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(x_i^*, \mathbf{X}_{-i}^{(i-1)}) \\ &= \mathcal{F}(Z_l, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(Z_u, \mathbf{X}_{-i}^{(i-1)}) + \mathcal{F}(Z_u, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(x_i^*, \mathbf{X}_{-i}^{(i-1)}) \\ &\geq \Big(\mathcal{F}(Z_l, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(Z_u, \mathbf{Y}_{-i}^{(i-1)})\Big) + \Big(\mathcal{F}(Z_u, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(x_i^*, \mathbf{X}_{-i}^{(i-1)})\Big) \geq 0 \ , \end{aligned}$$

where the first inequality uses the `weak DR-SM` property and the second inequality holds because both terms are non-negative, following the fact that:

$$Z_l \in \operatorname*{argmax}_{z\in[0,1]} \mathcal{F}(z, \mathbf{Y}_{-i}^{(i)}) \qquad \text{and} \qquad Z_u \in \operatorname*{argmax}_{z\in[0,1]} \mathcal{F}(z, \mathbf{X}_{-i}^{(i)}) \ .$$

Therefore, we finish the proof of the easy case.

☐ **Case $\mathbf{Z_l < Z_u}$ (*hard*):** In this case, `ALG` plays a mixed strategy over two points. To determine the two-point support, it considers the curve $\mathbf{r} = \{(g(z), h(z))\}_{z\in[Z_l, Z_u]}$ and finds a point $\mathcal{P}$ on `conc-env`$(\mathbf{r})$ (i.e., Definition 3) that lies on the line $h' - \beta = g' - \alpha$, where we recall that $\alpha = g(Z_u) \geq 0$ and $\beta = h(Z_l) \geq 0$ (see Lemma 7). Because this point is on the concave envelope, it should be a convex combination of two points on the curve $\mathbf{r}(z)$. Let's say $\mathcal{P} = \lambda\mathcal{P}_1 + (1-\lambda)\mathcal{P}_2$, where $\mathcal{P}_1 = \mathbf{r}(z^{(1)})$ and $\mathcal{P}_2 = \mathbf{r}(z^{(2)})$, and $\lambda \in [0,1]$. The final strategy of `ALG` is a mixed strategy over $\{z^{(1)}, z^{(2)}\}$ with probabilities $(\lambda, 1-\lambda)$. Fixing any mixed strategy of `ALG` over two points $\mathcal{P}_1 = (g_1, h_1)$ and $\mathcal{P}_2 = (g_2, h_2)$ with probabilities $(\lambda, 1-\lambda)$ (denoted by $\mathcal{F}_{\mathcal{P}}$), define the `ADV`'s *positive region*, i.e.,

$$(g', h') \in [-1,1]\times[-1,1] : \quad \mathbf{E}_{(g,h)\sim F_{\mathcal{P}}}\left[\frac{1}{2}g + \frac{1}{2}h - \max(g'-g, h'-h)\right] \geq 0.$$

Now, suppose `ALG` plays a mixed strategy with the property that its corresponding `ADV`'s positive region covers the entire curve $\{g(z), h(z)\}_{z\in[0,1]}$. Then, for any strategy x_i^* of `ADV` the expected utility of `ALG` is non-negative. In the rest of the proof, we geometrically characterize the `ADV`'s positive region against a mixed strategy of `ALG` over a 2-point support (Lemma 11), and then we show that for the particular choice of $\mathcal{P}_1$, $\mathcal{P}_2$ and λ in Algorithm 1 the positive region covers the entire curve $\{g(z), h(z)\}_{z\in[0,1]}$ (Lemma 12).

Lemma 11 *Suppose* ALG *plays a 2-point mixed strategy over* $\mathcal{P}_1 = \boldsymbol{r}(z^{(1)}) = (g_1, h_1) \in [0,\alpha] \times [0,\beta]$ *and* $\mathcal{P}_2 = \boldsymbol{r}(z^{(1)}) = (g_2, h_2) \in [0,\alpha] \times [0,\beta]$ *with probabilities* $(\lambda, 1-\lambda)$*, and w.l.o.g.* $h_1 - g_1 \geq h_2 - g_2$*. Then* ADV*'s positive region is the pentagon* $(\mathcal{M}_0, \mathcal{M}_1, \mathcal{Q}_1, \mathcal{Q}_2, \mathcal{M}_2)$*, where* $\mathcal{M}_0 = (-1,-1)$ *and (see Figure 2):*

1. $\mathcal{M}_1 = \big(-1, \lambda(\frac{3}{2}h_1 + \frac{1}{2}g_1) + (1-\lambda)(\frac{3}{2}h_2 + \frac{1}{2}g_2)\big) \in [-1,0] \times [0,1]$,

2. $\mathcal{M}_2 = \big(\lambda(\frac{3}{2}g_1 + \frac{1}{2}h_1) + (1-\lambda)(\frac{3}{2}g_2 + \frac{1}{2}h_2), -1\big) \in [0,1] \times [-1,0]$,

3. $\mathcal{Q}_1$ *is the intersection of the lines leaving* $\mathcal{P}_1$ *with slope* 1 *and leaving* $\mathcal{M}_1$ *along the g-axis,*

4. $\mathcal{Q}_2$ *is the intersection of the lines leaving* $\mathcal{P}_2$ *with slope* 1 *and leaving* $\mathcal{M}_2$ *along the h-axis.*

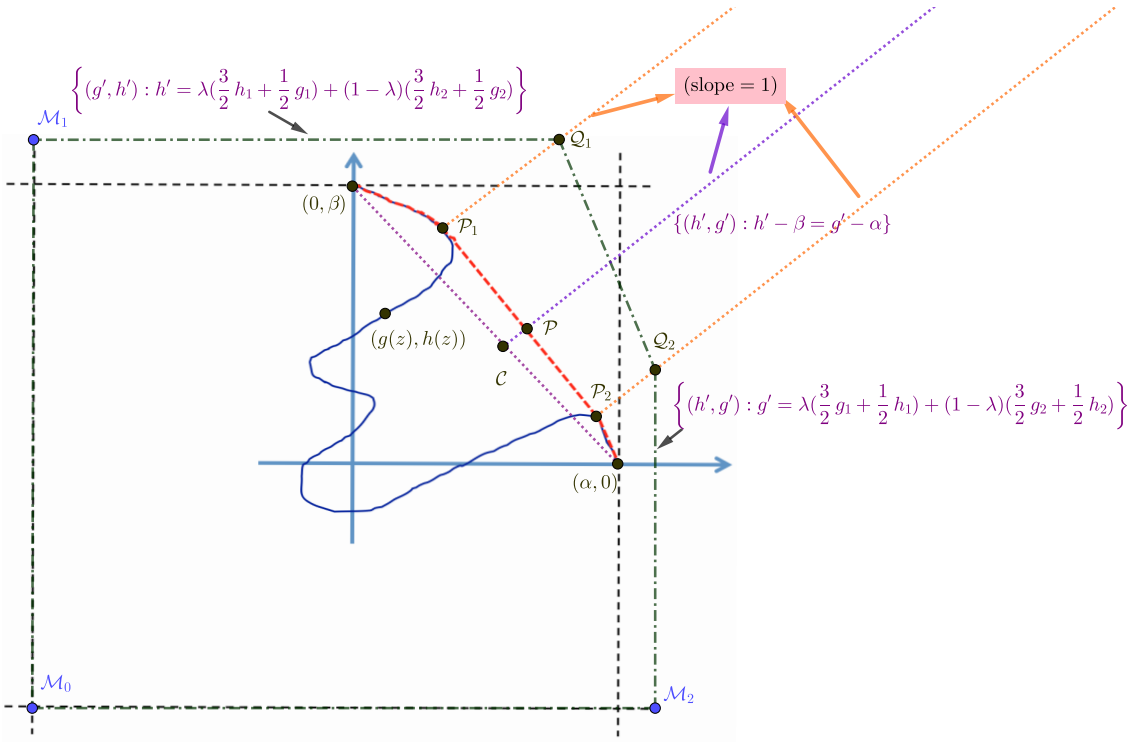


Figure 2: Pentagon $(\mathcal{M}_0, \mathcal{M}_1, \mathcal{Q}_1, \mathcal{Q}_2, \mathcal{M}_2)$= ADV player's positive region against a mixed strategy over two points $\mathcal{P}_1$ and $\mathcal{P}_2$. Note that line $h' - \beta = g' - \alpha$ always has an intersection with conc-env($\mathbf{r}$), simply because both conc-env($\mathbf{r}$) and line $\frac{g'}{\alpha} + \frac{h'}{\beta} = 1$ cross $(0,\beta)$ and $(\alpha,0)$, and conc-env($\mathbf{r}$) is always upper than $\frac{g'}{\alpha} + \frac{h'}{\beta} = 1$.

