Submodular maximization and its generalization through an intersection cut lens

Liding Xu, Leo Liberti

In press, Mathematical Programming series B, 2024

Journal name manuscript No.
(will be inserted by the editor)

Submodular maximization and its generalization through an intersection cut lens

Liding Xu · Leo Liberti

November 2, 2023

Abstract We study a mixed-integer set $\mathcal{S} := \{(x,t) \in \{0,1\}^n \times \mathbb{R} : f(x) \geq t\}$ arising in the submodular maximization problem, where f is a submodular function defined over $\{0,1\}^n$. We use intersection cuts to tighten a polyhedral outer approximation of $\mathcal{S}$. We construct a continuous extension $\bar{\mathsf{F}}_f$ of f, which is convex and defined over the entire space $\mathbb{R}^n$. We show that the epigraph $\mathrm{epi}(\bar{\mathsf{F}}_f)$ of $\bar{\mathsf{F}}_f$ is an $\mathcal{S}$-free set, and characterize maximal $\mathcal{S}$-free sets containing $\mathrm{epi}(\bar{\mathsf{F}}_f)$. We propose a hybrid discrete Newton algorithm to compute an intersection cut efficiently and exactly. Our results are generalized to the hypograph or the superlevel set of a submodular-supermodular function over the Boolean hypercube, which is a model for discrete nonconvexity. A consequence of these results is intersection cuts for Boolean multilinear constraints. We evaluate our techniques on max cut, pseudo Boolean maximization, and Bayesian D-optimal design problems within a MIP solver.

Keywords MINLP · submodular maximization · submodular-supermodular functions · intersection cuts · Boolean multilinear functions · D-optimal design

Mathematics Subject Classification (2000) MSC 90C10 · 90C26 · 90C57

1 Introduction

A function $f : \{0,1\}^n \to \mathbb{R}$ is called *submodular (over* $\{0,1\}^n$*)*, if for all $x, x' \in \{0,1\}^n$, one has $f(x) + f(x') \geq f(\max(x,x')) + f(\min(x,x'))$, where max and min are element-wise operators. We consider the submodular maximization problem:

$$\max_{t \in \mathbb{R}} t \quad \text{s.t.} \quad f(x) \geq t, \quad x \in \{0,1\}^n \cap \mathcal{X}. \tag{1}$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ is a set describing additional constraints. We study valid inequalities for the mixed-integer set $\mathrm{hypo}_{\{0,1\}^n}(f) := \{(x,t) \in \{0,1\}^n \times \mathbb{R} : f(x) \geq t\}$, which we call the hypograph of f over the Boolean hypercube $\{0,1\}^n$ (or, for brevity, the Boolean-hypograph of f).

The maximization of arbitrary submodular functions (i.e., Eq. (1)) can be reduced to a Mixed-Integer Linear Program (MILP) with exponentially many linear inequalities [70]. No polynomial-time algorithm is yet known to separate these inequalities. The Benders-like exact approach based on a branch-and-cut algorithm proposed in [32] provides global dual bounds for primal solutions, and achieves a finite convergence rate.

Many submodular maximization problems (e.g., max cut with positive edge weights [80], D-optimal design [78], and utility maximization [5]) have natural MILP or Mixed-Integer Nonlinear Programming (MINLP) formulations, which can be solved using general-purpose global optimization solvers. The algorithm underlying these solvers is typically a branch-and-cut algorithm,

Liding Xu, Leo Liberti
LIX CNRS, École Polytechnique, Institut Polytechnique de Paris, Palaiseau, 91128, France
E-mail: liding.xu@polytechnique.edu, liberti@lix.polytechnique.fr

which uses polyhedral outer approximations to construct LP relaxations [17,18,84]. For submodular maximization problems with convex MINLP formulations, a state-of-the-art algorithm also uses polyhedral outer approximations [26].

Intersection cuts can be used to strengthen polyhedral outer approximations of a nonconvex set $\mathcal{S}$ that is considered hard to optimize over. This paper considers the following setting of separating intersection cuts. There exists a point $\tilde{z}$ not in $\mathcal{S}$, and we want to strengthen the outer approximation of $\mathcal{S}$ by finding a hyperplane separating $\tilde{z}$. In order to do so, we should find two key ingredients [28] for constructing intersection cuts: a simplicial cone containing $\mathcal{S}$ with apex $\tilde{z}$, and an $\mathcal{S}$-free set, which is a convex set that does not contain any interior point of $\mathcal{S}$. Moreover, (inclusion-wise) maximal $\mathcal{S}$-free sets generate strong intersection cuts not dominated by other intersection cuts.

Intersection cuts were initially devised in the continuous setting (the papers [87], cited in [54, Ch. III], appeared before the classic paper [12]), where they could approximate the hypograph $\mathcal{S}$ of a convex function over a polytope. There is a unique maximal $\mathcal{S}$-free set: the epigraph of that convex function. Later, intersection cuts were used in the discrete setting [12], where $\mathcal{S}$ is a lattice. Several more families of lattice-free sets (e.g., splits, triangles, and spheres [28,60]) were described later.

The submodular maximization problem plays an intermediate role between these settings. On the one hand, the submodular function f is defined over the Boolean hypercube $\{0,1\}^n$. Therefore, the graph of f projected on $\mathbb{R}^n$ is a subset of a lattice. On the other hand, as a discrete analogue to convex functions, f has a convex (thus continuous) extension over the hypercube $[0,1]^n$, namely the Lovász extension [62]. We can extend the Lovász extension to a convex function, which we call $\bar{\mathsf{F}}_f$, over the entire n-dimensional Euclidean space $\mathbb{R}^n$. This (continuous) function $\bar{\mathsf{F}}_f$ inherits a rich combinatorial structure from f.

A function over $\{0,1\}^n$ is called a *submodular-supermodular* (SS) function, if it is the difference of two submodular functions over $\{0,1\}^n$. SS functions generalize submodular functions, which are also discrete analogs of difference-of-convex (DC) functions. The epigraphs/superlevel sets of DC functions can represent various nonconvex sets, e.g., quadratic sets [67] and signomial-term sets [90]. This representation facilitates the derivation of intersection cuts [81]. In fact, SS functions may represent some discrete nonconvex functions arising in combinatorial optimization. For example, we will show that any Boolean multilinear function is an SS function.

In this paper, we first consider $\mathcal{S}$ as the Boolean-hypograph of the submodular function f. We use convex extensions of f in order to construct some $\mathcal{S}$-free sets. The Boolean-hypograph set $\text{hypo}_{\{0,1\}^n}(f)$ is a specialization of the constraint set $\{(x,t) \in \{0,1\}^n \times \mathbb{R} : f_1(x) - f_2(x) \geq \ell t\}$, where f_1 and f_2 are two submodular functions over $\{0,1\}^n$ and $\ell \in \{0,1\}$. Then, we consider $\mathcal{S}$ as this general constraint set and extend our results to handle this general case. Finally, we propose an efficient algorithm to compute intersection cuts derived from $\mathcal{S}$-free sets. To the best of our knowledge, intersection cuts have not been applied directly to approximate problems with submodular and/or supermodular structures.

We implement intersection cuts within the `SCIP` solver [17] and test them on MAX CUT, PSEUDO BOOLEAN MAXIMIZATION, and BAYESIAN D-OPTIMAL DESIGN problems. We show the strengths and weaknesses of intersection cuts under these different settings.

1.1 Contributions

Our primary contribution is the construction of $\mathcal{S}$-free sets for $\mathcal{S}$ as Boolean-hypographs of submodular functions. We show that a maximal $\mathcal{S}$-free set $\mathcal{C} \times \mathbb{R}$ can be lifted from a maximal $\{0,1\}^n$-free set. We also give an alternative construction of $\mathcal{S}$-free sets by exploiting the submodularity. We relate the analytical properties of $\bar{\mathsf{F}}_f$ to its combinatorial properties, which inherit those of the Lovász extension. We show that the epigraph $\text{epi}(\bar{\mathsf{F}}_f)$ of $\bar{\mathsf{F}}_f$ is an $\mathcal{S}$-free set that is larger than the epigraph of the Lovász extension. However, unlike in the continuous setting, $\text{epi}(\bar{\mathsf{F}}_f)$ is not maximally $\mathcal{S}$-free. We give necessary and sufficient conditions on maximal $\mathcal{S}$-free sets that contain $\text{epi}(\bar{\mathsf{F}}_f)$.

The second contribution is the computation of intersection cuts. We reduce the intersection cut separation problem to solving univariate nonlinear equations, which we achieve by a hybrid discrete Newton algorithm like [49]. We show that facets of $\text{epi}(\bar{\mathsf{F}}_f)$ can be separated in strongly

polynomial time. This implies that the (sub)-gradients required by the Newton algorithm can be computed in a strongly polynomial time. The hybrid discrete Newton algorithm finds a zero point of a univariate nonlinear equation in a finite number of steps. By contrast, the conventional bisection algorithm only guarantees ϵ-approximated solutions for $\epsilon > 0$.

Lastly, we extend the previous findings to constraint sets involving SS functions. We show that any Boolean multilinear function is an SS function. This result yields intersection cuts for multilinear constraints in binary variables.

1.2 Literature review

The classical definition of submodular set functions [62] is equivalent to the definition of submodular functions over the Boolean hypercube through Boolean indicator-characterization of subsets. The latter definition is used in this paper, and it can be generalized over any Cartesian product of subsets of $\mathbb{R}$ [85]. The work of Jack Edmonds [43] plays a prominent role in the study of the combinatorial properties of submodular functions. We refer to [80] for basic concepts and definitions. The convex envelope of a submodular function f is its Lovász extension [11,62]. Submodular functions are a subclass of discrete convex functions, and we refer to [68] for more details about discrete convex analysis.

The *base inequalities* [70] are a class of valid linear inequalities for the Boolean-hypographs of general submodular functions. For a class of special submodular functions, lifting procedures [5,83] can strengthen the base inequalities. Given a point not in the Boolean-hypograph, the base inequalities can be separated either using heuristics [5] or a Benders-like framework [32] if the point to be separated is integer. The method defined in [11] combines valid inequalities for the submodular and supermodular components of an SS function. We refer to [9,10,20,21,52,56, 74,82,90,91] for more details about the exploitation of submodular/supermodular functions in mathematical programs. Supermodular polynomials in binary variables are defined and studied in [20,74]. The submodularity of the D-optimal design problem is exploited in [77,82].

As already mentioned, intersection cuts generate valid inequalities for sets that are hard to optimize over. Gomory introduced the corner polyhedron [50], and his celebrated mixed-integer cuts [51] are special intersection cuts derived from split disjunctions [69]. The definition of intersection cuts for arbitrary set $\mathcal{S}$ is due to [42,48]. We refer to [6,7,13,15,30,31,34,41,42, 75] for a more in-depth analysis. The method given in [86] can generate valid inequalities that cut off points outside $\mathcal{S}$-free sets. We refer to [7,16,58,59,64,65] for relevant recent developments in mixed-integer conic programming.

For the cases where the nonconvexity of $\mathcal{S}$ is not just due to integer variables, we refer to [45] for bilevel programs, [19] for outer-product sets, [67,66] for quadratic constraint sets, [90] for signomial-term sets, and [46] for bilinear sets. The method given in [81] constructs intersection cuts for sets arising from factorable programs that contain DC functions [55].

Next, we discuss valid inequalities for polynomial programming, because we use polynomial programs in binary variables as a benchmark in our computational study. In [19], intersection cuts approximate a nonconvex lifted set, namely the outer product set arising from the extended formulation of a polynomial program. Lifted sets link decision variables to auxiliary variables representing (graphs of) monomials up to a given degree. We remark that in most combinatorial optimization problems, decision variables are binaries. The polynomial program of interest is then a Boolean Multilinear Program (BMP). The corresponding lifted set is the *Boolean multilinear set* [35,47], the convex hull of which is the so-called Boolean multilinear polytope. Valid inequalities for the Boolean multilinear polytope may be stronger than those for the convex hull of the outer product set. Various Gomory-Chvátal-based inequalities [37,38,39,40] are valid for the multilinear polytope. The separation and strength of these inequalities depend on the hypergraph representing the underlying sparsity pattern of the multilinear set.