Proof Being equipped with Lemma 7, the positive region is the set of all points $(g', h') \in [-1, 1]^2$ such that

$$\mathbf{E}_{(g,h)\sim F_{\mathcal{P}}}\left[\frac{1}{2}g + \frac{1}{2}h - \max(g' - g, h' - h)\right]$$
$$= \lambda\left(\frac{1}{2}g_1 + \frac{1}{2}h_1 - \max(g' - g_1, h' - h_1)\right) + (1-\lambda)\left(\frac{1}{2}g_2 + \frac{1}{2}h_2 - \max(g' - g_2, h' - h_2)\right) \geq 0\ .$$

The above inequality defines a polytope. Our goal is to find the vertices and faces of this polytope. Now, to this end, we only need to consider three cases: 1) $h' - g' \geq h_1 - g_1$, 2) $h_2 - g_2 \leq h' - g' \leq h_1 - g_1$ and 3) $h' - g' \leq h_2 - g_2$ (note that $h_1 - g_1 \geq h_2 - g_2$). From the first and third case we get the half-spaces $h' \leq \lambda(\frac{3}{2}h_1 + \frac{1}{2}g_1) + (1-\lambda)(\frac{3}{2}h_2 + \frac{1}{2}g_2)$ and $g' \leq \lambda(\frac{3}{2}g_1 + \frac{1}{2}h_1) + (1-\lambda)(\frac{3}{2}g_2 + \frac{1}{2}h_2)$ respectively, that form two of the faces of the positive-region polytope. Note that $\lambda(\frac{3}{2}h_1 + \frac{1}{2}g_1) + (1-\lambda)(\frac{3}{2}h_2 + \frac{1}{2}g_2) \geq 0$ and $\lambda(\frac{3}{2}g_1 + \frac{1}{2}h_1) + (1-\lambda)(\frac{3}{2}g_2 + \frac{1}{2}h_2) \geq 0$. From the second case, we get another half-space, but the observation is that the transition from first case to second case happens when $h' - g' = h_1 - g_1$, i.e., on a line with slope one leaving $\mathcal{P}_1$ (simply because $\frac{h'-h_1}{g'-g_1} = 1$ for any point (g', h') on the transition line), and transition from second case to the third case happens when $h' - g' = h_2 - g_2$, i.e., on a line with slope one leaving $\mathcal{P}_2$ (again simply because $\frac{h'-h_2}{g'-g_2} = 1$ for any point (g', h') on the transition line). Therefore, the second half-space is the region under the line connecting two points $\mathcal{Q}_1$ and $\mathcal{Q}_2$, where $\mathcal{Q}_1$ is the intersection of $h' = \lambda(\frac{3}{2}h_1 + \frac{1}{2}g_1) + (1-\lambda)(\frac{3}{2}h_2 + \frac{1}{2}g_2)$ and the line leaving $\mathcal{P}_1$ with slope one (point $\mathcal{Q}_1$), and $\mathcal{Q}_2$ is the intersection of $g' = \lambda(\frac{3}{2}g_1 + \frac{1}{2}h_1) + (1-\lambda)(\frac{3}{2}g_2 + \frac{1}{2}h_2)$ and the line leaving $\mathcal{P}_2$ with slope one (point $\mathcal{Q}_2$). Due to the geometric interpretation of the second case (that it is the region between lines $\mathcal{P}_1 - \mathcal{Q}_1$ and $\mathcal{P}_2 - \mathcal{Q}_2$), and because points of the positive region are below $\mathcal{M}_1 - \mathcal{P}_1$ and to the left of $\mathcal{M}_2 - \mathcal{P}_2$, the line segment $\mathcal{Q}_1 - \mathcal{Q}_2$ defines another face of the positive region polytope. Moreover, $\mathcal{Q}_1$ and $\mathcal{Q}_2$ will be two vertices on this face. By intersecting the three mentioned half-spaces with $g' \geq -1$ and $h \geq -1$ (which define the two remaining faces of the positive region polytope), the postive region will be the polytope defined by the pentagon $(\mathcal{M}_0, \mathcal{M}_1, \mathcal{Q}_1, \mathcal{Q}_2, \mathcal{M}_2)$, as claimed (see Figure 2 for a pictorial proof). ■

By applying Lemma 11, we have the following main technical lemma. The proof is geometric and is pictorially visible in Figure 2. This lemma finishes the proof of Proposition 10.

Lemma 12 (*main technical lemma*) *If* `ALG` *plays the two point mixed strategy described in Algorithm 1, then for every $x_i^* \in [0, 1]$ the point $(g', h') = (g(x_i^*), h(x_i^*))$ is in the* `ADV`*'s positive region.*

Proof sketch. For simplicity assume $Z_l = 0$ and $Z_u = 1$. To understand the `ADV`'s positive region that results from playing a two-point mixed strategy by `ALG`, we consider the positive region that results from playing a one point pure strategy. When `ALG` chooses a point (g, h), the positive term of the utility is one-half of its one-norm. The negative term of the utility is the worse between how much the `ADV`'s point is above `ALG`'s point, and how much it is to the right of `ALG`'s point. The resulting positive region is defined by an upper boundary $g' \leq \frac{3}{2}g + \frac{1}{2}h$ and a right boundary $h' \leq \frac{1}{2}g + \frac{3}{2}h$.

Next, let's consider what happens when we pick point (g_1, h_1) with probability λ and point (g_2, h_2) with probability $(1-\lambda)$. We can compute the expected point: let $(g_3, h_3) =$

$\lambda(g_1, h_1) + (1-\lambda)(g_2, h_2)$. As suggested by Lemma 11, the positive region for our mixed strategy has three boundary conditions: an upper boundary, a right boundary, and a corner-cutting boundary. The first two boundary conditions correspond to a pure strategy which picks (g_3, h_3). By design, (g_3, h_3) is located so that these boundaries cover the entire $[-1, \alpha] \times [-1, \beta]$ rectangle. This leaves us with analyzing the corner-cutting boundary, which is the focus of Figure 2. As it turns out, the intersections of this boundary with the two other boundaries lie on lines of slope 1 extending from $(g_j, h_j)_{j=1,2}$. If we consider the region between these two lines, the portion under the envelope (where the curve $\mathbf{r}$ may lie) is distinct from the portion outside the corner-cutting boundary. However, if $\mathbf{r}$ were to ever violate the corner-cutting boundary condition without violating the other two boundary conditions, it must do so in this region. Hence the resulting positive region covers the entire curve $\mathbf{r}$, as desired.

Proof [*formal proof of Lemma 12*] First of a all, we claim that any ADV's strategy $x_i^* \in [0, Z_l)$ (or $x_i^* \in (Z_u, 1]$) is weakly dominated by Z_l (or Z_u) if ALG plays a (randomized) strategy $\hat{z}_i \in [Z_l, Z_u]$. To see this, note that if $x_i^* \in [0, Z_l)$,

$$\begin{aligned}
&\max\left(g(x_i^*) - g(\hat{z}_i), h(x_i^*) - h(\hat{z}_i)\right) \\
&= \max\left(\mathcal{F}(x_i^*, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(\hat{z}_i, \mathbf{X}_{-i}^{(i-1)}), \mathcal{F}(x_i^*, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(\hat{z}_i, \mathbf{Y}_{-i}^{(i-1)})\right) \\
&= \mathcal{F}(x_i^*, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(\hat{z}_i, \mathbf{Y}_{-i}^{(i-1)}) \leq \mathcal{F}(Z_l, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(\hat{z}_i, \mathbf{Y}_{-i}^{(i-1)}) \\
&= h(Z_l) - h(\hat{z}_i) \leq \max\left(g(Z_l) - g(\hat{z}_i), h(Z_l) - h(\hat{z}_i)\right) \ ,
\end{aligned}$$

and therefore $\mathcal{V}^{(i)}(\hat{z}_i, Z_l) \leq \mathcal{V}^{(i)}(\hat{z}_i, x_i^*)$ for any $x_i^* \in [0, Z_l)$. Similarly, $\mathcal{V}^{(i)}(\hat{z}_i, Z_u) \leq \mathcal{V}^{(i)}(\hat{z}_i, x_i^*)$ for any $x_i^* \in (Z_u, 1]$. So, without loss of generality, we can assume ADV's strategy x_i^* is in $[Z_l, Z_u]$.

Now, consider the curve $\mathbf{r} = \{(g(z), h(z)\}_{z \in [Z_l, Z_u]}$ as in Figure 2. ALG's strategy is a 2-point mixed strategy over $\mathcal{P}_1 = (g_1, h_1) = \mathbf{r}(z^{(1)})$ and $\mathcal{P}_2 = (g_2, h_2) = \mathbf{r}(z^{(1)})$, where these two points are on different sides of the line $\mathcal{L} : h' - \beta = g' - \alpha$ (or both of them are on the line $\mathcal{L}$). Without loss of generality, assume $h_1 - g_1 \geq \beta - \alpha \geq h_2 - g_2$. Note that $\mathbf{r}(Z_l) = (0, \beta)$ is above the line $\mathcal{L}$ and $\mathbf{r}(Z_l) = (\alpha, 0)$ is below the line $\mathcal{L}$. So, because $h(z) - g(z)$ is monotone non-increasing due to Lemma 7, we should have $Z_l \leq z^{(1)} \leq z^{(2)} \leq Z_u$.

Using Lemma 11, the ADV's positive region is $(\mathcal{M}_0, \mathcal{M}_1, \mathcal{Q}_1, \mathcal{Q}_2, \mathcal{M}_2)$, where $\{\mathcal{M}_j\}_{j=1,2,3}$ and $\{\mathcal{Q}_j\}_{j=1,2}$ are as described in Lemma 11. The upper concave envelope conc-env($\mathbf{r}$) upper-bounds the curve $\mathbf{r}$. Therefore, to show that curve $\mathbf{r}$ is entirely covered by the ADV's positive region, it is enough to show its upper concave envelope conc-env($\mathbf{r}$) is entirely covered (as can also be seen from Figure 2). Lets denote the line leaving $\mathcal{P}_j$ with slope one by $\mathcal{L}_j$ for $j = 1, 2$. The curve conc-env($\mathbf{r}$) consists of three parts: the part above $\mathcal{L}_1$, the part below $\mathcal{L}_2$ and the part between $\mathcal{L}_1$ and $\mathcal{L}_2$ (the last part is indeed the line segment connecting $\mathcal{P}_1$ and $\mathcal{P}_2$). Interestingly, the line connecting $\mathcal{P}_1$ to $\mathcal{Q}_1$ and the line connecting $\mathcal{P}_2$ to $\mathcal{Q}_2$ both have slope 1. So, as it can be seen from Figure 2, if we show $\mathcal{Q}_1$ is above the line $h' = \beta$ and $\mathcal{Q}_2$ is to the right of the line $g' = \alpha$, then the conc-env($\mathbf{r}$) will entirely be covered by the positive region and we are done. To see why this holds, first note that λ has

been picked so that $\mathcal{P} \triangleq (\mathcal{P}_g, \mathcal{P}_h) = \lambda\mathcal{P}_1 + (1-\lambda)\mathcal{P}_2$). Due to Lemma 11,

$$\mathcal{Q}_{1,h} = \lambda(\frac{3}{2}h_1 + \frac{1}{2}g_1) + (1-\lambda)(\frac{3}{2}h_2 + \frac{1}{2}g_2) = \frac{3}{2}\mathcal{P}_h + \frac{1}{2}\mathcal{P}_g \ ,$$
$$\mathcal{Q}_{2,g} = \lambda(\frac{3}{2}g_1 + \frac{1}{2}h_1) + (1-\lambda)(\frac{3}{2}g_2 + \frac{1}{2}h_2) = \frac{3}{2}\mathcal{P}_g + \frac{1}{2}\mathcal{P}_h \ .$$

Moreover, the point $\mathcal{P} = (\mathcal{P}_g, \mathcal{P}_h)$ dominates the point $\mathcal{C} \triangleq (\frac{\alpha^2}{\alpha+\beta}, \frac{\beta^2}{\alpha+\beta})$ coordinate-wise. This dominance is simply true because the points $\mathcal{C}$ and $\mathcal{P}$ are actually the intersections of the line $\mathcal{L} : h' - \beta = g' - \alpha$ (with slope one) with the line connecting $(0, \beta)$ to $(\alpha, 0)$ and with the curve $\texttt{conc-env}(\mathbf{r})$ respectively. As $\texttt{conc-env}(\mathbf{r})$ upper-bounds the line connecting $(0, \beta)$ to $(\alpha, 0)$, and because $\mathcal{L}$ has slope one, $\mathcal{P}_h \geq \mathcal{C}_h = \frac{\beta^2}{\alpha+\beta}$ and $\mathcal{P}_g \geq \mathcal{C}_g = \frac{\alpha^2}{\alpha+\beta}$. Putting all the pieces together,

$$\mathcal{Q}_{1,h} \geq \frac{3}{2}\frac{\beta^2}{\alpha+\beta} + \frac{1}{2}\frac{\alpha^2}{\alpha+\beta} = \frac{\left(\alpha^2+\beta^2-2\alpha\beta\right)+2\beta^2+2\alpha\beta}{2(\alpha+\beta)} = \beta + \frac{(\alpha-\beta)^2}{2(\alpha+\beta)} \geq \beta \ ,$$
$$\mathcal{Q}_{2,g} \geq \frac{3}{2}\frac{\alpha^2}{\alpha+\beta} + \frac{1}{2}\frac{\beta^2}{\alpha+\beta} = \frac{\left(\alpha^2+\beta^2-2\alpha\beta\right)+2\alpha^2+2\alpha\beta}{2(\alpha+\beta)} = \alpha + \frac{(\alpha-\beta)^2}{2(\alpha+\beta)} \geq \alpha \ ,$$

which implies $\mathcal{Q}_1$ is above the line $h' = \beta$ and $\mathcal{Q}_2$ is to the right of the line $g' = \alpha$, as desired. ■

2.2. Polynomial-time Implementation Under Lipschitz Continuity

In this subsection, we give an efficient implementation of the continuous randomized bi-greedy (Algorithm 1) under the assumption that the function $\mathcal{F}$ is coordinate-wise Lipschitz continuous. The main result is as follows.

Theorem 13 *If $\mathcal{F}$ is coordinate-wise Lipschitz continuous with known constant $C > 0$, then there exists an implementation of Algorithm 1 that runs in time $O(n^2/\epsilon)$ and returns a (randomized) point $\hat{\mathbf{z}}$ s.t.*

$$2\mathbf{E}\left[\mathcal{F}(\hat{\mathbf{z}})\right] \geq \mathcal{F}(\mathbf{x}^*) - \epsilon,$$

where $\mathbf{x}^* \in \underset{\boldsymbol{x}\in[0,1]^n}{\operatorname{argmax}} \mathcal{F}(\boldsymbol{x})$ is the optimal solution.

Proof The plan is that instead of optimizing over $[0,1]^n$ (the "continuous" space), we define $\epsilon' = 2\epsilon/nC$ and optimize over the ϵ'-net: $\{0, \epsilon', 2\epsilon', ..., 1\}^n$ (the "lattice" space). We will compare the algorithm's randomized lattice point $\hat{\mathbf{z}}$ with the continuous optimum $\mathbf{x}^*$ and do so via the lattice optimum $\tilde{\mathbf{x}}^*$. Applying the definition of coordinate-wise Lipschitz, we can bound the gap between the continuous optimum and lattice optimum. The continuous optimum is within $\epsilon'/2$ ℓ_∞-distance of a lattice point, so applying the Lipschitz property to each of our n dimensions yields:

$$\mathcal{F}(\mathbf{x}^*) \leq \mathcal{F}(\tilde{\mathbf{x}}^*) + (\epsilon'/2)nC = \mathcal{F}(\tilde{\mathbf{x}}^*) + \epsilon.$$

Note that this is the only point where we use the Lipschitz property.

Next, we explain the key modifications to the algorithm to achieve a 1/2-approximation to the best lattice point. Algorithm 1 interfaces with function $\mathcal{F}$ and its domain in two ways: (i) when performing optimization to compute Z_l, Z_u and (ii) when computing the upper-concave envelope. Task (i) is relatively straightforward; the algorithm instead optimizes just over $\{0, \epsilon', 2\epsilon', ..., 1\}$ rather than all of $[0, 1]$, i.e.,

$$Z_l \in \operatorname*{argmax}_{z \in \{0,\epsilon',2\epsilon',\ldots,1\}} \mathcal{F}(z, \mathbf{Y}_{-i}^{(i-1)})$$

$$Z_u \in \operatorname*{argmax}_{z \in \{0,\epsilon',2\epsilon',\ldots,1\}} \mathcal{F}(z, \mathbf{X}_{-i}^{(i-1)})$$

Computing the modified Z_l and Z_u can be done via a linear scan over our lattice points, of which there are $O(nC/\epsilon)$. For task (ii), we now have a discrete sequence of points on the 2-D curve $(g(z), h(z))$ instead of the entire continuous curve, i.e.,

$$\left[(g(Z_l), h(Z_l)), \left(g(Z_l + \epsilon'), h(Z_l + \epsilon')\right), \ldots, (g(Z_u), h(Z_u))\right]$$

Note that these points still have a well defined concave envelope, and this concave envelope will have a piece-wise linear shape. We can compute the points that define the corners of this piece-wise linear concave envelope in linear time using Graham's algorithm (Graham, 1972). Although Graham's typically requires an additional log factor in its running time to sort its input, we avoid this by processing our sequence in order of increasing z-coordinate. As a result, we obtain a running time of $O(nC/\epsilon)$ in our implementation. For completeness, the detailed implementations of these two steps can be found as Algorithm 2 and Algorithm 3. All other steps of Algorithm 1 remain unchanged.

We now reprove Proposition 10 when the algorithm has been modified as above and the optimal solution is constrained to a lattice point. After doing so, the proof will be completed by redefining the initial optimal point $\mathbf{O}^{(0)}$ to be the lattice optimum $\tilde{\mathbf{x}}^*$. The key idea is to replace $[0, 1]$ with $\{0, \epsilon', 2\epsilon', ..., 1\}$ everywhere, but we briefly walk through each step of the proof to better contextualize the effect of this change.