We consider a constrained polynomial program, and assume that some of its constraints are neither integrality constraints nor variable bound constraints. After lifting, those constraints are linear and thus define a convex set $\mathcal{S}_1$. The lifted set $\mathcal{S}_2$ is nonconvex, and $\mathcal{S}_1 \not\subseteq \mathcal{S}_2$. The polynomial program is then equivalent to linear optimization over $\text{conv}(\mathcal{S}_1 \cap \mathcal{S}_2)$. However, in general, $\text{conv}(\mathcal{S}_1 \cap \mathcal{S}_2) \neq \mathcal{S}_1 \cap \text{conv}(\mathcal{S}_2)$, so the convexification of the lifted set may not yield an equivalent convex problem. To address this issue, one attempt is to directly consider $\text{conv}(\mathcal{S}_1 \cap$

$\mathcal{S}_2$) and generate valid inequalities for it. Some work in this sense exists for certain interesting special cases, e.g. the intersection of multilinear sets with additional constraint sets such as cardinality constraints [23]. Another attempt is to consider constraints in projected formulations, e.g., in mixed-integer quadratically constrained quadratic programs [79]. Since the variables of the projected formulation are fewer than those of the extended formulation, this approach is also amenable to computation. In [24,67], intersection cuts for the set defined by a quadratic constraint are derived. If additionally, some of the nonbasic variables of the LP relaxation need to be integer, the monoidal technique [25] can strengthen such intersection cuts.

However, generating valid inequalities for Boolean multilinear constraints, and, more generally, constructing $\mathcal{S}$-free sets for nonlinear constraints on discrete variables, remain problems of considerable interest. In this paper, we look at these questions through a "submodularity lens".

1.3 Notation

We let $[n] := \{1, \ldots, n\}$ for any positive integer n. We denote $\mathcal{B} := \{0,1\}^n$, $\bar{\mathcal{B}} := [0,1]^n$. We assume that $[n]$ is equipped with the natural number order. $\mathbf{1}$ denotes the all-one vector, $\mathbf{1}_j$ denotes the j-th unit vector, and $\mathbf{0}$ denotes the all-zero vector. For $S \subseteq [n]$, we denote by $\mathsf{char}(S) \in \mathcal{B}$ the characteristic vector of S. For vectors a, b, we let (a, b) be their concatenation, and extend this notation naturally to the case where b is a scalar. Given a set $\mathcal{D} \subseteq \mathbb{R}^n$ and a function $g : \mathcal{D} \to \mathbb{R}$, we adopt the usual notation $\mathrm{epi}_{\mathcal{D}}(g), \mathrm{graph}_{\mathcal{D}}(g), \mathrm{hypo}_{\mathcal{D}}(g)$ to denote the epigraph, graph and hypograph of g over $\mathcal{D}$, respectively. For example, $\mathrm{graph}_{\mathcal{D}}(g) := \{(x,t) \in \mathcal{D} \times \mathbb{R} : g(x) = t\}$. When $\mathcal{D}$ is omitted in the subscript, it is assumed to be $\mathbb{R}^n$. For any set $\mathcal{S}$, we denote by $\mathrm{bd}(\mathcal{S})$, $\mathrm{ext}(\mathcal{S})$, $\mathrm{int}(\mathcal{S})$ its boundary, extreme points, interior, respectively. When $\mathcal{S}$ is not full-dimensional, $\mathrm{relint}(\mathcal{S})$ and $\mathrm{relbd}(\mathcal{S})$ denote its relative interior and relative boundary, respectively.

1.4 Outline

The rest of the paper is organized as follows. In Sect. 2, we recall some preliminaries for intersection cuts. In Sect. 3, we study extensions of submodular functions. In Sect. 4, we study the $\mathcal{S}$-free sets for the submodular function. In Sect. 5, we generalize the previous results for sets involving SS functions. In Sect. 6, we consider applications for intersection cuts to Boolean multilinear constraints and Bayesian D-optimal design. In Sect. 7, we propose the hybrid discrete Newton algorithm for computing intersection cuts. In Sect. 8, we analyze the computational results.

2 Intersection cut preliminaries

In this section, we review the basic concept of intersection cuts. Given a nonconvex set $\mathcal{S}$ and a point $\tilde{z} \notin \mathcal{S}$, we show how to construct intersection cuts, which constitute a specific class of valid linear inequalities for $\mathcal{S}$ that separate $\tilde{z}$.

The construction of intersection cuts [28] needs two components: a simplicial cone $\mathcal{R}$ containing $\mathcal{S}$ with apex $\tilde{z}$, and an $\mathcal{S}$-free set $\mathcal{C}$ with $\tilde{z} \in \mathrm{int}(\mathcal{C})$ defined as follows.

Definition 1 Given $\mathcal{S} \subseteq \mathbb{R}^p$, a closed set $\mathcal{C}$ is called (convex) $\mathcal{S}$-free, if $\mathcal{C}$ is convex and $\mathrm{int}(\mathcal{C}) \cap \mathcal{S} = \varnothing$.

Then an intersection cut separates $\tilde{z}$ from $\mathrm{conv}\,(\mathcal{R} \setminus \mathrm{int}(\mathcal{C}))$ (a set which, we note, contains $\mathcal{S}$) as follows. We write the half-space and ray representations of $\mathcal{R}$:

$$\mathcal{R} = \{z \in \mathbb{R}^p : A(z - \tilde{z}) \leq 0\} = \{z \in \mathbb{R}^p : \exists \eta \in \mathbb{R}^p_+ \ z = \tilde{z} + \sum_{j=1}^{p} \eta_j r^j\}, \tag{2}$$

where $A \in \mathbb{R}^{p \times p}$ is an invertible matrix, and r^j is the j-th column of $-A^{-1}$ and an extreme ray of $\mathcal{R}$.

Define the *step length*

$$\eta_j^* := \sup_{\eta_j \geq 0} \{\eta_j : \tilde{z} + \eta_j r^j \in \mathcal{C}\}. \tag{3}$$

The point $\tilde{z}$ is separated by an intersection cut

$$\sum_{j=1}^{p} \frac{1}{\eta_j^*} A_j(z - \tilde{z}) \leq -1. \tag{4}$$

Let $\mathcal{C}, \mathcal{C}^*$ be two $\mathcal{S}$-free sets such that $\mathcal{C} \subseteq \mathcal{C}^*$. Then the intersection cut derived from $\mathcal{C}^*$ dominates the intersection cut derived from $\mathcal{C}$, see Remark 3.2 of [31]. This makes deriving maximal $\mathcal{S}$-free sets desirable.

3 Extensions of submodular functions

In this section, we study continuous extensions of submodular functions. W.l.o.g., we assume in the sequel that, for any submodular function f, $f(\mathbf{0}) = 0$ holds (by a translation of a constant). It is known that the Lovász extension [62] extends f from $\mathcal{B}$ to $\bar{\mathcal{B}}$. Based on this extension, we construct another extension $\bar{\mathsf{F}}_f$ of f defined over the entire space $\mathbb{R}^n$, and study its analytical and combinatorial structures.

We first look at some polyhedra associated with the submodular function f [11,80]. Its *extended polymatroid* is defined as

$$\mathsf{EPM}_f := \{s \in \mathbb{R}^n : \forall x \in \mathcal{B},\ sx \leq f(x)\}, \tag{5}$$

and the convex hull of the Boolean-epigraph f over $\mathcal{B}$ is defined as

$$Q_f := \mathrm{conv}(\mathrm{epi}_{\mathcal{B}}(f)).$$

Recall that $\mathrm{ext}(\mathsf{EPM}_f)$ are the vertices of EPM_f. We further define the polyhedron

$$\mathsf{EE}_f := \{(x,t) \in \mathbb{R}^{n+1} : \forall s \in \mathrm{ext}(\mathsf{EPM}_f),\ sx \leq t\}. \tag{6}$$

In fact, EE_f contains Q_f, because of the following lemma:

Lemma 1 ([11]) $Q_f = \mathsf{EE}_f \cap (\bar{\mathcal{B}} \times \mathbb{R})$.

Therefore, $x \in \bar{\mathcal{B}}$ defines trivial facets of Q_f, and non-trivial facets of Q_f are $sx \leq t$, where s is a vertex of EPM_f.

These polyhedra in turn give rise to some functions associated with f. A convex function g is a *convex underestimating function* of f over $\mathcal{B}$, if for all $x \in \mathcal{B}$, $g(x) \leq f(x)$. The *convex envelope* F_f is defined as the maximal convex underestimating function of f over $\mathcal{B}$. Since Q_f is the epigraph of F_f, by Lemma 1,

$$\mathsf{F}_f : \bar{\mathcal{B}} \to \mathbb{R}, \quad x \mapsto \max_{s \in \mathrm{ext}(\mathsf{EPM}_f)} sx. \tag{7}$$

We remark that F_f is equivalent to the Lovász extension of f [11]. We will show that the cardinality $|\mathrm{ext}(\mathsf{EPM}_f)|$ is not polynomial in n. Thus, when computing F_f, it is inefficient to evaluate all sx for $s \in \mathrm{ext}(\mathsf{EPM}_f)$. However, the value and the (sub)-gradients of F_f at points in $\bar{\mathcal{B}}$ can be computed in a strongly polynomial time [11].

We define the envelope of f extended to $\mathbb{R}^n$ as

$$\bar{\mathsf{F}}_f : \mathbb{R}^n \to \mathbb{R}, \quad x \mapsto \max_{s \in \mathrm{ext}(\mathsf{EPM}_f)} sx. \tag{8}$$

We note that $\bar{\mathsf{F}}_f$ simply enlarges the domain of F_f from $\bar{\mathcal{B}}$ to $\mathbb{R}^n$. This extension is algebraically simple, but analytically less so. The analytical properties of $\bar{\mathsf{F}}_f$ outside $\bar{\mathcal{B}}$ will be studied in further detail. We find that EE_f is the epigraph of $\bar{\mathsf{F}}_f$, i.e., $\mathsf{EE}_f = \mathrm{epi}(\bar{\mathsf{F}}_f)$, so $\bar{\mathsf{F}}_f$ is a convex function. Since every facet $sx \leq t$ of EE_f is in one-to-one correspondence to a linear underestimator function sx of $\bar{\mathsf{F}}_f$, we call EE_f the *extended envelope epigraph*.

The problem of efficiently evaluating $\bar{\mathsf{F}}_f$ and its associated sub-gradients at a point in $\mathbb{R}^n$ is very important, because it is crucial in constructing intersection cuts. Regarding $\bar{\mathsf{F}}_f$, one can compute its value and sub-gradients at points in $\bar{\mathcal{B}}$ in a strongly polynomial time using a sorting algorithm [11]. As the Lovász extension F_f is a restriction of $\bar{\mathsf{F}}_f$ to the hypercube $\bar{\mathcal{B}}$, this fact implies that many properties of F_f may also hold for $\bar{\mathsf{F}}_f$. In the following, we will show how we

can reuse the sorting algorithm to compute $\bar{\mathsf{F}}_f$ over the entire space $\mathbb{R}^n$. This extension requires us to study the properties of $\bar{\mathsf{F}}_f$ and EE_f.

We first look at the combinatorial structure associated with the facets of EE_f. Recall that a permutation π on $[n]$ is a bijective map from $[n]$ to itself. The map $\pi(i) \in [n]$ is the image of an element $i \in [n]$ under this permutation. We denote by S_n the set of permutations on $[n]$. We define the following sets and vectors related to permutations.

Definition 2 Given $\pi \in S_n$ and $i \in \{0, \ldots, n\}$, we define $\pi([0]) := \varnothing$, $\pi([i]) := \{\pi(1), \ldots, \pi(i)\}$, and $v^i(\pi) := \mathsf{char}(\pi([i]))$. We also define the map $\sigma : S_n \to \mathbb{R}^n$ such that it satisfies $\sigma(\pi)_{\pi(i)} = f(v^i(\pi)) - f(v^{i-1}(\pi))$ for all $\pi \in S_n$ and for all $i \in [n]$.

The set of vertices $\mathrm{ext}(\mathsf{EPM}_f)$ is the image of S_n under the map σ.

Lemma 2 ([43]) $\sigma(S_n) = \mathrm{ext}(\mathsf{EPM}_f)$.

Every permutation $\pi \in S_n$ induces a vertex $\sigma(\pi)$ of $\mathrm{ext}(\mathsf{EPM}_f)$ through the map σ, so the cardinality of $\mathrm{ext}(\mathsf{EPM}_f)$ is $n!$ (not polynomial in n). The above lemma shows that every facet of EE_f (a non-trivial facet of Q_f) is given as $\sigma(\pi)x \le t$, and every linear underestimator of $\bar{\mathsf{F}}_f$ is given as $\sigma(\pi)x$.