Fix a coordinate i and consider a similar zero-sum game as in the proof of Proposition 10. We show the above modified algorithm picks a (randomized) strategy $\hat{z}_i$, so that $\mathbf{E}\left[\mathcal{V}^{(i)}(\hat{z}_i, \tilde{x}_i^*)\right] \geq 0$ for any strategy $\tilde{x}_i^* \in \{0, \epsilon', 2\epsilon', \ldots, 1\}$ of the adversary, where

$$\mathcal{V}^{(i)}(\hat{z}_i, \tilde{x}_i^*) \triangleq \frac{1}{2}g(\hat{z}_i) + \frac{1}{2}h(\hat{z}_i) - \max\left(g(\tilde{x}_i^*) - g(\hat{z}_i), h(\tilde{x}_i^*) - h(\hat{z}_i)\right)$$

Algorithm 2: Approximate One-Dimensional Optimization

input: function $f : [0, 1] \to [0, 1]$, additive error $\epsilon > 0$, Lipschitz Constant $C > 0$;
output: coordinate value $z \in [0, 1]^n$;
Set $\epsilon' \leftarrow \frac{\epsilon}{2nC}$;
Initialize $z^* \leftarrow 0$;
Initialize $z \leftarrow 0$;
while $z \leq 1$ **do**
  if $f(z) > f(z^*)$ **then**
    $z^* \leftarrow z$;
  $z \leftarrow z + \epsilon'$;
return z^*

Algorithm 3: Approximate Annotated Upper-Concave Envelope

input: function $g : [0,1] \to [0,1]$, function $h : [0,1] \to [0,1]$, additive error $\epsilon > 0$, Lipschitz Constant $C > 0$;
output: stacks s and t ;
Set $\epsilon' \leftarrow \frac{\epsilon}{2nC}$;
Initialize stacks s, t ;
Initialize $z \leftarrow 0$;
while $z \leq 1$ **do**
  if *s is empty or $g(z)$ is strictly larger than the first coordinate of the the top element of s* **then**
    while *s has at least two elements and the slope from (the second-to-top element of s) to (the top element of s) is less than the slope from (the top element of s) to $(g(z), h(z))$* **do**
      Pop the top element of s ;
      Pop the top element of t ;
    Push $(g(z), h(z))$ onto s ;
    Push z onto t ;
  $z \leftarrow z + \epsilon'$;
return (s, t)

Recall that we split on two cases based on the values of Z_l and Z_u. Under the easy case, i.e., when $Z_l \geq Z_u$, the algorithm picks $\hat{z}_i = Z_l$. The proof of this case is then identical to the proof of easy case of Proposition 10. Now consider the hard case, i.e., when $Z_l < Z_u$. In this case, the algorithm plays a mixed strategy over two points which it obtains by considering the lattice 2D-curve $\tilde{\mathbf{r}} \triangleq \{g(z), h(z))\}_{z \in \{Z_l, Z_l+\epsilon', \ldots, Z_u\}}$ first, and then finding a point $\mathcal{P}$ on $\texttt{conc-env}(\tilde{\mathbf{r}})$ that lies on the line $h' - \beta = g' - \alpha$, where $\alpha \triangleq g(Z_u) \geq 0$ and $\beta \triangleq h(Z_l) \geq 0$. Since this point is on the upper-concave envelope, it is a convex combination of two points on the lattice curve $\tilde{\mathbf{r}}$; let's write it as $\mathcal{P} = \lambda \mathcal{P}_1 + (1-\lambda)\mathcal{P}_2$, where $\lambda \in [0,1]$, and $\mathcal{P}_1$ and $\mathcal{P}_2$ are two points on the lattice curve corresponding to lattice coordinates values $z^{(i)} \in \{Z_l, Z_l + \epsilon', \ldots, Z_u\}$ for $i = \{1, 2\}$. The definition of adversary's positive region and all of its properties are exactly as before (Lemma 11). The proof of the hard case is finished by proving a slightly modified version of Lemma 12: under the mentioned two-point mixed strategy of the algorithm (i.e., playing $\big(g(z^{(1)}), h(z^{(1)})\big)$ w.p. λ and $\big(g(z^{(2)}), h(z^{(2)})\big)$ w.p. $1-\lambda$), for every $\tilde{x}_i^* \in \{0, \epsilon', 2\epsilon', \ldots, 1\}$, the adversary's positive region contains the point (g', h'), where $g' = g(\tilde{x}_i^*)$ and $h' = h(\tilde{x}_i^*)$. The proof of this lemma is identical to the proof of Lemma 12 and we omit for brevity.

■

3. Strong DR-SM Maximization: Binary-Search Bi-Greedy

Our second result is a fast binary search algorithm, achieving the tight $\frac{1}{2}$-approximation factor (up to additive error δ) in quasi-linear time in n, but only for the special case of `strong DR-SM` functions (a.k.a. DR-submodular); see Theorem 1. This algorithm leverages the coordinate-wise concavity to identify a coordinate-wise *monotone equilibrium condition*.

In each iteration, it hunts for an equilibrium point by using binary search. Satisfying the equilibrium at each iteration then guarantees the desired approximation factor. Formally we propose Algorithm 4. As a technical assumption, we assume $\mathcal{F}$ is Lipschitz continuous

Algorithm 4: Binary-Search Continuous Bi-greedy

input: function $\mathcal{F} : [0,1]^n \to [0,1]$, error $\epsilon > 0$;
output: vector $\hat{\mathbf{z}} = (\hat{z}_1, \ldots, \hat{z}_n) \in [0,1]^n$;
Initialize $\mathbf{X} \leftarrow (0, \ldots, 0)$ and $\mathbf{Y} \leftarrow (1, \ldots, 1)$;
for $i = 1$ *to* n **do**
  if $\frac{\partial \mathcal{F}}{\partial x_i}(0, \mathbf{X}_{-i}) \leq 0$ **then**
    $\hat{z}_i \leftarrow 0$
  else if $\frac{\partial \mathcal{F}}{\partial x_i}(1, \mathbf{Y}_{-i}) \geq 0$ **then**
    $\hat{z}_i \leftarrow 1$
  else
    `// we do binary search.`
    while $Y_i - X_i > \epsilon/n$ **do**
      Let $\hat{z}_i \leftarrow \frac{X_i + Y_i}{2}$;
      if $\frac{\partial \mathcal{F}}{\partial x_i}(\hat{z}_i, \mathbf{X}_{-i}) \cdot (1 - \hat{z}_i) + \frac{\partial \mathcal{F}}{\partial x_i}(\hat{z}_i, \mathbf{Y}_{-i}) \cdot \hat{z}_i > 0$ **then**
        `// we need to increase` $\hat{z}_i$`.`
        Set $X_i \leftarrow \hat{z}_i$;
      else
        `// we need to decrease` $\hat{z}_i$`.`
        Set $Y_i \leftarrow \hat{z}_i$;
  Let $X_i \leftarrow \hat{z}_i$ and $Y_i \leftarrow \hat{z}_i$; `// after this,` $\mathbf{X}$ `and` $\mathbf{Y}$ `will agree at coordinate` i

with some constant $C > 0$, so that we can relate the precision of our binary search with additive error. We arrive at the following theorem, whose proof is postponed to Section 3.1.

Theorem 14 *If $\mathcal{F}(.)$ is non-negative and DR-submodular (a.k.a* `Strong DR-SM`*) and is coordinate-wise Lipschitz continuous with constant $C > 0$, then Algorithm 4 runs in time $O\left(n \log\left(\frac{n}{\epsilon}\right)\right)$ and is a deterministic $\frac{1}{2}$-approximation algorithm up to $O(\epsilon)$ additive error, i.e., returns $\hat{\mathbf{z}} \in [0,1]^n$ s.t.*

$$2\mathcal{F}(\hat{\mathbf{z}}) \geq \mathcal{F}(\mathbf{x}^*) - 2C\epsilon \ , \qquad \textit{where } \mathbf{x}^* \in \operatorname*{argmax}_{\boldsymbol{x} \in [0,1]^n} \mathcal{F}(\boldsymbol{x}) \textit{ is the optimal solution.}$$

Running Time. Clearly the binary search terminates in $O(\log(n/\epsilon))$ steps following the fact that $Y_i - X_i \leq 1$ in Algorithm 4. Hence the total running time/query complexity is $O(n \log(n/\epsilon))$.

3.1. Analysis of the Binary-Search Bi-Greedy (proof of Theorem 14)

We start by the following technical lemma, which is used in various places of our analysis. The proof is immediate by `strong DR-SM` property (Definition 1).

Lemma 15 *For any $\mathbf{y}, \mathbf{z} \in [0,1]^n$ such that $\mathbf{y} \leq \mathbf{z}$, we have $\frac{\partial \mathcal{F}}{\partial x_i}(\mathbf{y}) - \frac{\partial \mathcal{F}}{\partial x_i}(\mathbf{z}) \geq 0, \forall i$.*

Proof We rewrite this difference as a sum over integrals of the second derivatives:

$$\frac{\partial \mathcal{F}}{\partial x_i}(\mathbf{y}) - \frac{\partial \mathcal{F}}{\partial x_i}(\mathbf{z}) = \sum_{j=1}^{n} \left[\begin{array}{l} \frac{\partial \mathcal{F}}{\partial x_i}(y_1, \ldots, y_{j-1}, y_j, z_{j+1}, \ldots, z_n) \\ -\frac{\partial \mathcal{F}}{\partial x_i}(y_1, \ldots, y_{j-1}, z_j, z_{j+1}, \ldots, z_n) \end{array} \right]$$
$$= \sum_{j=1}^{n} \int_{y_j}^{z_j} -\frac{\partial^2 \mathcal{F}}{\partial x_i \partial x_j}(y_1, \ldots, y_{j-1}, w, z_{j+1}, \ldots, z_n) dw \geq 0 \ .$$

To see why the last inequality holds, note that `strong DR-SM` Proposition 19 implies that all of the second derivatives of $\mathcal{F}$ are always non-positive. As $\forall i : z_i \geq y_i$, the RHS is non-negative. ■

A modified zero-sum game. We follow the same approach and notations as in the proof of Theorem 4 (Section 2.1). Suppose $\mathbf{x}^*$ is the optimal solution. For each coordinate i we again define a two-player zero-sum game between `ALG` and `ADV`, where the former plays $\hat{z}_i$ and the latter plays x_i^*. The payoff matrix for the `strong DR-SM` case, denoted by $\mathcal{V}_S^{(i)}(\hat{z}_i, x_i^*)$ is defined as before (Equation (1)); the only major difference is we redefine $h(.)$ and $g(.)$ to be the following functions,:

$$g(z) \triangleq \mathcal{F}(z, \mathbf{X}_{-i}^{(i-1)}) - \mathcal{F}(0, \mathbf{X}_{-i}^{(i-1)}) \quad , \quad h(z) \triangleq \mathcal{F}(z, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(1, \mathbf{Y}_{-i}^{(i-1)}).$$

Now, similar to Lemma 9, we have a lemma that shows how to prove the desired approximation factor using the above zero-sum game. The proof is exactly as Lemma 9 and is omitted for brevity.

Lemma 16 *Suppose* $\forall i \in [n] : \mathcal{V}_S^{(i)}(\hat{z}_i, x_i^*) \geq -\delta/n$ *for constant* $\delta > 0$. *Then* $2\mathcal{F}(\hat{\mathbf{z}}) \geq \mathcal{F}(\mathbf{x}^*) - \delta$.

Analyzing zero-sum games. We show that $\mathcal{V}_S^{(i)}(\hat{z}_i, x_i^*)$ is lower-bounded by a small constant, and then by using Lemma 16 we finish the proof. The formal proof uses both ideas similar to those of Buchbinder et al. (2015b), as well as new ideas on how to relate the algorithm's equilibrium condition to the value of the two-player zero-sum game.

Proposition 17 *If* `ALG` *plays the strategy* $\hat{z}_i$ *described in Algorithm 4, then* $\mathcal{V}_S^{(i)}(\hat{z}_i, x_i^*) \geq -2C\epsilon/n$.

Proof [*proof of Proposition 17*] Consider the easy case where $\frac{\partial \mathcal{F}}{\partial x_i}(0, \mathbf{X}_{-i}^{(i-1)}) \leq 0$ (and therefore we have $\frac{\partial \mathcal{F}}{\partial x_i}(0, \mathbf{Y}_{-i}^{(i-1)}) \leq 0$ due to `Strong DR-SM`). In this case, $\hat{z}_i = 0$ and hence $g(\hat{z}_i) = g(0) = 0$. Moreover, because of the `Strong DR-SM` property,

$$h(0) = \mathcal{F}(0, \mathbf{Y}_{-i}^{(i-1)}) - \mathcal{F}(1, \mathbf{Y}_{-i}^{(i-1)}) \geq -\frac{\partial \mathcal{F}}{\partial x_i}(0, \mathbf{Y}_{-i}^{(i-1)}) \geq 0,$$
$$h(x_i^*) - h(0) \leq g(x_i^*) - g(0) \leq x_i^* \cdot \frac{\partial \mathcal{F}}{\partial x_i}(0, \mathbf{X}_{-i}^{(i-1)}) \leq 0,$$

and therefore $\mathcal{V}_S^{(i)}(\hat{z}_i, x_i^*) = \frac{1}{2}g(0) + \frac{1}{2}h(0) - \max\left(g(x_i^*) - g(0), h(x_i^*) - h(0)\right) \geq 0$. The other easy case is when $\frac{\partial \mathcal{F}}{\partial x_i}(1, \mathbf{Y}_{-i}^{(i-1)}) \geq 0$ (and therefore $\frac{\partial \mathcal{F}}{\partial x_i}(1, \mathbf{X}_{-i}^{(i-1)}) \geq 0$, again because of `Strong DR-SM`). In this case $\hat{z}_i = 1$ and a similar proof shows $\mathcal{V}_S^{(i)}(1, x_i^*) \geq 0$.

The only remaining case (the not-so-easy one) is when

$$\frac{\partial \mathcal{F}}{\partial x_i}(0, \mathbf{X}_{-i}^{(i-1)}) > 0 \quad \text{and} \quad \frac{\partial \mathcal{F}}{\partial x_i}(1, \mathbf{Y}_{-i}^{(i-1)}) < 0.$$

In this case, Algorithm 4 runs the binary search and ends up at a point $\hat{z}_i$. We first show that $f(z) \triangleq \frac{\partial \mathcal{F}}{\partial x_i}(z, \mathbf{X}_{-i})(1-z) + \frac{\partial \mathcal{F}}{\partial x_i}(z, \mathbf{Y}_{-i})z$ is monotone non-increasing in z. To see the monotonicity,

$$f'(z) = (1-z)\frac{\partial^2 \mathcal{F}}{\partial {x_i}^2}(z, \mathbf{X}_{-i}) + z\frac{\partial^2 \mathcal{F}}{\partial {x_i}^2}(z, \mathbf{Y}_{-i}) + \left(\frac{\partial \mathcal{F}}{\partial x_i}(z, \mathbf{Y}_{-i}) - \frac{\partial \mathcal{F}}{\partial x_i}(z, \mathbf{X}_{-i})\right) \leq 0 \ ,$$

where the inequality holds due to `strong DR-SM` and the fact that all of the Hessian entries (including diagonal) are non-positive. Because of the monotonicity and continuity of the equilibrium condition of the binary search, there exists $\tilde{z}$ that is (ϵ/n)-close to $\hat{z}_i$ and $\frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{X}_{-i})(1-\tilde{z}) + \frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{Y}_{-i})\tilde{z} = 0$. By a straightforward calculation, using the Lipschitz continuity of $\mathcal{F}$ with constant C and knowing that $|\tilde{z} - \hat{z}_i| \leq \epsilon/n$, we have:

$$\mathcal{V}_S^{(i)}(\hat{z}_i, x_i^*) = \frac{1}{2}g(\hat{z}_i) + \frac{1}{2}h(\hat{z}_i) - \max\left(g(x_i^*) - g(\hat{z}_i), h(x_i^*) - h(\hat{z}_i)\right) \geq \mathcal{V}_S^{(i)}(\tilde{z}, x_i^*) - \frac{2C\epsilon}{n} \ .$$

So, we only need to show $\mathcal{V}_S^{(i)}(\tilde{z}, x_i^*) \geq 0$. Let $\alpha \triangleq \frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{X}_{-i}^{(i-1)})$ and $\beta \triangleq -\frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{Y}_{-i}^{(i-1)})$. Because of Theorem 15, $\alpha + \beta \geq 0$. Moreover, $\alpha(1-\tilde{z}) = \beta\tilde{z}$, and therefore we should have $\alpha \geq 0$ and $\beta \geq 0$. We now have two cases:

Case 1 ($\tilde{\mathbf{z}} \geq \mathbf{x_i^*}$): $g(x_i^*) - g(\tilde{z}) \leq h(x_i^*) - h(\tilde{z})$ due to `strong DR-SM` and that $\tilde{z} \geq x_i^*$, so:

$$\begin{aligned}
\mathcal{V}_S^{(i)}(\tilde{z}, x_i^*) &= \frac{1}{2}g(\tilde{z}) + \frac{1}{2}h(\tilde{z}) + \left(h(\tilde{z}) - h(x_i^*)\right) \\
&= \frac{1}{2}\int_0^{\tilde{z}} \frac{\partial \mathcal{F}}{\partial x_i}(x, \mathbf{X}_{-i}^{(i-1)})dx + \frac{1}{2}\int_{\tilde{z}}^1 -\frac{\partial \mathcal{F}}{\partial x_i}(x, \mathbf{Y}_{-i}^{(i-1)})dx + \int_{\tilde{z}}^{x_i^*} -\frac{\partial \mathcal{F}}{\partial x_i}(x, \mathbf{Y}_{-i}^{(i-1)}) \\
&\overset{(1)}{\geq} \frac{\tilde{z}}{2} \cdot \frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{X}_{-i}^{(i-1)}) + \frac{(1-\tilde{z})}{2} \cdot \left(-\frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{Y}_{-i}^{(i-1)})\right) + (x_i^* - \tilde{z})\left(-\frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{Y}_{-i}^{(i-1)})\right) \\
&= \frac{\tilde{z}\alpha}{2} + \frac{(1-\tilde{z})\beta}{2} + (x_i^* - \tilde{z})\beta \\
&\overset{(2)}{\geq} \frac{\tilde{z}\alpha}{2} + \frac{(1-\tilde{z})\beta}{2} - \tilde{z}\beta \\
&\overset{(3)}{=} \frac{\alpha^2}{2(\alpha+\beta)} + \frac{\beta^2}{2(\alpha+\beta)} - \frac{\alpha\beta}{(\alpha+\beta)} = \frac{(\alpha-\beta)^2}{2(\alpha+\beta)} \geq 0,
\end{aligned}$$

where inequality (1) holds due to the coordinate-wise concavity of $\mathcal{F}$, inequality (2) holds as $\beta \geq 0$ and $x_i^* \geq 0$, and equality (3) holds as $\beta\tilde{z} = \alpha(1-\tilde{z})$.