Proposition 1 *Given a permutation $\pi \in S_n$, for all $i \in [n] \cup \{0\}$, the facet-defining inequality $\sigma(\pi)x \le t$ is supported by $\big(v^i(\pi), f(v^i(\pi))\big)$, i.e., $\sigma(\pi)v^i(\pi) = f(v^i(\pi))$.*

Proof We have:

$$\sigma(\pi)v^i(\pi) = \sum_{j\in[i]} \sigma(\pi)_{\pi(j)} = \sum_{j\in[i]} \Big(f(v^j(\pi)) - f(v^{j-1}(\pi))\Big) = f(v^i(\pi)) - f(0) = f(v^i(\pi)),$$

where the first equation follows from Defn. 2, the second from Lemma 2, and the last two from the expansion of the sum. □

Conversely to Prop. 1, given a point in the graph of f, we can construct all the facets supported by it.

Corollary 1 *For a point $v \in \mathcal{B}$, let ι be the number of ones in v. If a permutation $\pi \in S_n$ satisfies that $v = v^\iota(\pi)$, then $(v, f(v))$ supports the facet-defining inequality $\sigma(\pi)x \le t$ of EE_f.*

By Lemma 1, one may observe that a facet of EE_f includes and extends geometrically a facet of Q_f. This observation reveals the close relation between F_f in (7) and $\bar{\mathsf{F}}_f$ in (8). In order to separate facets of EE_f, we consider $\bar{\mathsf{F}}_f$ since EE_f is its epigraph.

From a convex analysis perspective, the nonsmooth polyhedral function $\bar{\mathsf{F}}_f$ is the maximum of a set of linear functions, so it is convex and positive homogeneous of degree 1. This means that $\bar{\mathsf{F}}_f$ is subdifferentiable [53]. Moreover, $\bar{\mathsf{F}}_f$ has the following analytical properties.

Proposition 2 *For all $x' \in \mathbb{R}^n$, $\partial\bar{\mathsf{F}}_f(x') = \mathrm{conv}(\mathrm{argmax}_{s\in\mathrm{ext}(\mathsf{EPM}_f)}\, sx')$. Moreover, for all $s \in \partial\bar{\mathsf{F}}_f(x')$, $\bar{\mathsf{F}}_f(x') = sx'$ and $\bar{\mathsf{F}}_f(x) \ge sx$ for any $x \in \mathbb{R}^n$.*

Proof As $\bar{\mathsf{F}}_f(x) = \max_{s\in\mathrm{ext}(\mathsf{EPM}_f)} sx$, $\bar{\mathsf{F}}_f$ is convex. By Lemma 3.1.13 of [71], it follows that $\partial\bar{\mathsf{F}}_f(x') = \mathrm{conv}(\mathrm{argmax}_{s\in\mathrm{ext}(\mathsf{EPM}_f)}\, sx')$. Since $s \in \partial\bar{\mathsf{F}}_f(x')$, this implies that $s = \sum_i \lambda_i s^i$ with $s^i \in \mathrm{ext}(\mathsf{EPM}_f)$ and $\sum_i \lambda_i = 1$. Thus, $sx' = \sum_i \lambda_i s^i x' = \sum_i \lambda_i \bar{\mathsf{F}}_f(x') = \bar{\mathsf{F}}_f(x')$. By convexity of $\bar{\mathsf{F}}_f$, we have $\bar{\mathsf{F}}_f(x) \ge \bar{\mathsf{F}}_f(x') + s(x - x') = sx' + s(x - x') = sx$. □

Given $\tilde{x} \in \mathbb{R}^n$, the evaluation of $\bar{\mathsf{F}}_f(\tilde{x})$ is called the *extended polymatroid vertex maximization problem*, as by definition $\bar{\mathsf{F}}_f(\tilde{x})$ equals

$$\max_{s\in\mathrm{ext}(\mathsf{EPM}_f)} s\tilde{x}. \tag{9}$$

By Prop. 2, any optimal solution s^* of Eq. (9) is a subgradient of $\bar{\mathsf{F}}_f$ at $\tilde{x}$, i.e., $s^* \in \partial\bar{\mathsf{F}}_f(\tilde{x})$. By Lemma 2, $\max_{s\in\mathrm{ext}(\mathsf{EPM}_f)} s\tilde{x} = \max_{\pi\in S_n} \sigma(\pi)\tilde{x}$, so (9) asks for a permutation π^* that maximizes $\sigma(\pi^*)\tilde{x}$. One of the main results of this section is an algorithm to solve (9).

To tackle (9), we look at a related relaxed problem, namely the *extended polymatroid maximization problem*:

$$\max_{s\in \mathsf{EPM}_f} s\tilde{x}. \tag{10}$$

If $\tilde{x} \geq 0$, a strongly polynomial time *sorting algorithm* (a greedy algorithm, in fact) can be used to solve the extended polymatroid maximization [43]. This algorithm first sorts the entries of $\tilde{x}$ in an non-increasing order, then it finds a permutation $\pi^* \in S_n$ such that $\tilde{x}_{\pi^*(1)} \geq \cdots \geq \tilde{x}_{\pi^*(n)} \geq 0$; finally, it finds an optimal solution to (10) being the vector $\sigma(\pi^*) = (f(v^1(\pi^*)) - f(v^0(\pi^*)), \ldots, f(v^n(\pi^*)) - f(v^{n-1}(\pi^*)))$. See [43] for more details.

We note that the vertices $\text{ext}(\mathsf{EPM}_f)$ are a finite set, so (9) is always bounded. Moreover, by the Minkowski-Weyl theorem [29], EPM_f is the Minkowski sum of the polytope $\text{conv}(\text{ext}(\mathsf{EPM}_f))$ and the recession cone of EPM_f. By Proposition 3.15 of [29] and (5), the recession cone admits the form $\{s \in \mathbb{R}^n : \forall x \in \mathcal{B}, sx \leq 0\}$, so the cone is non-empty, and EPM_f is unbounded. This means that (10) may be unbounded.

Lemma 3 ([11,43]) *When $\tilde{x} \geq 0$, the optimum of (10) is a vertex of EPM_f (i.e., in $\text{ext}(\mathsf{EPM}_f)$), and (9) is equivalent to (10); when $\tilde{x}$ has some negative entries, (9) is unbounded, and therefore not equivalent to (10).*

Even if (9) is not equivalent to (10) in general, we show that (9) can still be solved by the sorting algorithm.

Proposition 3 *The output of the sorting algorithm is an optimal solution of the extended polymatroid vertex maximization problem (9).*

Proof Let π^* be the permutation found by the sorting algorithm. By Lemma 2, $\sigma(\pi^*)$ is in $\text{ext}(\mathsf{EPM}_f)$ and hence a feasible solution to (9). Next, we prove the optimality of $\sigma(\pi^*)$. Let the scalar $d := \min_{i\in[n]} \tilde{x}_i$. We can write $\tilde{x}$ as the sum of $(\tilde{x} - d\mathbf{1})$ and $d\mathbf{1}$, where the translated vector $\tilde{x} - d\mathbf{1} = (\tilde{x}_i - d)_{i\in[n]}$ has non-negative components. We find that the following inequalities hold:

$$\begin{aligned} \sigma(\pi^*)\tilde{x} &\leq \max_{s\in \text{ext}(\mathsf{EPM}_f)} s\tilde{x} = \max_{s\in \text{ext}(\mathsf{EPM}_f)} s(\tilde{x} - d\mathbf{1} + d\mathbf{1}) \\ &\leq \max_{s\in \text{ext}(\mathsf{EPM}_f)} s(\tilde{x} - d\mathbf{1}) + \max_{s\in \text{ext}(\mathsf{EPM}_f)} s(d\mathbf{1}), \end{aligned} \tag{11}$$

where the first inequality follows from the fact given by Lemma 2 that $\sigma(\pi^*)$ is in $\text{ext}(\mathsf{EPM}_f)$, the last inequality follows from the fact that maximum of the sum is at most the sum of maxima. We next construct the optimal solutions to $\max_{s\in \text{ext}(\mathsf{EPM}_f)} s(\tilde{x} - d\mathbf{1})$ and $\max_{s\in \text{ext}(\mathsf{EPM}_f)} s(d\mathbf{1})$, respectively. First, since the permutation π^* maps $\tilde{x}$ into a vector with non-increasing entries and the entries of $d\mathbf{1}$ are identical, we have that $(\tilde{x} - d\mathbf{1})_{\pi^*(1)} \geq \cdots \geq (\tilde{x} - d\mathbf{1})_{\pi^*(n)}$. Since $\tilde{x} - d\mathbf{1}$ is constructed to be non-negative, by Lemma 3, $\max_{s\in \mathsf{EPM}_f} s(\tilde{x} - d\mathbf{1}) = \max_{s\in \text{ext}(\mathsf{EPM}_f)} s(\tilde{x} - d\mathbf{1})$. Moreover, since π^* is also the permutation that sorts $(\tilde{x} - d\mathbf{1})$ in a non-increasing order, it follows again from Lemma 3 that $\sigma(\pi^*)$ is an optimal solution to $\max_{s\in \mathsf{EPM}_f} s(\tilde{x} - d\mathbf{1})$. This implies that $\sigma(\pi^*)$ is also an optimal solution to $\max_{s\in \text{ext}(\mathsf{EPM}_f)} s(\tilde{x} - d\mathbf{1})$. Secondly, for any $\pi \in S_n$, it follows from Defn. 2 that $v^n(\pi) = \mathbf{1}$. This implies that $\sigma(\pi)v^n(\pi) = \sigma(\pi)\mathbf{1} = f(\mathbf{1})$, where the last equation follows from Prop. 1. In addition, by Lemma 2, $\text{argmax}_{s\in \text{ext}(\mathsf{EPM}_f)} s(d\mathbf{1}) = \sigma(\text{argmax}_{\pi\in S_n} \sigma(\pi)(d\mathbf{1}))$. Since all $\sigma(\pi)(d\mathbf{1})$ are identically equal to $df(\mathbf{1})$, we can pick $\sigma(\pi^*)$ as the optimal solution to $\max_{s\in \text{ext}(\mathsf{EPM}_f)} s(d\mathbf{1})$. Finally, we find that $\max_{s\in \text{ext}(\mathsf{EPM}_f)} s(\tilde{x} - d\mathbf{1})$ and $\max_{s\in \text{ext}(\mathsf{EPM}_f)} s(d\mathbf{1})$ have a common optimal solution $\sigma(\pi^*)$. This implies that the inequalities in (11) become equations, because

$$\sigma(\pi^*)\tilde{x} \leq \max_{s\in \text{ext}(\mathsf{EPM}_f)} s\tilde{x} \leq \sigma(\pi^*)(\tilde{x} - d\mathbf{1}) + \sigma(\pi^*)(d\mathbf{1}) = \sigma(\pi^*)\tilde{x}.$$

Therefore, $\sigma(\pi^*)$ is an optimal solution to $\max_{s\in \text{ext}(\mathsf{EPM}_f)} s\tilde{x}$.

□

Given $\tilde{x} \in \mathbb{R}^n$, the sorting algorithm outputs a permutation acting on the entries of $\tilde{x}$. The sorting algorithm is also translation-invariant, i.e., translating each entry of $\tilde{x}$ by the same value does not change the output permutation. A by-product of Prop. 3 is that $\bar{\mathsf{F}}_f$ is linear over specific lines specified as follows.

Corollary 2 *Let $\tilde{x} \in \mathbb{R}^n$, then $\bar{\mathsf{F}}_f$ is linear on $\tilde{x} + \lambda \mathbf{1}$ w.r.t. $\lambda \in \mathbb{R}$.*

We look at the boundary of EE_f. By Prop. 1 and Cor. 1, for all $x \in \mathcal{B}$, the point $(x, f(x))$ supports some facets of EE_f.

Theorem 1 $\mathsf{EE}_f \cap \text{hypo}_{\mathcal{B}}(f) = \text{graph}_{\mathcal{B}}(f) \subseteq \text{bd}(\mathsf{EE}_f)$.