Case 2 ($\tilde{\mathbf{z}} < \mathbf{x}_\mathbf{i}^*$): This case is the reciprocal of Case 1, with a similar proof. Note that $g(x_i^*) - g(\tilde{z}) \geq h(x_i^*) - h(\tilde{z})$ due to `strong DR-SM` and the fact that $\tilde{z} < x_i^*$, so:

$$\begin{aligned}
\mathcal{V}_S^{(i)}(\tilde{z}, x_i^*) &= \frac{1}{2}g(\tilde{z}) + \frac{1}{2}h(\tilde{z}) + (g(\tilde{z}) - g(x_i^*)) \\
&= \frac{1}{2}\int_0^{\tilde{z}} \frac{\partial \mathcal{F}}{\partial x_i}(x, \mathbf{X}_{-i}^{(i-1)})dx + \frac{1}{2}\int_{\tilde{z}}^1 -\frac{\partial \mathcal{F}}{\partial x_i}(x, \mathbf{Y}_{-i}^{(i-1)})dx + \int_{x_i^*}^{\tilde{z}} \frac{\partial \mathcal{F}}{\partial x_i}(x, \mathbf{X}_{-i}^{(i-1)}) \\
&\overset{(1)}{\geq} \frac{\tilde{z}}{2} \cdot \frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{X}_{-i}^{(i-1)}) + \frac{(1-\tilde{z})}{2} \cdot \left(-\frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{Y}_{-i}^{(i-1)})\right) + (\tilde{z} - x_i^*)\left(\frac{\partial \mathcal{F}}{\partial x_i}(\tilde{z}, \mathbf{X}_{-i}^{(i-1)})\right) \\
&= \frac{\tilde{z}\alpha}{2} + \frac{(1-\tilde{z})\beta}{2} + (\tilde{z} - x_i^*)\alpha \\
&\overset{(2)}{\geq} \frac{\tilde{z}\alpha}{2} + \frac{(1-\tilde{z})\beta}{2} + (\tilde{z} - 1)\alpha \\
&\overset{(3)}{=} \frac{\alpha^2}{2(\alpha+\beta)} + \frac{\beta^2}{2(\alpha+\beta)} - \frac{\alpha\beta}{(\alpha+\beta)} = \frac{(\alpha-\beta)^2}{2(\alpha+\beta)} \geq 0,
\end{aligned}$$

where inequality (1) holds due to the coordinate-wise concavity of $\mathcal{F}$, inequality (2) holds as $\alpha \geq 0$ and $x_i^* \leq 1$, and equality (3) holds as $\beta\tilde{z} = \alpha(1-\tilde{z})$. ■

Combining Proposition 17 and Lemma 16 for $\delta = 2C\epsilon$ finishes the analysis and the proof of Theorem 14.

4. Experimental Results

We empirically measure the solution quality of five algorithms. Three of them serve as baslines: (`UNIFORM`) chooses an independent uniform random value in $[0, 1]$ for each coordinate, (`ONE-HALF`) chooses all coordinates to be $1/2$, and (`BMBK`) is the Bi-Greedy algorithm of Bian et al. (2017b). We compare these baselines with our two algorithms based on the continuous double-greedy framework: Algorithm 1 for maximizing `weak DR-SM` continuous submodular functions, which is denoted (`GAME`), and Algorithm 4 for maximizing `strong DR-SM` continuous submodular functions, which is denoted (`BINARY`). To run these two algorithms, we iterate over coordinates in a random order. These algorithms also do not solely rely on oracle access to the function; they invoke one-dimensional optimizers, concave envelopes, and derivatives. We implement the last two (Algorithm 2 and Algorithm 3 in Section 2.2), and numerically compute derivatives by discretization.

We consider two application domains, namely Non-concave Quadratic Programming (NQP) (Bian et al., 2017b; Kim and Kojima, 2003; Luo et al., 2010), under both `strong-DR` and `weak-DR`, and maximization of softmax extension for MAP inference of determinantal point process (Kulesza et al., 2012; Gillenwater et al., 2012). For each experiment, we use $n = 100$ dimensional functions. Moreover, we use a randomized generative model in each experiment to create samples of our problem instances; each experiment consists of 20 such instances (i.e. a 20 sample Monte Carlo experiment) to estimate the performance our algorithms and baselines. We then report the mean and the variance for three different quantities: objective value, running time (in seconds), and number of oracle calls to the function. We further complement our results by reporting the objective value's confidence

intervals through box-and-whisker plots in all of our experiments. Our experiments are implemented in python. See below for the detailed specifics of each experiment.

`Strong-DR` Non-concave Quadratic Programming (NQP) We generated synthetic functions of the form $\mathcal{F}(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x} + c$. We generated $\mathbf{H} \in \mathbb{R}^{n\times n}$ as a matrix with every entry *uniformly distributed* in $[-1, 0]$, and then symmetrized $\mathbf{H}$. The choice of the uniform distribution is just for the purpose of exposition. We then generated $\mathbf{h} \in \mathbb{R}^n$ as a vector with every entry uniformly distributed in $[0, +1]$. Finally, we solved for the value of c to make $\mathcal{F}(\vec{0}) + \mathcal{F}(\vec{1}) = 0$.

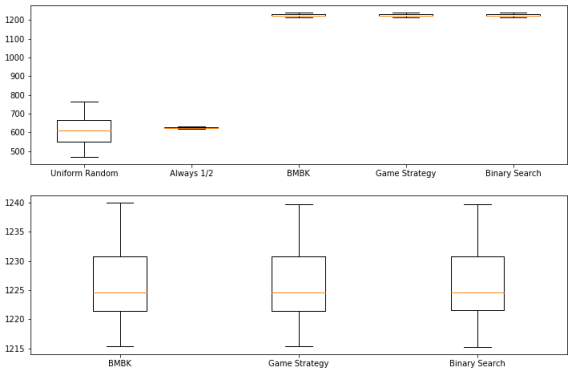
`Weak-DR` Non-concave Quadratic Programming (NQP) This experiment is the same as in the previous subsection, except that the diagonal entries of $\mathbf{H}$ are uniformly distributed in $[0, +1]$ instead, making the resulting function $\mathcal{F}(\mathbf{x})$ only `weak DR-SM` instead.

Softmax extension of Determinantal Point Processes (DPP) We generated synthetic functions of the form $\mathcal{F}(\mathbf{x}) = \log\det(\mathrm{diag}(\mathbf{x})(\mathbf{L} - \mathbf{I}) + \mathbf{I})$, where $\mathbf{L}$ needs to be positive semidefinite. We generated $\mathbf{L}$ in the following way. First, we generate each of the n eigenvalues by drawing a uniformly random number in $[-0.5, 1.0]$ and taking that power of e. This yields a diagonal matrix $\mathbf{D}$. We then generate a random unitary matrix $\mathbf{V}$ and then set $\mathbf{L} = \mathbf{V}\mathbf{D}\mathbf{V}^T$. By construction, $\mathbf{L}$ is positive semidefinite and has the specified eigenvalues.

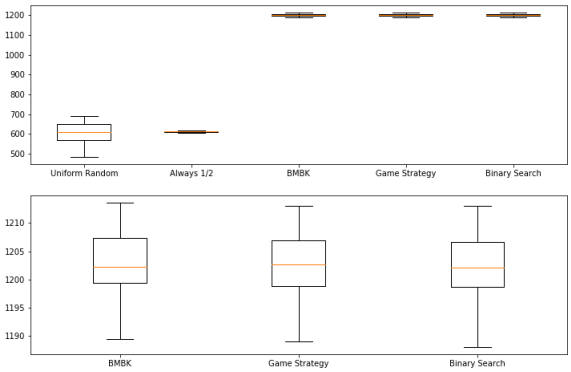
Interpretations from our experiments. The results of our experiments are in Table 2, Table 3, and Table 4, and the corresponding box-and-whisker plots are in Figure 3. The data suggests that for all three experiments the three algorithms obtain very similar objective values, and they all outperform the baselines we considered. For example, in the `weak-DR` NQP experiment, the upper and lower quartiles are distant by roughly 10, while the mean values of the three algorithms deviate by less than 1. We remind the reader that all of our experiments use synthetic and randomly generated data, and the observed tie can be an artifact of this choice. We should also note that in a contemporaneous work (which apeared after an earlier conference version of our paper), Bian et al. (2019) proposed an alternate optimal algorithm for `strong DR-SM` functions; moreover, they provide more extensive experiments comparing our Algorithm 4 with their method, on real world data. We leave studying the experimental performance of Algorithm 1 on these application domains as future work.