Proof We consider a point $v \in \mathcal{B}$ and look at the line $\ell = \{(v,t) : t \in \mathbb{R}\}$. It can be separated into the restricted epigraph $\ell_+ := \{(v,t) : f(v) \leq t\}$ and the restricted hypograph $\ell_- := \{(v,t) : f(v) \geq t\}$, as $\ell_+ \cap \ell_- = (v, f(v))$ and $\ell = \ell_+ \cup \ell_-$. First, we know that, by definition of Q_f and Lemma 1, $\ell_+ \subseteq Q_f \subseteq \mathsf{EE}_f$. Second, by Prop. 1, the point $(v, f(v))$ supports some facets of EE_f, so the point (v,t) with $t < f(v)$ is separated by these facets from EE_f. Thereby, we know that $\ell_- \cap \mathsf{EE}_f = \{(v, f(v))\}$. To summarize, we know that $\mathsf{EE}_f \cap \ell = \ell_+$ and $(v, f(v)) \in \text{bd}(\mathsf{EE}_f)$. As $\text{graph}_{\mathcal{B}}(f) = \bigcup_{v \in \mathcal{B}} \{(v, f(v))\}$, we have that $\text{graph}_{\mathcal{B}}(f) \subseteq \text{bd}(\mathsf{EE}_f)$. As the hypograph $\text{hypo}_{\mathcal{B}}(f) = \bigcup_{v \in \mathcal{B}} \{(v,t) : f(v) \geq t\}$ (union of restricted hypographs), we have that $\mathsf{EE}_f \cap \text{hypo}_{\mathcal{B}}(f) = \text{graph}_{\mathcal{B}}(f)$. □
□

As mentioned above, $\bar{\mathsf{F}}_f$ is convex and $\mathsf{EE}_f = \text{epi}(\bar{\mathsf{F}}_f)$, so $\bar{\mathsf{F}}_f$ is also a continuous extension of f. As EE_f contains Q_f, $\bar{\mathsf{F}}_f$ further extends F_f (the Lovász extension).

We now understand enough of the facial structure of EE_f. By Prop. 2, we can also compute the value and subgradients of $\bar{\mathsf{F}}_f$ at any point in $\mathbb{R}^n$, which are used in the construction of intersection cuts.

4 $\mathcal{S}$-free sets for submodular functions

We consider two types of $\mathcal{S}$-free sets for the Boolean-hypograph of a given submodular function f.

First, we show that one can lift a maximal $\mathcal{B}$-free set into a maximal $\text{hypo}_{\mathcal{B}}(f)$-free set.

Theorem 2 *Let $f : \mathcal{B} \to \mathbb{R}$ be an arbitrary function, and let $\mathcal{K}$ be a maximal $\mathcal{B}$-free set in $\mathbb{R}^n$. Then $\mathcal{C} := \mathcal{K} \times \mathbb{R}$ is a maximal $\text{hypo}_{\mathcal{B}}(f)$-free set.*

Proof We note that $\text{int}(\mathcal{C}) = \text{int}(\mathcal{K}) \times \mathbb{R}$. Since $\text{int}(\mathcal{C}) \cap \text{hypo}_{\mathcal{B}}(f) = \varnothing$, $\mathcal{C}$ is $\text{hypo}_{\mathcal{B}}(f)$-free. Assume that there exists a $\text{hypo}_{\mathcal{B}}(f)$-free set $\mathcal{C}'$ containing $\mathcal{C}$. Then the recession cone of $\mathcal{C}'$ must contain that of $\mathcal{C}$, so $\mathcal{C}' = \mathcal{K}' \times \mathbb{R}$ for some closed convex set $\mathcal{K}'$ containing $\mathcal{K}$. Moreover, $\mathcal{K}'$ must be a $\mathcal{B}$-free set, otherwise, there exists a point $x \in \mathcal{B} \cap \text{int}(\mathcal{K}')$ such that $(x, f(x)) \in \text{int}(\mathcal{K}') \times \mathbb{R} = \text{int}(\mathcal{C}')$. However, since $\mathcal{K}$ is maximally $\mathcal{B}$-free, this implies that $\mathcal{K} = \mathcal{K}'$. As a result, $\mathcal{C} = \mathcal{C}'$, so $\mathcal{C}$ is maximal. □

This construction does not rely on any structure of f, as it just lifts a $\mathcal{B}$-free set. For any $j \in [n]$, the simple lifted split $\{x \in \mathbb{R}^n : 0 \leq x_j \leq 1\} \times \mathbb{R}$ is a maximal $\text{hypo}_{\mathcal{B}}(f)$-free set. We next construct $\text{hypo}_{\mathcal{B}}(f)$-free sets using submodularity, for both theoretical and computational interests. We show that both the extended epigraph EE_f and its strict subset Q_f are $\text{hypo}_{\mathcal{B}}(f)$-free sets.

Proposition 4 EE_f, Q_f *are* $\text{hypo}_{\mathcal{B}}(f)$*-free sets.*

Proof Since $\text{graph}_{\mathcal{B}}(f) \subseteq \text{bd}(\mathsf{EE}_f)$, we conclude that $\mathsf{EE}_f \cap \text{hypo}_{\mathcal{B}}(f) \subseteq \text{bd}(\mathsf{EE}_f)$ and hence $\text{int}(\mathsf{EE}_f) \cap \text{hypo}_{\mathcal{B}}(f) = \varnothing$. Additionally, EE_f is convex and hence $\text{hypo}_{\mathcal{B}}(f)$-free. As $Q_f \subseteq \mathsf{EE}_f$, Q_f is $\text{hypo}_{\mathcal{B}}(f)$-free set. □

It is known that the maximal $\mathcal{S}$-free set for $\mathcal{S}$ as the hypograph of a convex function is its epigraph. We shall show, however, that the extended epigraph EE_f of a submodular function f is not a maximal $\text{hypo}_{\mathcal{B}}(f)$-free set. At a high-level, a possible way to test the maximality of EE_f is as follows. The set Q_f is the convex hull of $\text{epi}_{\mathcal{B}}(f)$. Geometrically, Q_f is the "minimal" convex set containing $\text{epi}_{\mathcal{B}}(f)$. Intuitively, it is unlikely that a "minimal" set turns out to be a good "maximal" $\text{hypo}_{\mathcal{B}}(f)$-free set. We therefore remove some facets from Q_f in order to enlarge this polyhedron. After removing trivial facets of Q_f, the enlarged polyhedron is the extended epigraph EE_f of the envelope of f. However, this enlargement is still not sufficient. We therefore look at a further enlargement of EE_f.

The following fundamental theorem gives a sufficient and necessary condition on (maximal) $\text{hypo}_{\mathcal{B}}(f)$-free sets containing EE_f.

Theorem 3 *Let $\mathcal{C}$ be a full-dimensional closed convex set in $\mathbb{R}^{n+1}$ containing EE_f. Then $\mathcal{C}$ is a $\text{hypo}_{\mathcal{B}}(f)$-free set if and only if $\mathcal{C}$ is $\text{graph}_{\mathcal{B}}(f)$-free. Moreover, $\mathcal{C}$ is a maximal $\text{hypo}_{\mathcal{B}}(f)$-free set if and only if $\mathcal{C}$ is a polyhedron and there is at least one point of $\text{graph}_{\mathcal{B}}(f)$ in the relative interior of each facet of $\mathcal{C}$.*

Proof We note that by Thm. 1, $\text{graph}_{\mathcal{B}}(f) \subseteq \text{bd}(\mathsf{EE}_f) \subseteq \mathsf{EE}_f \subseteq \mathcal{C}$. Thereby, $\text{graph}_{\mathcal{B}}(f) \cap \text{int}(\mathcal{C}) = \varnothing$ (i.e., $\mathcal{C}$ is $\text{graph}_{\mathcal{B}}(f)$-free) if and only if $\text{graph}_{\mathcal{B}}(f) \subseteq \text{bd}(\mathcal{C})$.

We consider the $\text{hypo}_{\mathcal{B}}(f)$-freeness first. We prove the forward direction. Assume that $\mathcal{C}$ is a $\text{hypo}_{\mathcal{B}}(f)$-free set. Suppose, to aim at a contradiction, that there exists a point $(v, f(v)) \in \text{int}(\mathcal{C}) \cap \text{graph}_{\mathcal{B}}(f)$. Then there exists a sufficiently small $\epsilon > 0$ such that $(v, f(v) - \epsilon) \in \text{int}(\mathcal{C})$, but $(v, f(v) - \epsilon) \in \text{hypo}_{\mathcal{B}}(f)$, which leads to a contradiction. We prove the reverse direction. Assume that $\mathcal{C}$ is $\text{graph}_{\mathcal{B}}(f)$-free. Suppose, to aim at a contradiction, that there exists a point $(v, f(v)-\delta) \in \text{int}(\mathcal{C})$ with $v \in \mathcal{B}$ and $\delta > 0$. As, for some $\epsilon > 0$, $(v, f(v)+\epsilon) \subseteq \text{int}(\mathsf{EE}_f) \subseteq \text{int}(\mathcal{C})$, by convexity of $\mathcal{C}$, $(v, f(v)) \in \text{int}(\mathcal{C})$, which leads to a contradiction. This implies that $\mathcal{C}$ is $\text{hypo}_{\mathcal{B}}(f)$-free if and only if $\text{graph}_{\mathcal{B}}(f)$-free (or $\text{graph}_{\mathcal{B}}(f) \subseteq \text{bd}(\mathcal{C})$).

We consider maximality next. The proof is similar to that [14] for the bounded maximal lattice-free set. Let $\mathcal{C}$ be a maximal $\text{graph}_{\mathcal{B}}(f)$-free set. For each $v \in \text{graph}_{\mathcal{B}}(f)$, it follows from the separating hyperplane theorem that there exists a half-space $\{z : a^v z \leq b_v\}$ containing $\mathcal{C}$ such that $a^v v = b_v$. As $\text{graph}_{\mathcal{B}}(f)$ is a finite set, the set $P := \{z : \forall v \in \text{graph}_{\mathcal{B}}(f), a^v z \leq b_v\}$ is a polyhedron. By construction, P is $\text{graph}_{\mathcal{B}}(f)$-free and $\mathcal{C} \subseteq P$, thus $\mathcal{C} = P$ by maximality of $\mathcal{C}$.

We now show that there is at least one point of $\text{graph}_{\mathcal{B}}(f)$ in the relative interior of each facet of $\mathcal{C}$. Assume $\mathcal{C} = \{z : \forall i \in M, a^i z \leq b_i\}$, where $a^i z \leq b_i, i \in M$, are all distinct facet-defining inequalities for $\mathcal{C}$. Suppose, to aim at a contradiction, that the facet $F_t = \{z \in \mathcal{C} : a^t z = b_t\}$ does not contain any point of $\text{graph}_{\mathcal{B}}(f)$ in its relative interior. Let $\epsilon > 0$, enlarge $\mathcal{C}$ to another polyhedron $\mathcal{C}' := \{z : \forall i \in M \setminus \{t\}, a^i z \leq b_i, a^t z \leq b_t + \epsilon\}$. As $\mathcal{C} \subsetneq \mathcal{C}'$ and $\mathcal{C}$ is maximally $\text{graph}_{\mathcal{B}}(f)$-free, $\mathcal{C}'$ contains points of $\text{graph}_{\mathcal{B}}(f)$ in its interior. Thus, the point $z' := \text{argmin}_{z \in \text{int}(\mathcal{C}') \cap \text{graph}_{\mathcal{B}}(f)} a^t z$ exists. It follows from $z' \in \text{int}(\mathcal{C}')$ that $\forall i \in M \setminus \{t\}$, $a^i z' < b_i$, and $a^t z' < b_t + \epsilon$. Since $F_t = \{z : \forall i \in M \setminus \{t\}, a^i z \leq b_i, a^t z = b_t\}$, $a^t z'$ cannot equal b_t, otherwise, this implies that F_t contains $z' \in \text{graph}_{\mathcal{B}}(f)$ in its relative interior. Since $\mathcal{C}$ is $\text{graph}_{\mathcal{B}}(f)$-free, $a^t z'$ cannot be strictly less than b_t, otherwise, this implies that $\mathcal{C}$ contains $z' \in \text{graph}_{\mathcal{B}}(f)$ in its interior. Then, it must be that $a^t z' > b_t$, and $\mathcal{C}^* := \{\forall i \in M \setminus \{t\}, a^i z \leq b_i, a^t z \leq a^t z'\}$ strictly includes $\mathcal{C}$. By construction, $\mathcal{C}^*$ does not contain any point of $\text{graph}_{\mathcal{B}}(f)$ in its interior. This contradicts the maximality of $\mathcal{C}$. □

□

The above theorem is purely geometrical. Since submodular functions are combinatorial objects, we translate this theorem to a combinatorial language. We first define a combinatorial object in the Boolean hypercube $\mathcal{B}$.

Definition 3 Let $x^0, x^1, \ldots, x^n$ be $n+1$ distinct points of $\mathcal{B}$. They are called *monotone*, if $\mathbf{0} = x^0 < x^1 < \cdots < x^n = \mathbf{1}$. We call the corresponding ordered set $(x^0, \ldots, x^n) \subseteq \mathcal{B}$ a *monotone chain* in $\mathcal{B}$.

Therefore, we use a monotone chain to represent a set of monotone points. Recall that $v^i(\pi) = \mathsf{char}(\pi([i]))$ defined in Defn. 2 is the characteristic vector of the image of the set $[i] = \{1, \ldots, i\}$ under the permutation π. We have the following observation.

Proposition 5 *The set S_n of permutations is in one-to-one correspondence to the set of monotone chains via the map V defined as follows: for all $\pi \in S_n$, $V(\pi) := (v^i(\pi) \mid i \in \mathcal{N} \cup \{0\})$.*

Proof It suffices to prove that, under the map, each permutation is mapped to a monotone chain, and for each monotone chain, there exists a permutation mapped to it. By Prop. 1, since $\varnothing = \pi(\varnothing) = \pi([0]) \subsetneq \cdots \subsetneq \pi([n]) = [n]$, by Defn. 2, $\mathbf{0} = v^0(\pi) < \cdots < v^n(\pi) = \mathbf{1}$, so $V(\pi)$ is a monotone chain. Conversely, given a monotone chain $(x^0, \ldots, x^n)$, we construct π as follows: by default, $\pi(0) = 0$; and for all $i \in [n]$, $\pi(i)$ is the index of the unique non-zero entry of $x^i - x^{i-1}$. It follows that $V(\pi)$ is the chain. □

We find that permutations and monotone chains are indeed equivalent. We note that any $n+1$ distinct points from $\text{graph}_{\mathcal{B}}(f)$ are affinely independent in $\mathbb{R}^{n+1}$ and hence support a hyperplane in $\mathbb{R}^{n+1}$. Thereby, we can infer from Prop. 1 and Prop. 5 that

Corollary 3 *If* $(x^0, \ldots, x^n)$ *is a monotone chain in* $\mathcal{B}$*, then distinct points* $(x^0, f(x^0)), \ldots, (x^n, f(x^n))$ *of* $\text{graph}_{\mathcal{B}}(f)$ *define (or support) a facet of the extended envelope epigraph* EE_f*.*

We say that this monotone chain *induces the facet*. In fact, we find that facets of EE_f, permutations on $[n]$, and monotone chains in $\mathcal{B}$ are in one-to-one correspondence. Therefore, we can view them as the same objects. In particular, Prop. 5 relates permutations and monotone chains. We give the following characterization of permutations on $[n]$.

Definition 4 A subset S'_n of permutations of S_n is called a *cover*, if $\bigcup_{\pi \in S'_n} V(\pi) = \mathcal{B}$; moreover, S'_n is called a *minimal cover* if, additionally, for all $\pi \in S'_n$, $V(\pi) \setminus \bigcup_{\pi' \in S'_n : \pi' \neq \pi} V(\pi')$ is not empty.

We want to enlarge EE_f by removing its facets. This is equivalent to removing permutations from S_n. Let S'_n be a subset of permutations of S_n, and $\mathcal{C}(S'_n) := \{(x,t) : \forall \pi \in S'_n, \sigma(\pi)x \leq t\}$ denotes the relaxation of the extended envelope epigraph induced by S'_n. It is obvious that $\mathsf{EE}_f = \mathcal{C}(S_n) \subseteq \mathcal{C}(S'_n)$ for any $S'_n \subseteq S_n$. The following corollary translates Thm. 3 to a combinatorial language.

Corollary 4 *Let* S'_n *be a subset of permutations of* S_n*.* $\mathcal{C}(S'_n)$ *is* $\text{hypo}_{\mathcal{B}}(f)$*-free if and only if* S'_n *is a cover.* $\mathcal{C}(S'_n)$ *is maximally* $\text{hypo}_{\mathcal{B}}(f)$*-free if and only if* S'_n *is a minimal cover.*

Proof First, we note that $C(S'_n)$, as a relaxation of EE_f contains $\text{graph}_{\mathcal{B}}(f)$. Next, we assume that S'_n is a cover. Then points of $\text{graph}_{\mathcal{B}}(f)$ support facets of $C(S'_n)$. By Thm. 3, $C(S'_n)$ is $\text{hypo}_{\mathcal{B}}(f)$-free if and only if it is a cover. Finally, S'_n is a minimal cover, if and only if then each facet of $C(S'_n)$ has a point of $\text{graph}_{\mathcal{B}}(f)$ in its interior. By Thm. 3, the later is equivalent to that $C(S'_n)$ is maximally $\text{hypo}_{\mathcal{B}}(f)$-free. □

We can now disprove the maximality EE_f by means of a counter-example. Thanks to Cor. 4, we can use a counting argument to show that we can remove facets from EE_f. This results in a new enlarged $\text{hypo}_{\mathcal{B}}(f)$-free polyhedron.

Proposition 6 EE_f *is not maximally* $\text{hypo}_{\mathcal{B}}(f)$*-free.*

Proof It suffices to find a counter-example. Consider $n = 3$, $\mathcal{B} = \{0,1\}^3$, there are 6 permutations, and 6 monotone chains (see Fig. 1). We assume that, in a non-degenerate case, the associated extended envelope epigraph EE_f has 6 facets induced by 6 chains respectively. The vertices $(0,0,0)$ and $(1,1,1)$ are visited by all the chains, while the other vertices are visited twice each. Therefore, a chain cannot "exclusively" visit a vertex, so the corresponding facet cannot contain one point of $\text{graph}_{\mathcal{B}}(f)$ in its relative interior. In fact, we can remove some facets from the extended envelope epigraph. We keep three chains:

$$
\begin{aligned}
&(\,(0,0,0),(0,0,1),(0,1,1),(1,1,1)\,)\,,\\
&(\,(0,0,0),(0,1,0),(1,1,0),(1,1,1)\,)\,,\\
&(\,(0,0,0),(1,0,0),(1,0,1),(1,1,1)\,)\,.
\end{aligned}
$$

These chains induce 3 facets such that at least one point of $\text{graph}_{\mathcal{B}}(f)$ is in the relative interior of each facet and each point of $\mathcal{B}$ is in these 3 facets, so the polyhedron defined by these 3 facets is a $\text{hypo}_{\mathcal{B}}(f)$-free set larger than EE_f. □

We discuss why enlarging EE_f is not a trivial feat. We build a bipartite graph $G := (\mathcal{B} \cup S_n, E)$. An edge e of E connects a vertex $v \in \mathcal{B}$ to a permutation $\pi \in S_n$ if $v \in V(\pi)$. Then, a minimal cover is a subset S'_n of S_n such that: i) each vertex of $\mathcal{B}$ is incident to at least one permutation in S'_n; ii) each permutation in S'_n is incident to a vertex of $\mathcal{B}$ that no other permutation in S'_n is incident to. As $|\mathcal{B}| = 2^n$ and $|S_n| = n!$, the size of such a graph is not polynomial in n. We need additional information in order to enlarge EE_f efficiently.

We relax the submodular maximization problem (1) through a polyhedral outer approximation $\mathcal{P}$ of $\text{hypo}_{\mathcal{B}}(f)$. Let X be the orthogonal projection of $\mathcal{P}$ on x-space. We remark that, within a branch-and-cut algorithm, X might be within a low-dimensional face of $\bar{\mathcal{B}}$. Let $\tilde{z} := (\tilde{x}, \tilde{t})$ be an optimal basic feasible solution to the LP relaxation $\max_{(x,t) \in \mathcal{P}} t$, which corresponds to a vertex of $\mathcal{P}$. We assume that $\tilde{x} \notin \mathcal{B}$, otherwise, $\tilde{x}$ is already an optimal solution to (1).

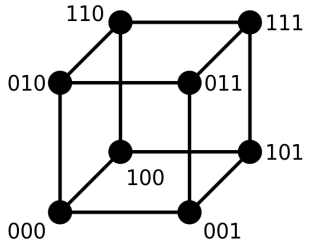

Fig. 1: a Boolean cube $\mathcal{B} = \{0,1\}^3$ © [44]

As $\tilde{z}$ is the point that we want to separate from $\text{hypo}_{\mathcal{B}}(f)$, we follow the method presented in Sect. 2 to construct an intersection cut. According to [27,50], we can use a feasible basis of the LP relaxation to create a simplicial cone $\mathcal{R}$. This cone $\mathcal{R}$ can be easily obtained from the simplex tableau associated with the chosen basis. In our case, we select the optimal basis defining $\tilde{z}$ so that $\tilde{z}$ is the apex of the corresponding cone $\mathcal{R}$. Moreover, we use EE_f as $\text{hypo}_{\mathcal{B}}(f)$-free set. To determine whether the linear inequality (4) separates $\tilde{z}$ from $\text{hypo}_{\mathcal{B}}(f)$, we need to verify whether $\tilde{z} \in \text{int}(\mathsf{EE}_f)$.

The polyhedral outer approximation $\mathcal{P}$ gives rise to a piece-wise linear concave overestimating function of f over X: $\bar{f}(x) := \max_{(x,t)\in\mathcal{P}} t$, such that $\max_{(x,t)\in\mathcal{P}} t = \max_{x\in X} \bar{f}(x)$. This implies that $\tilde{t} = \bar{f}(\tilde{x})$. We then have the following observation.

Proposition 7 *Assume that f is not affine over X. If $\tilde{x} \in \text{relint}(X)$, then $\bar{f}(\tilde{x}) > \bar{\mathsf{F}}_f(\tilde{x})$, i.e., $(\tilde{x}, \bar{f}(\tilde{x})) \in \text{int}(\mathsf{EE}_f)$.*

Proof As $\bar{f}$ is concave overestimator of f over X and $\bar{\mathsf{F}}_f$ is convex underestimator of f over X, $\bar{f} \geq \bar{\mathsf{F}}_f$ over X. Suppose, to aim at a contradiction, that $\bar{f}(\tilde{x}) = \bar{\mathsf{F}}_f(\tilde{x})$. Define a concave function $g := \bar{f} - \bar{\mathsf{F}}_f$, then for all $x \in X$, $g(x) \geq 0$, and $g(\tilde{x}) = 0$. By its concavity, there exists an affine overestimating function a of g, such that $g(\tilde{x}) = a(\tilde{x}) = 0$, and for all $x \in X$, $0 \leq g(x) \leq a(x)$. As $\tilde{x} \in \text{relint}(X)$, the affinity of a implies that $a = g = 0$ over X, i.e., $\bar{f} = \bar{\mathsf{F}}_f$ over X. So f is concave and convex over X and thus affine over X, which is a contradiction. □

The measure of the relative boundary $\text{relbd}(X)$ is zero, so we can assume that a mild *relative interior condition* that $\tilde{x} \in \text{relint}(X)$ holds with probability one. Under this assumption, the relaxation point $\tilde{z} = (\tilde{x}, \bar{f}(\tilde{x}))$ is in the relative interior of the extended envelope epigraph with probability one. In Sect. 8, we will empirically evaluate the effectiveness of these intersection cuts for various submodular maximization problems.

Throughout the rest of this paper, we will encounter multiple nonconvex optimization problems. W.l.o.g., we will consider the simplicial cone defined by the simplex tableau associated with an optimal feasible basis of their LP relaxations. Such simplicial cones are commonly employed in computational implementations. For the sake of brevity, we refer to such cones as *optimal tableau cones*.

5 Extensions to SS functions

This section considers Boolean-hypograph and Boolean-superlevel sets for an SS function $f := f_1 - f_2$, where f_1 and f_2 are two submodular functions. We extend our previous results on the Boolean-hypograph set of submodular functions; thus one can generate intersection cuts for a larger family of discrete nonconvex sets.

More specifically, we consider the following nonconvex set

$$\mathcal{S} := \{(x,t) \in \mathcal{B} \times \mathbb{R} : f(x) \geq \ell t\}, \tag{12}$$

with $\ell \in \{0,1\}$. Given a relaxation point $(\tilde{x}, \tilde{t}) \notin \mathcal{S}$, we want to find cutting planes separating this point from $\mathcal{S}$.