	NQP, $\forall i, j : H_{i,j} \leq 0$, (`strong-DR`)	NQP, $\forall i \neq j : H_{i,j} \leq 0$, (`weak-DR`)	Softmax Ext. (`strong-DR`)
`UNIFORM`	612.559 ± 81.096	604.579 ± 55.866	15.057 ± 2.402
`ONE-HALF`	625.108 ± 3.260	612.541 ± 3.326	15.311 ± 2.425
`BMBK`	1225.577 ± 6.310	1202.393 ± 6.937	24.754 ± 4.154
`GAME`	1225.593 ± 6.279	1202.523 ± 6.722	24.755 ± 4.153
`BINARY`	1225.636 ± 6.293	1201.852 ± 6.989	24.639 ± 4.129

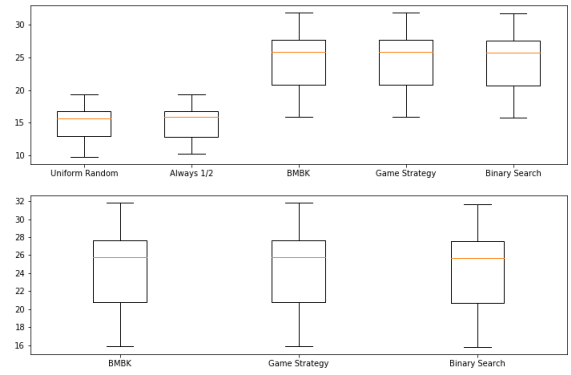
Table 2: Average objective value (plus/minus standard deviation) of $T = 20$ repeated trials, with dimension $n = 100$.



(a) `Strong DR-SM` NQP



(b) `Weak DR-SM` NQP



(c) `Strong DR-SM` Softmax

Figure 3: Box and whisker plots of our experimental results. Lower graph in each section zooms on `BMBK`, `GAME`, and `BINARY` to show more detail.

	NQP, $\forall i, j : H_{i,j} \leq 0$, (`strong-DR`)	NQP, $\forall i \neq j : H_{i,j} \leq 0$, (`weak-DR`)	Softmax Ext. (`strong-DR`)
`BMBK`	0.491 ± 0.080	0.465 ± 0.028	45.729 ± 16.306
`GAME`	1.108 ± 0.408	1.093 ± 0.166	45.845 ± 16.014
`BINARY`	0.077 ± 0.050	0.063 ± 0.003	5.752 ± 1.552

Table 3: Average time in seconds (plus/minus standard deviation) of $T = 20$ repeated trials, with dimension $n = 100$. `UNIFORM` and `ONE-HALF` (omitted) ran in 2ms or less.

	NQP, $\forall i, j : H_{i,j} \leq 0$, (`strong-DR`)	NQP, $\forall i \neq j : H_{i,j} \leq 0$, (`weak-DR`)	Softmax Ext. (`strong-DR`)
`UNIFORM`	1 ± 0	1 ± 0	1 ± 0
`ONE-HALF`	1 ± 0	1 ± 0	1 ± 0
`BMBK`	20801 ± 0	20801 ± 0	20801 ± 0
`GAME`	43489.4 ± 865.980	48716.2 ± 7350.117	20666.2 ± 580.952
`BINARY`	2801 ± 0	2801 ± 0	2801 ± 0

Table 4: Average number of oracle calls (plus/minus standard deviation) of $T = 20$ repeated trials, with dimension $n = 100$.

5. Conclusion

We proposed a tight approximation algorithm for continuous submodular maximization, and a quasilinear time tight approximation algorithm for the special case of DR-submodular maximization. Our experiments also verify the applicability of these algorithms in practical domains in machine learning. One interesting avenue for future research is to generalize our techniques to maximization over any arbitrary separable convex set, which would broaden the application domains.

Acknowledgments

The authors would also like to thank Jan Vondrák and Amin Karbasi for helpful comments and discussions on an earlier draft of this work.

References

Anestis Antoniadis, Irène Gijbels, and Mila Nikolova. Penalized likelihood regression for generalized linear models with non-quadratic penalties. *Annals of the Institute of Statistical Mathematics*, 63(3):585–615, 2011.

Francis Bach et al. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends in Machine Learning*, 6(2-3):145–373, 2013.

An Bian, Kfir Levy, Andreas Krause, and Joachim M Buhmann. Continuous DR-submodular maximization: Structure and algorithms. In *Advances in Neural Information Processing Systems*, pages 486–496, 2017a.

Andrew An Bian, Baharan Mirzasoleiman, Joachim Buhmann, and Andreas Krause. Guaranteed non-convex optimization: Submodular maximization over continuous domains. In *Artificial Intelligence and Statistics*, pages 111–120, 2017b.

Yatao Bian, Joachim Buhmann, and Andreas Krause. Optimal continuous dr-submodular maximization and applications to provable mean field inference. In *International Conference on Machine Learning*, pages 644–653, 2019.

Niv Buchbinder and Moran Feldman. Deterministic algorithms for submodular maximization problems. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 392–403. SIAM, 2016.

Niv Buchbinder, Moran Feldman, and Roy Schwartz. Online submodular maximization with preemption. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1202–1216. Society for Industrial and Applied Mathematics, 2015a.

Niv Buchbinder, Moran Feldman, Joseph Seffi, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM Journal on Computing*, 44(5):1384–1402, 2015b.

Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40 (6):1740–1766, 2011.

Lin Chen, Hamed Hassani, and Amin Karbasi. Online continuous submodular maximization. *arXiv preprint arXiv:1802.06052*, 2018.

Josip Djolonga and Andreas Krause. From map to marginals: Variational inference in bayesian submodular models. In *Advances in Neural Information Processing Systems*, pages 244–252, 2014.

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

Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 570–579. IEEE, 2011.

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.

Alkis Gotovos, Amin Karbasi, and Andreas Krause. Non-monotone adaptive submodular maximization. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

Ronald L Graham. An efficient algorith for determining the convex hull of a finite planar set. *Information processing letters*, 1(4):132–133, 1972.

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.

Hamed Hassani, Mahdi Soltanolkotabi, and Amin Karbasi. Gradient methods for submodular maximization. In *Advances in Neural Information Processing Systems*, pages 5843–5853, 2017.

Shinji Ito and Ryohei Fujimaki. Large-scale price optimization via network flow. In *Advances in Neural Information Processing Systems*, pages 3855–3863, 2016.

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

Michael Kapralov, Ian Post, and Jan Vondrák. Online submodular welfare maximization: Greedy is optimal. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1216–1225. SIAM, 2013.

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.

Andreas Krause and Daniel Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*, pages 71–104. Cambridge University Press, 2014.

Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286, 2012.

Chengtao Li, Suvrit Sra, and Stefanie Jegelka. Fast mixing markov chains for strongly rayleigh measures, dpps, and constrained sampling. In *Advances in Neural Information Processing Systems*, pages 4188–4196, 2016.

Zhi-Quan Luo, Wing-Kin Ma, Anthony Man-Cho So, Yinyu Ye, and Shuzhong Zhang. Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Processing Magazine*, 27(3):20–34, 2010.

Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *Advances in Neural Information Processing Systems*, pages 2049–2057, 2013.

Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Stochastic conditional gradient methods: From convex minimization to submodular maximization. *arXiv preprint arXiv:1804.09554*, 2018.

Tim Roughgarden and Joshua R Wang. An optimal learning algorithm for online unconstrained submodular maximization. In *To Appear in Proceedings of the 31st Conference on Learning Theory (COLT)*, 2018.

Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.

Tasuku Soma and Yuichi Yoshida. A generalization of submodular cover via the diminishing return property on the integer lattice. In *Advances in Neural Information Processing Systems*, pages 847–855, 2015.

Tasuku Soma and Yuichi Yoshida. Non-monotone dr-submodular function maximization. In *AAAI*, volume 17, pages 898–904, 2017.

Matthew Staib and Stefanie Jegelka. Robust budget allocation via continuous submodular functions. *arXiv preprint arXiv:1702.08791*, 2017.

Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM Journal on Computing*, 42(1):265–304, 2013.

Jian Zhang, Josip Djolonga, and Andreas Krause. Higher-order inference for multi-class log-supermodular models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1859–1867, 2015.

Appendix A. More details on different application domains

Here is a list containing further details about applications in machine learning, electrical engineering and other application domains.

Special Class of Non-Concave Quadratic Programming (NQP).