Let $\bar{\mathsf{F}}_{f_1} := \max_{s\in\mathsf{EPM}_{f_1}} sx$ and $\bar{\mathsf{F}}_{f_2} := \max_{s\in\mathsf{EPM}_{f_2}} sx$ be extended envelopes of f_1, f_2, respectively. As $\bar{\mathsf{F}}_{f_1}$ (resp. $\bar{\mathsf{F}}_{f_2}$) is a convex extension of f_1 (resp. f_2), we have that $\mathcal{S} = \{(x,t) \in \mathcal{B} \times \mathbb{R} :$

$\bar{\mathsf{F}}_{f_1}(x) - \bar{\mathsf{F}}_{f_2}(x) \geq \ell t\}$. By relaxing $\mathcal{B}$ to $\mathbb{R}^n$, a (nonconvex) continuous outer approximation of $\mathcal{S}$ is

$$\bar{\mathcal{S}} := \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : \bar{\mathsf{F}}_{f_1}(x) - \bar{\mathsf{F}}_{f_2}(x) \geq \ell t\}. \tag{13}$$

Moreover, for all $x \in \mathcal{B}$, $(x,t) \in \bar{\mathcal{S}}$ if and only if $(x,t) \in \mathcal{S}$.

Special cases. When $\ell = 1$, $\mathcal{S}$ is the Boolean-hypograph of the SS function f; when $\ell = 0$, $\mathcal{S}$ is the 0-superlevel set of the SS function f over the Boolean hypercube. Setting $f_2 = 0$ and $\ell = 1$, the set $\mathcal{S}$ becomes the Boolean-hypograph $\{(x,t) \in \mathcal{B} \times \mathbb{R} : f_1(x) \geq t\}$, which is studied in the previous section. Setting $f_1 = 0$, the relaxed set $\bar{\mathcal{S}}$ becomes $\{(x,t) \in \mathcal{B} \times \mathbb{R} : \bar{\mathsf{F}}_{f_2}(x) \leq -\ell t\}$. Let $(\tilde{x}, \tilde{t}) \notin \bar{\mathcal{S}}$. Since, by Prop. 2, $\bar{\mathsf{F}}_{f_2}(x) \geq \gamma^* x$ and $\bar{\mathsf{F}}_{f_2}(\tilde{x}) = \gamma^* \tilde{x}$ for any $\gamma^* \in \partial \bar{\mathsf{F}}_{f_2}(\tilde{x})$, then the simple outer approximation cut $\gamma^* x \leq -\ell t$ is a valid inequality for $\bar{\mathcal{S}}$ (hence for $\mathcal{S}$).

In general, we should separate intersection cuts specifically for SS functions. Let $\gamma^* \in \partial \bar{\mathsf{F}}_{f_2}(\tilde{x})$ be a solution to (9) associated with f_2, and we define the set

$$\mathcal{C}_{\tilde{x}} := \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : \bar{\mathsf{F}}_{f_1}(x) - \gamma^* x \leq \ell t\}. \tag{14}$$

The following proposition characterizes the $\mathcal{S}$-free set $\mathcal{C}_{\tilde{x}}$ for Eq. (13).

Proposition 8 *The set $\mathcal{C}_{\tilde{x}}$ in (14) is an $\mathcal{S}$-free set. Moreover, if $(\tilde{x}, \tilde{t}) \notin \bar{\mathcal{S}}$, then $\mathcal{C}_{\tilde{x}}$ does not contain $\tilde{x}$ in its interior.*

Proof We first prove that $\mathcal{C}_{\tilde{x}}$ is $\bar{\mathcal{S}}$-free. By definition, $\gamma^* x \leq \bar{\mathsf{F}}_{f_2}(x)$, which implies that $\bar{\mathsf{F}}_{f_1}(x) - \gamma^* x \geq \bar{\mathsf{F}}_{f_1}(x) - \bar{\mathsf{F}}_{f_2}(x)$. Therefore, for $(x,t) \in \operatorname{int}(\mathcal{C}_{\tilde{x}})$, we have that $\ell t > \bar{\mathsf{F}}_{f_1}(x) - \gamma^* x \geq \bar{\mathsf{F}}_{f_1}(x) - \bar{\mathsf{F}}_{f_2}(x)$, which implies that $(x,t) \notin \bar{\mathcal{S}}$. Hence, $\operatorname{int}(\mathcal{C}_{\tilde{x}}) \cap \bar{\mathcal{S}} = \varnothing$. Additionally, $\mathcal{C}_{\tilde{x}}$ is convex. These two facts imply that $\mathcal{C}_{\tilde{x}}$ is $\bar{\mathcal{S}}$-free. Since $\mathcal{S} \subseteq \bar{\mathcal{S}}$, $\mathcal{C}_{\tilde{x}}$ is also an $\mathcal{S}$-free set. Next, assume that $(\tilde{x}, \tilde{t}) \notin \bar{\mathcal{S}}$, then $\ell \tilde{t} > \bar{\mathsf{F}}_{f_1}(\tilde{x}) - \bar{\mathsf{F}}_{f_2}(\tilde{x}) \leq \bar{\mathsf{F}}_{f_1}(\tilde{x}) - \gamma^* \tilde{x}$, so $(\tilde{x}, \tilde{t}) \in \operatorname{int}(\mathcal{C}_{\tilde{x}})$. □

In [67,81,90], the authors study the sub/superlevel sets of some DC functions. Their construction of $\mathcal{S}$-free sets relies on a common *reverse-linearization technique*: reverse the set $\mathcal{S}$ by changing the sign of its defining inequality, and linearize one convex function.

In our case, f is an SS function, so we first need to extend the submodular and supermodular components of f. After the extension, we obtain a DC function. We can then apply the reverse-linearization technique to its continuous extension.

6 Application

In this section, we discuss the application of intersection cuts to BMP and D-optimal design. We exploit the submodular structures in these two problems.

6.1 Boolean multilinear constraints

We consider the construction of $\mathcal{S}$-free sets for Boolean multilinear constraints. Since $x \in \{0,1\} \Leftrightarrow x^2 = x$, one can reduce a polynomial function defined on binary variables to a multilinear function, whose monomials do not include powers. For example, $x_1^2 x_3^3 + x_2^2$ can be reduced to $x_1 x_3 + x_2$. A Boolean multilinear function is sometimes called a pseudo Boolean function.

A similar case is the construction of $\mathcal{S}$-free sets for continuous quadratic constraints [67]. We call this construction the "continuous approach". This approach applies eigenvalue decomposition to factor the symmetric matrix representing quadratic terms in a quadratic constraint. Through this factorization, the quadratic constraint is reformulated to a DC constraint, possibly intersected with additional linear constraints. This reformulation is amenable to the reverse-linearization technique. Applying the technique with possibly additional operations, one can construct the so-called continuous-quadratic-free sets [67] [1]. Multilinear terms, however, are represented by tensors. High-order tensor decomposition is more complicated than matrix decomposition [57].

[1] The construction is *de facto* discussed case by case. For some cases, the reverse-linearization technique already suffices to produce continuous-quadratic-free sets. For other cases, one needs additional operations, e.g., projecting out a lineality space. Notably, all cases require the eigenvalue decomposition and its resulting DC constraint.

It is doubtful whether the continuous approach can be extended so as to produce DC functions from tensors.

Here we consider an alternative discrete approach. It exploits the submodularity and the supermodularity of Boolean multilinear functions. In [20,70], a class of Boolean multilinear functions is shown to be supermodular. We give a submodular-supermodular decomposition for general Boolean multilinear functions in the following.

Proposition 9 *Consider a Boolean multilinear function $f : \mathcal{B} \to \mathbb{R}, x \mapsto \sum_{k\in[K]} a_k \prod_{j\in A_k} x_j$ with K multilinear terms, where $A_k \subseteq [n]$. Let $f = f_1 - f_2$ where*

$$f_1(x) := \sum_{\substack{k\in[K]\\ a_k<0}} a_k \prod_{j\in A_k} x_j \tag{15}$$

$$f_2(x) := -\sum_{\substack{k\in[K]\\ a_k>0}} a_k \prod_{j\in A_k} x_j. \tag{16}$$

Then f_1, f_2 are submodular over $\mathcal{B}$.

Proof It follows from Theorem 13.21 of [36] that f_1, f_2 are submodular functions over $\mathcal{B}$. □

Since every Boolean multilinear function is an SS function, we can construct $\mathcal{S}$-free sets for the corresponding Boolean-superlevel set or Boolean-hypograph set.

Corollary 5 *Consider a multilinear function $f : \mathcal{B} \to \mathbb{R}$, where $f(x) = \sum_{k\in[K]} a_i \prod_{j\in A_k} x_j$ for $A_k \subseteq [n]$ as in Prop. 9, and $f_1(x), f_2(x)$ as in Eq. (15)-(16). Let $\mathcal{S}$, $\overline{\mathcal{S}}$, and $\mathcal{C}_{\tilde{x}}$ be as (12), (13), (14), respectively. Then, the set $\mathcal{C}_{\tilde{x}}$ is an $\mathcal{S}$-free set. Moreover, if $\tilde{x} \notin \overline{\mathcal{S}}$, then $\mathcal{C}_{\tilde{x}}$ does not contain $\tilde{x}$ in its interior.*

Proof By Prop. 9, we know that both f_1 and f_2 are submodular. Hence, the result follows by applying Prop. 8. □

Importing the notation in Prop. 9, a BMP problem has the following form:

$$\max \quad t \tag{17a}$$

$$\sum_{k\in\mathcal{K}_0} a_{ik} \prod_{j\in A_k} x_j \geq t \tag{17b}$$

$$\forall i \in [m] \quad \sum_{k\in\mathcal{K}_i} a_{ik} \prod_{j\in A_k} x_j \geq 0 \tag{17c}$$

$$\forall j \in [n] \quad x_j \in \{0,1\}, \tag{17d}$$

where m is the number of constraints, K is the number of distinct multilinear terms in the BMP, $\mathcal{K}_i \subseteq [K]$ is the index set of multilinear terms in the i-th constraint (0 for objective). Unconstrained BMP has several synonyms: PSEUDO BOOLEAN MAXIMIZATION or MULTILINEAR UNCONSTRAINED BINARY OPTIMIZATION (MUBO).

To construct $\mathcal{S}$-free sets for Boolean multilinear constraints in the BMP, we need to write them as the standard form (12). For all $i \in [m]$ or $i = 0$, let

$$f_i(x) := \sum_{k\in\mathcal{K}_i} a_{ik} \prod_{j\in A_k} x_j,$$

and write

$$f_i(x) = f_{i1}(x) - f_{i2}(x),$$

where $f_{i1} := \sum_{k\in\mathcal{K}_i : a_{ik}<0} a_{ik} \prod_{j\in A_k} x_j$ and $f_{i2} := -\sum_{k\in\mathcal{K}_i : a_{ik}>0} a_{ik} \prod_{j\in A_k} x_j$ are two submodular functions.

The objective and constraints of (17) can be represented as

$$f_{i1}(x) - f_{i2}(x) \geq \ell_i t$$

(for all $i \in [m]$, $\ell_i = 0$, and $\ell_0 = 1$), which, by Cor. 5, is in the standard form.

Separating intersection cuts requires LP relaxations or simplicial cones. One can first lift multilinear terms to obtain an extended formulation:

$$\max \quad t \tag{18a}$$

$$\sum_{k\in\mathcal{K}_0} a_{0k}y_k \geq t \tag{18b}$$

$$\forall i \in [m] \quad \sum_{k\in\mathcal{K}_i} a_{ik}y_k \geq 0 \tag{18c}$$

$$k \in [K] \qquad y_k = \prod_{j\in A_k} x_j \tag{18d}$$

$$\forall j \in [n] \qquad x_j \in \{0,1\} \tag{18e}$$

The standard Boolean linearization technique [35] can reformulate a multilinear term $\prod_{j\in A_k} x_j$ by its underestimators and overestimators:

$$\forall j \in A_k \quad y_k \leq x_j \tag{19a}$$

$$y_k \geq |A_k| + 1 - \sum_{j\in A_k} x_j, \tag{19b}$$

where $|A_k|$ is the cardinality of A_k. Then, by linearizing each nonlinear constraint (18d) as linear constraints in (19), one obtains a MILP reformulation of (18).

To construct LP relaxations, one can simply drop the integrality constraints $x_j \in \{0,1\}$. The direct LP relaxation of the MILP reformulation is also an LP relaxation of the BMP (18). Following the method at the end of Sect. 4, we can construct an optimal tableau cone in the extended space (x,y,t). The $\mathcal{S}$-free set belongs to a projected space (i.e., (x,t)-space). By extracting the (x,t) entries of the rays of the optimal tableau cone, we project the optimal tableau cone into the (x,t)-space. Given the projection of this optimal tableau cone, it is straightforward to construct intersection cuts for the BMP: we separate the intersection cuts constructed by means of the $\mathcal{S}$-free sets given by Prop. 8.