- The objective is to maximize $\mathcal{F}(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x} + c$, where off-diagonal entries of $\mathbf{H}$ are non-positive (and hence these functions are `Weak DR-SM`).
- Minimization of this function (or equivalently maximization of this function when off-diagonal entries of $\mathbf{H}$ are non-negative) has been studied in Kim and Kojima (2003) and Luo et al. (2010), and has applications in communication systems and detection in MIMO channels (Luo et al., 2010).
- Another application of quadratic submodular optimization is large-scale price optimization on the basis of demand forecasting models, which has been studied in Ito and Fujimaki (2016). They show the price optimization problem is indeed an instance of weak-DR minimization.

Map Inference for Determinantal Point Processes (DPP) & Its Softmax-Extension.

- DPP are probabilistic models that arise in statistical physics and random matrix theory, and their applications in machine learning have been recently explored, e.g. (Kulesza et al., 2012).
- DPPs can be used as generative models in applications such as text summarization, human pose estimation, or news threading tasks (Kulesza et al., 2012).

- A discrete DPP is a distribution over sets, where $p(S) \sim \det(A_S)$ for a given PSD matrix A. The log-likelihood estimation task corresponds to picking a set $\hat{S} \in \mathcal{P}$ (feasible set, e.g. a matching) that maximizes $f(S) = \log(\det(A_S))$. This function is non-monotone and submodular. Note that as a technical condition to apply bi-greedy algorithms, we require that $\det(A) \geq 1$ (implying $f(\vec{1}) \geq 0$).

- The approximation question was studied in (Gillenwater et al., 2012). Their idea is to first find a fractional solution for a continuous extension (hence a continuous submodular maximization step is required) and then rounding the solution. However, they sometimes need a fractional solution in $\text{conv}(\mathcal{P})$ (so, the optimization task sometimes fall out of the hypercube, making rounding more complicated).

- Beyond multilinear extension, there are other continuous extensions used in this literature. One such extension is called the *softmax extension* (Gillenwater et al., 2012; Bian et al., 2017a):

$$\mathcal{F}(\mathbf{x}) = \log \mathbf{E}_{S \sim \mathcal{I}_{\mathbf{x}}}[\exp(f(S))] = \log \det \left(\text{diag}(\mathbf{x})(A - I) + I\right) \ ,$$

  where $\mathcal{I}_{\mathbf{x}}$ is the independent distribution with marginals $\mathbf{x}$ (i.e. each item i is independently in the set w.p. x_i). The advantage of softmax extension over multi-linear extension is in its computation; multi-linear extension can only be computed approximately (up to additive ϵ error), however softmax extension has a closed-form for DPPs and can be computed exactly (cf. Kulesza et al. (2012) and Bian et al. (2017b)).

- $\mathcal{F}(\mathbf{x})$ is `Strong DR-SM` and non-monotone (Bian et al., 2017a). In almost all machine learning applications, the rounding works on an unrestricted problem. Hence the optimization that needs to be done is `Strong DR-SM` optimization over unit hypercube.

- One can think of adding a regularizer term $\lambda\|\mathbf{x}\|^2$ to the log-likelihood objective function to avoid overfitting. In that case, the underlying fractional problem becomes a `Weak DR-SM` optimization over the unit hypercube when λ is large enough.

Log-Submodularity and Mean-Field Inference.

- Another probabilistic model that generalizes DPP and all other strong Rayleigh measures (Li et al., 2016; Zhang et al., 2015) is the class of *log-submodular* distributions over sets, i.e. $p(S) \sim \exp(f(S))$ where $f(\cdot)$ is a discrete submodular functions. MAP inference over this distribution has applications in machine learning and beyond (Djolonga and Krause, 2014).

- One variational approach towards this MAP inference task is to do *mean-field inference* to approximate the distribution p with a product distribution $\mathbf{x} \in [0,1]^n$, i.e. finding $\mathbf{x}^*$ that:

$$\mathbf{x}^* \in \operatorname*{argmax}_{\mathbf{x} \in [0,1]^n} \ \mathbb{H}(\mathbf{x}) - \mathbf{E}_{S \sim \mathcal{I}_x}[\log p(S)] = \operatorname*{argmin}_{\mathbf{x} \in [0,1]^n} \ \text{KL}(\mathbf{x}||p) \ ,$$

  where $\text{KL}(\mathbf{x}||p) = \mathbf{E}_{S \sim \mathcal{I}}[\frac{\log \mathcal{I}_x(S)}{\log p(S)}]$.

- The function $\mathcal{F}(\mathbf{x}) = \mathbb{H}(\mathbf{x}) - \mathbf{E}_{S \sim \mathcal{I}_x}[\log p(S)]$ is `Strong DR-SM` (Bian et al., 2017a).

Revenue Maximization over Social Networks.

- The model was proposed in Bian et al. (2017b) and is a generalization of the revenue maximization problem addressed in Hartline et al. (2008).
- A seller wishes to sell a product to a social network of buyers. We consider restricted seller strategies which freely give (possibly fractional) trial products to buyers: this fractional assignment is our input $\mathbf{x}$ of interest.
- The objective takes two effects into account: (i) the revenue gain from buyers who didn't receive free product, where the revenue function for each such buyer is a non-negative nondecreasing `Weak DR-SM` function and (ii) the revenue loss from those who received free product, where the revenue function for each such buyer is a nonpositive nonincreasing `Weak DR-SM` function. The combination for all buyers is a nonmonotone `Weak DR-SM` function and additionally is nonnegative at $\vec{0}$ and $\vec{1}$.

Cone Extension of Continuous Submodular Maximization.

- Suppose $\mathcal{K}$ is a proper cone. By considering the lattice corresponding to this cone one can generalize DR submodularity to $\mathcal{K}$-DR submodularity (Bian et al., 2017a).
- An interesting application of this cone generalization is minimizing the loss in the logistic regression model with a particular non-separable and non-convex regularizer, as described in (Antoniadis et al., 2011; Bian et al., 2017a). Bian et al. (2017a) show the vanilla version is a $\mathcal{K}$-`Strong DR-SM` function maximization for some particular cone.
- Note that by adding a $\mathcal{K}$-ℓ_2-norm regularizer $\lambda\|\mathbf{Ax}\|^2$, the function will become `Weak DR-SM`, where $\mathbf{A}$ is a matrix with generators of $\mathcal{K}$ as its column. Here is the logistic loss:
$$l(\mathbf{x}, \{y_t\}) = \frac{1}{T}\sum_{t=1}^{T} f_t(\mathbf{x}, y_t) = \frac{1}{T}\sum_{t=1}^{T} \log\big(1 + \exp\big(-y_t \mathbf{x}^T \mathbf{z}^t\big)\big) \ ,$$
where y_t is the label of the t^{th} data-point, $\mathbf{x}$ are the model parameters, and $\{\mathbf{z}^t\}$ are feature vectors of the data-points.

Remark 18 *In many machine learning applications, and in particular MAP inference of DPPs and log-submodular distributions, unless we impose some technical assumptions, the underlying* `Strong DR-SM` *(or* `Weak DR-SM`*) function is* not *necessarily positive (or may not even satisfy the weaker yet sufficient condition $\mathcal{F}(\vec{0}) + \mathcal{F}(\vec{1}) \geq 0$). In those cases, adding a positive constant to the function can fix the issue, but the multiplicative approximation guarantee becomes weaker. However, this trick tends to work in practice since these algorithms tend to be near optimal.*

Appendix B. Equivalent definitions of weakly and strongly DR-SM.

Proposition 19 ((Bian et al., 2017b)) *Suppose $\mathcal{F} : [0,1]^n \to [0,1]$ is continuous and twice differentiable, and $\mathbf{H}$ is the Hessian of $\mathcal{F}$, i.e. $\forall i, j \in [n]$, $H_{ij} \triangleq \frac{\partial^2 \mathcal{F}}{\partial x_i \partial x_j}$. The followings are equivalent:*

1. $\mathcal{F}$ *satisfies the* `weak DR-SM` *property as in Definition 1.*
2. *Continuous submodularity:* $\forall \boldsymbol{x}, \boldsymbol{y} \in [0,1]^n,\ \mathcal{F}(\boldsymbol{x}) + \mathcal{F}(\boldsymbol{y}) \geq \mathcal{F}(\boldsymbol{x} \vee \boldsymbol{y}) + \mathcal{F}(\boldsymbol{x} \wedge \boldsymbol{y})$.
3. $\forall i \neq j \in [n],\ H_{ij} \leq 0$, *i.e., all off-diagonal entries of Hessian are non-positive.*

Also, the following statements are equivalent:

1. $\mathcal{F}$ *satisfies the* `strong DR-SM` *property as in Definition 1.*
2. $\mathcal{F}(.)$ *is coordinate-wise concave along all the coordinates and is continuous submodular, i.e.* $\forall \boldsymbol{x}, \boldsymbol{y} \in [0,1]^n,\ \mathcal{F}(\boldsymbol{x}) + \mathcal{F}(\boldsymbol{y}) \geq \mathcal{F}(\boldsymbol{x} \vee \boldsymbol{y}) + \mathcal{F}(\boldsymbol{x} \wedge \boldsymbol{y})$
3. $\forall i, j \in [n],\ H_{ij} \leq 0$, *i.e., all entries of Hessian are non-positive.*