As explained above, Boolean quadratic constraints belong to Boolean multilinear constraints, and continuous quadratic constraints relax Boolean quadratic constraints. Both the continuous and discrete approaches can construct valid $\mathcal{S}$-free sets for Boolean quadratic constraints. We remark that maximal continuous-quadratic-free sets are no longer maximally Boolean-quadratic-free. It is easy to see that the discrete approach preserves the term-wise sparsity patterns of the SS functions and requires no factorizations. Therefore, the discrete approach is computationally amenable to ill-conditioned or sparse coefficient matrices.

6.2 D-optimal design

In statistical estimation, optimal designs are a class of experimental designs that are optimal with respect to some statistical criteria. We derive an extended convex MINLP formulation for the Bayesian D-optimal design problem. In this formulation, the problem is a cardinality-constrained submodular maximization problem.

Let $\mathbb{S}^m$ denote the set of m-by-m symmetric matrices, and let $\mathbb{S}^m_+$ (resp. $\mathbb{S}^m_{++}$) denote the set of m-by-m positive semi-definite (resp. positive definite) matrices. Given a set of full row-rank matrices $\{M_j \in \mathbb{R}^{m\times r_k}\}_{j\in[n]}$, an optimal design problem usually has the following form:

$$\max \quad \Phi\Big(\sum_{j\in[n]} M_j {M_j}^\top x_j\Big) \tag{20a}$$

$$\sum_{j\in[n]} x_j = k \tag{20b}$$

$$\forall j \in [n] \qquad x_j \in \{0,1\}, \tag{20c}$$

where k is the size of the design and $\Phi : \mathbb{S}^m \to \mathbb{R}$ is the design criterion. The matrix $M(x) := \sum_{j\in[n]} M_j {M_j}^\top x_j$ is called the *information matrix*. For the D-optimal criterion [21,78], Φ is the log determinant function $\log\det$.

People usually study Bayesian D-optimal design, where a statistical prior on the data $\{M_i\}_{i\in[n]}$ adds a regularization term ϵI into the information matrix $M(x)$. Thus, $M(x) = \epsilon I + \sum_{j\in[n]} M_j {M_j}^\top x_j$. The additional term is also due to the well-posedness: when $x = 0$, we have that $\log\det(M(0)) = \log\det(\epsilon I)$ is well defined. Then, the submodular maximization version of the Bayesian D-optimal design problem has the following formulation:

$$\max \quad \log\det\left(\epsilon I + \sum_{j\in[n]} M_j {M_j}^\top x_j\right) \tag{21a}$$

$$\sum_{j\in[n]} x_j = k \tag{21b}$$

$$\forall j \in [n] \qquad x_j \in \{0,1\}, \tag{21c}$$

The log determinant function is concave and has a semi-definite programming (SDP) and geometric programming representation [8]. The scalability of the mixed-integer log determinant formulation above is limited by the current state of SDP solvers. Based on the second order cone representation of the determinant function $\det(M(x))$ [78], we give an extended formulation for (21):

$$\max \quad t \tag{22a}$$

$$t \le \sum_{i\in[m]} \log(J_{ii}) \tag{22b}$$

$$\sum_{j\in[n]\cup\{0\}} M_j Z_j = J \tag{22c}$$

$$J \text{ is lower triangular} \tag{22d}$$

$$j \in [n]\cup\{0\}\, i \in [m] \qquad \|Z_j e_i\|^2 \le u_{ji} x_j \tag{22e}$$

$$i \in [m] \qquad \sum_{j\in[n]\cup\{0\}} u_{ji} \le J_{ii} \tag{22f}$$

$$\sum_{j\in[n]} x_j = k \tag{22g}$$

$$x \in \{1\} \times \mathcal{B} \tag{22h}$$

$$J \in \mathbb{R}^{m\times m} \tag{22i}$$

$$j \in [n]\cup\{0\} \qquad Z_j \in \mathbb{R}^{r_j\times m} \tag{22j}$$

$$j \in [n]\cup\{0\}\, i \in [m] \qquad u_{ji} \in \mathbb{R}^{r_j\times m}_+, \tag{22k}$$

where $M_0 = \epsilon^{1/2} I$ is an auxiliary matrix. One can represent this formulation by low-dimensional convex cones [8], e.g., (rotated) second-order cones, and exponential cones. Therefore, this extended formulation is amenable to computation.

Proposition 10 (22) *is equivalent to* (21)*, and the objective of* (22) *is submodular w.r.t.* x*.*

Proof One can modify the original D-optimal design problem by adding a slack variable $x_0 = 1$. Applying the logarithmic transformation to results in [78], (22) is equivalent to (21). It follows from [77,82] that (22) is submodular w.r.t. x. □

A global optimization solver like `SCIP` can linearize the constraints in the extended formulation (22), and thus produces an LP relaxation in the extended space. We can obtain an optimal tableau cone as the approach dealing with the BMP. Then, we can construct intersection cuts from $\mathcal{S}$-free sets.

7 Separation problem

In this section, we consider the separation problem for an intersection cut derived from an $\mathcal{S}$-free set. Summarizing the previous sections, the $\mathcal{S}$-free set is in the form of

$$\mathcal{C} := \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : \mathsf{G}(x) \le \ell t\},$$

where $\mathsf{G}(x) = \max_{s \in \mathrm{ext}(\mathsf{EPM}_g)} sx$ is the extended envelope of some submodular function g over $\mathcal{B}$ and $\ell \in \{0,1\}$. We remark that the extended envelope epigraph EE_f in (6) is a special case with $\ell = 1$ and $g = f$; the set $\mathcal{C}_{\tilde{x}}$ in (14) is also a special case that $g(x) = f_1(x) - \gamma^* x$.

Assume that $z^* := (\tilde{x}, \tilde{t})$ is the vertex of an optimal tableau cone $\mathcal{R}$, and $z^* \in \mathrm{int}(\mathcal{C})$. Recalling the cut coefficient formula in Sect. 2, the separation problem consists in computing the step length along each ray r^j:

$$\eta_j^* = \sup_{\eta_j \ge 0} \{\eta_j : z^* + \eta_j r^j \in \mathcal{C}\}. \tag{23}$$

This line search problem asks for the step length to the border of $\mathcal{C}$ along the ray r^j from z^* which, we recall, is an interior point of $\mathcal{C}$. We denote by r_x^j, r_t^j the projection of r^j on x- and t-spaces. Looking at the function defining $\mathcal{C}$, the intersection step length η_j^* is the zero point of the following function:

$$\zeta^j : \mathbb{R}_+ \to \mathbb{R}, \text{ where } \zeta^j(\eta_j) = \ell(\tilde{t} + r_t^j \eta_j) - \mathsf{G}(\tilde{x} + r_x^j \eta_j).$$

This function enjoys the following properties.

Proposition 11 *ζ^j is a concave piece-wise linear function over $[0, +\infty]$ with $\zeta^j(0) > 0$. If $\eta_j^* < \infty$ and there exists an $\eta_j' > 0$ with $\zeta^j(\eta_j') = 0$, then $\eta_j' = \eta_j^*$, i.e., the solution η_j^* must be unique. For all $s^* \in \mathrm{argmax}_{s \in \mathrm{ext}(\mathsf{EPM}_g)} s(\tilde{x} + \eta_j r_x^j)$, $\ell r_t^j - s^* r_x^j$ is a subgradient in $\partial\zeta^j(\eta_j)$. For $\eta_j > \eta_j^*$, $\partial\zeta^j(\eta_j) \le \partial\zeta^j(\eta_j^*)$.*

Proof Since the extended envelope G is the maximum of linear functions, it is convex and piece-wise linear, so ζ^j is concave and piece-wise linear. Since $\zeta^j(0) = \ell\tilde{t} - \mathsf{G}(\tilde{x})$, it follows from the assumption $z^* \in \mathrm{int}(\mathcal{C})$ that $\ell\tilde{t} > \mathsf{G}(\tilde{x})$ and thus $\zeta^j(0) > 0$. Since $\mathcal{C}$ is closed and convex, $\eta_j' = \eta_j^*$ if and only if $z^* + \eta_j' r^j \in \mathrm{bd}(\mathcal{C})$. That is $\mathsf{G}(r_x^j \eta_j + \tilde{x}) = \mathsf{G}(\tilde{x}) + r_t^j \eta_j'$, i.e., $\zeta^j(\eta_j') = 0$. Since $s^* \in \partial\mathsf{G}(\tilde{x} + r_x^j \eta_j)$, by the chain rule, $\ell r_t^j - s^* r_x^j$ is a subgradient of ζ^j. By the concavity of ζ^j, its subgradients are non-increasing. □

By Prop. 11, the line search problem (23) is reduced to solving the univariate nonlinear equation:

$$\zeta^j(\eta^j) = 0. \tag{24}$$

For each ray r^j, solving (24) gives the unique zero point of the univariate function ζ^j, or certifies that no such point exists.

To solve the univariate nonlinear equation (24), it is natural to deploy a Newton-like algorithm. Therefore, we need the value and (sub)gradient information of ζ^j: the computation of ζ^j can then be reduced to the computation of G. The value and subgradients of G are obtained by means of a sorting algorithm (see Prop. 3). We note that these computations can be carried out in strongly polynomial time.

Previous works [25,90] use the bisection algorithm, which guarantees finding the zero point within a given tolerance. Our implementation, which we call *hybrid discrete Newton algorithm*, is a combination of the discrete Newton algorithm [49] and the bisection algorithm. The role of the bisection algorithm in Alg. 1 is to help find a starting point for the Newton algorithm. Thanks to the piece-wise linearity of the univariate function ζ^j, our algorithm finds an exact zero point in a finite time.

Proposition 12 *The hybrid discrete Newton algorithm terminates in a finite number of steps and finds the zero point η_j^*.*

Algorithm 1: Hybrid discrete Newton algorithm

Input: The univariate function ζ^j, (scalar) starting point $\Delta > 0$ (default: 0.2), a numeric η_∞ representing $+\infty$, and the maximum number I of search steps (default: 500);
Output: $\eta_j > 0$ such that $\zeta^j(\eta_j) = 0$;
Let step number $i = 0$, and let step length $\eta_j = \Delta$;
if $\zeta^j(\eta_\infty) > 0$ **then**
  $\eta_j = \eta_\infty$; ▷ safeguard
else
  while $i < I$ **do**
    Let $s^* \in \operatorname{argmax}_{s \in \operatorname{ext}(\mathsf{EPM}_g)} s(\tilde{x} + r_x^j \eta_j)$;
    Compute a subgradient $\beta = r_t^j - s^* r_x^j$;
    if $\zeta^j(\eta_j) = 0$ **then**
      break;
    else if $\beta < 0$ **then**
      $\eta_j = \eta_j - \frac{\zeta^j(\eta_j)}{\beta}$; ▷ Newton step
    else
      $\eta_j = 2\eta_j$; ▷ bisection step
    $i = i + 1$;

Proof For all $\eta \in \mathbb{R}_+$, we assume that Algorithm 1 chooses and computes a unique subgradient β at η_j, we denote it $\nabla\zeta^j(\eta_j)$, and call it algorithmic gradient. The concavity of ζ^j implies that its algorithmic gradient is monotone-decreasing w.r.t. η_j. There is a threshold $\eta'_j \geq 0$ such that, for all $\eta_j \in [0, \eta'_j)$, the algorithmic gradient $\nabla\zeta^j(\eta_j) > 0$; for all $\eta_j \in [\eta'_j, +\infty]$ (called the Newton step region), the algorithmic gradient $\nabla\zeta^j(\eta_j) \leq 0$.

After a finite number of bisection steps (at most $\lceil \log(\eta'_j/\Delta) \rceil$), the algorithm enters the Newton step region $[\eta'_j, +\infty]$, where the algorithmic gradient is always negative. Then, we prove that the algorithmic gradient $\nabla\zeta^j(\eta_j)$ at step i is different from that at step $i-1$, and the algorithm stays in the Newton step region. Since ζ^j is piece-wise linear (the number of its distinct algorithmic gradients is finite), the algorithm must terminate in a finite number of steps.

If at step $i-1$, $\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) = 0$, then the algorithm terminates at this step and finds the zero point. If at step $i-1$, $\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) < 0$, then we prove that $\nabla\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) \neq \nabla\zeta^j(\eta_j)$ and $\nabla\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) \leq 0$.

First, assume, to aim at a contradiction, that $\nabla\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) = \nabla\zeta^j(\eta_j)$. Knowing that the algorithmic gradient is monotone-decreasing, the piece-wise linearity of ζ^j implies that this algorithmic gradient is constant in the range $[\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}, \eta_j]$. It follows that for all $\delta \in [0, \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}]$, $\zeta^j(\eta_j - \delta) = \zeta^j(\eta_j) - \delta\nabla\zeta^j(\eta_j)$. Hence, $\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) = 0$, which leads to a contradiction.

Second, we show that $\nabla\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) \leq 0$. When $\frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)} \leq 0$, by the mononcity of $\nabla\zeta^j$, $\nabla\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) \leq \nabla\zeta^j(\eta_j) < 0$. When $\frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)} > 0$, as by assumption that $\nabla\zeta^j(\eta_j) < 0$, $\zeta^j(\eta_j)$ must be negative. Then, by the concavity of ζ^j, $\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) \leq \zeta^j(\eta_j) - \nabla\zeta^j(\eta_j)\frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)} = 0$. This implies that $\nabla\zeta^j(\eta_j - \frac{\zeta^j(\eta_j)}{\nabla\zeta^j(\eta_j)}) \leq 0$. □

From Prop. 12, the hybrid discrete Newton algorithm first executes bisection steps with increasing η_j and $\zeta^j(\eta_j)$. Then it enters into the Newton step region. After a single Newton step, $\zeta^j(\eta_j)$ becomes negative, and then monotonically increases to zero in a finite number of steps.

The discrete Newton algorithm in [49] is applied to the line search problem for submodular polyhedra, which are related to extended polymatroids. In that context, it runs in a strongly polynomial time. In our case, $\mathcal{C}$ contains the extended polymatroid, but it is unbounded in general. The corresponding line search problem may have no solutions, if the ray r^j is contained in the recession cone of $\mathcal{C}$. Therefore, Algorithm 1 needs a safeguard step, where we evaluate ζ^j at a user-defined infinity. One may also prove that Algorithm 1 runs in a strongly polynomial time, but a careful analysis for the unbounded case is needed.

8 Computational results

In this section, we conduct computational experiments to test the proposed cuts. The source code, data, and detailed results can be found in our online repository: github.com/lidingxu/Subcut.

8.1 Setup and performance metrics

The experiments are conducted on a server with Intel Xeon W-2245 CPU @ 3.90GHz and 126GB main memory. We use `SCIP` 8.0 [17] as a MINLP framework to solve the natural formulations of test problems. `SCIP` is equipped with `CPLEX` 22.1 as an LP solver, and `IPOPT` 3.14 as an NLP solver.

By Thm. 2, the simple lifted split $H_j := \{x \in \mathbb{R}^n : 0 \leq x_j \leq 1\} \times \mathbb{R}$ is a maximal $\text{hypo}_{\mathcal{B}}(f)$-free set for any function f defined over $\mathcal{B}$, where the splitting variable x_j is chosen as the most fractional entry of the relaxation solution. We have three settings of cut separation routines (cut separators). The *submodular cut* (resp. the *split cut*) setting adds intersection cuts derived from $\mathcal{S}$-free sets based on extensions (resp. splits), and the *default* setting does not add any intersection cut. Our separators adhere to unified parameter settings that aim to maximize the likelihood of `SCIP` invoking our separators. We refer to Appendix for detailed parameter settings.

`SCIP` has internal routines of higher authority than any individual cut separator. These routines can control whether to invoke a cut separator and whether to apply the cuts found by the separator. Interfaces of these routines are not exposed publicly, but `SCIP` allows us to affect these routines through the parameters of cut separators. Therefore, we conduct the above three settings respectively in two distinct configurations: the *standalone* and the *embedded* configurations.

In the *standalone* configuration, we aim at measuring the performance of our cuts in a "clean" environment without interacting with other cuts, so we deactivate all of `SCIP`'s internal cut separators. In the *embedded* configuration, we aim at measuring the performance of our cuts in a "real" environment. According to Example 6.10 of [29], our split cuts correspond to Gomory mixed integer cuts. To ensure a fair comparison, we require an equal level of implementation of intersection cuts, including the data structure and parameter settings. Hence, we replace `SCIP`'s implementation of Gomory mixed integer cuts with our own implementation, thereby disabling `SCIP`'s internal Gomory mixed integer cut separators in the embedded configuration.

We focus on the root node performance and measure the *closed root gap*. Let d_1 be the value of the first LP relaxation (without cuts added), let d_2 be the dual bound after all the cuts are added, and let p be a reference primal bound. The closed root gap $(d_2 - d_1)/(p - d_1)$ is the closed gap improvement of d_2 with respect to d_1. We also record the number of added cuts, the relative improvement to the default setting, and the total running time. For each configuration and setting, we compute these statistics' shifted geometric means (SGMs) with a shift of 1 over our test sets.

For each of the following experiments, we present and analyze computational results in the form of tables and scatter plots. The tables contain SGMs of the statistics, including the closed root gap (abbreviated as "closed"), the total running time (abbreviated as "time"), and the number of applied cuts (abbreviated as "cuts"). Moreover, the "relative" column displays the relative value of the closed root gap of one configuration with our cuts to that of the default configuration. Thus, the "relative improvement" due to our cuts is defined as the "relative" minus one. The scatter plots compare the closed root gap of each instance between two different settings. Furthermore, each scatter plot indicates the number of instances where one setting outperforms the other, referred to as "win" instances.

8.2 Results and analysis

Experiment 1: MAX CUT. Consider an undirected graph $G = (V, E, w)$, where V is the set of nodes, E is the set of edges, and w is a weight function over E. For a subset S of V, its associated cut capacity is the sum of the weights of edges with one end node in S and the other end node in $V \setminus S$. The MAX CUT problem aims at finding a subset $S \subseteq V$ with the maximum cut capacity. Let $V = [n]$, and we use a binary variable vector $x \in \mathcal{B}$ indicating whether vertices belong to S.

The problem can be formulated as the following Quadratic Unconstrained Binary Optimization (QUBO) problem:

$$\max_{x\in\mathcal{B}} \sum_{\{i,j\}\in E} w_{ij}((1-x_i)x_j + x_i(1-x_j)).$$

When w is nonnegative, and the cut capacity function (the objective function) is submodular.

The Biq Mac library [89] offers a collection of MAX CUT and QUBO instances of medium size. Our benchmark consists of two sub-benchmarks with 30 "g05" and respectively 30 "pw" MAX CUT instances with nonnegative weights from the library. These instances are generated randomly by Giovanni Rinaldi's `rudy` code [73, 76]. For each dimension $n = 60, 80, 100$, the "g05" sub-benchmark consists of 10 unweighted graphs with edge probability 0.5. For each graph density in $\{0.1, 0.5, 0.9\}$, the "pw" sub-benchmark consists of 10 graphs with integer edge weights chosen from $[0, 10]$.

The reference primal bounds are also from the Biq Mac library. We encode the hypograph reformulation (1) of the QUBO. `SCIP` will automatically reformulate the problem into a MILP via the reformulation-linearization technique (RLT) [4]. This MILP formulation is a special case of the extended formulation (22) of a degree-2 BMP with $m = 0$.

For the standalone configuration, the relative improvement of submodular cuts is 342% compared to 178% of split cuts. In this configuration, we can compare the "clean" strengths of intersection cuts derived from different $\mathcal{S}$-free sets. As observed from the scatter plots in Fig. 2, the submodular cut setting outperforms the split cut setting in 42 instances under the standalone configuration. Although split cuts are derived from maximal $\mathcal{S}$-free sets and submodular cuts are derived from non-maximal ones, the clean performance of split cuts is worse. Regarding the embedded configuration, the relative improvement of submodular cuts is 85%, compared to 58% of split cuts. The scatter plot shows that the submodular cut setting surpasses the split cut setting in 34 instances under this configuration.

We observe that fewer split cuts are generated than submodular cuts. This means that the efficiency of some split cuts does not satisfy `SCIP`'s internal criteria, so `SCIP` abandons more split cuts than submodular cuts. As two types of cuts are derived using the same principle but from different $\mathcal{S}$-free sets, the distances between the relaxation points to the boundary of $\mathcal{S}$-free sets determine the cut efficiency. This observation suggests that relaxation points are further from the boundary of the extended envelope epigraph than from the splits. The separation time of split cuts is shorter than that of submodular cuts, particularly for the "pw" instances with a high graph density (0.9). This is because separating submodular cuts requires solving nonlinear equations that involve sorting and computing graph cuts, while the split cuts can be computed in a closed form.

Configuration	Default		Submodular cut				Split cut			
	closed	time	closed	relative	time	cuts	closed	relative	time	cuts
standalone	0.026	4.33	0.111	4.418	22.9	215.48	0.075	2.78	7.04	75.04
embedded	0.097	4.77	0.161	1.852	68.19	162.5	0.139	1.575	9.4	67.6

Table 1: Summary of MAX CUT results

Experiment 2: PSEUDO BOOLEAN MAXIMIZATION. As mentioned, PSEUDO BOOLEAN MAXIMIZATION is a MUBO problem, a generalization of QUBO. We can use techniques from Sect. 5 to generate intersection cuts.

POLIP [72] is a library of polynomially constrained mixed-integer programming instances. All MUBO instances in POLIP with degree higher than 2 are 41 "autocorr_bern" instances, which are also included in MINLPLib [22, 88]. These instances arise from short ranged non-disordered lattice spin model (the Bernasconi model) [61] in theoretical physics. The problem is to determine a ground state in the Bernasconi model minimizing a degree-four energy polynomial: $\frac{n}{n-r+1}\sum_{i=0}^{n-r}\frac{1}{r(r-1)}\sum_{d=1}^{r-1}(\sum_{j=i}^{i+r-1-d} z_j z_{j+d})^2$, where $z \in \{-1, 1\}^n$. The number n of variables in these instances is chosen from 20 to 60, and the interaction range r is chosen from 3 to 6. The problem is reformulated into a degree-4 BMP with $m = 0$ in MINLPLib through the transformation $z_j = 2x_j - 1$. `SCIP` constructs the extended formulation (22). We use the best-known primal bound from MINLPLib as the reference primal bound.

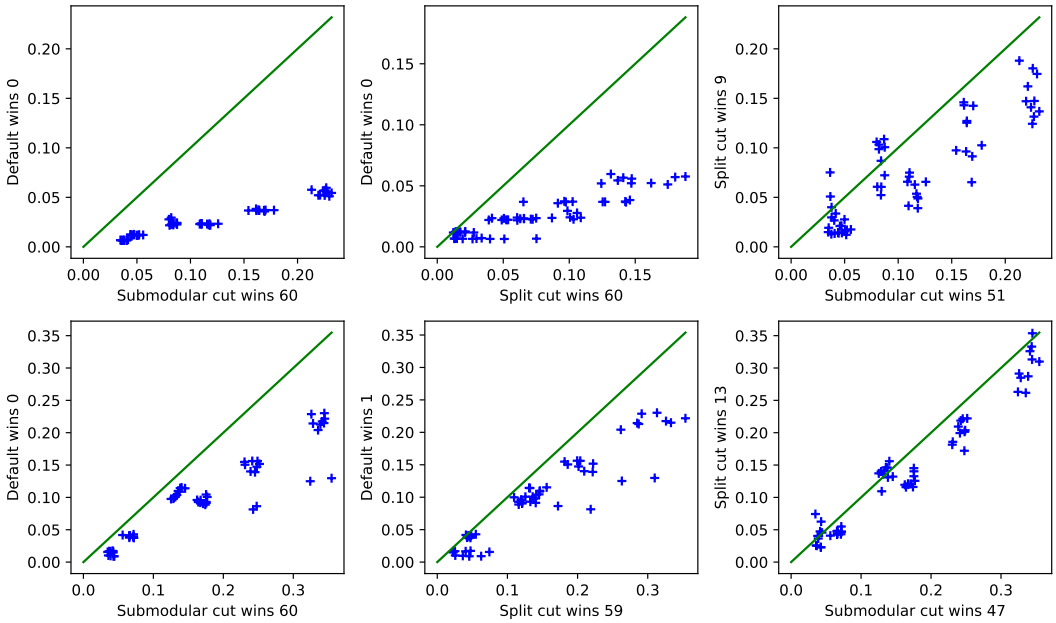

Fig. 2: Closed root gaps of MAX CUT instances in standalone (top) and embedded (bottom) configurations

In Table 2, we report the computational results. For the standalone (resp. the embedded) configuration, the relative improvement of submodular cuts is 504% compared to 117% of split cuts. As indicated by the scatter plots in Fig. 3, the submodular cut setting outperforms the split cut setting in 29 instances under the standalone configuration. Regarding the embedded configuration, the relative improvement of submodular cuts is 98%, compared to 49% of split cuts. As indicated by the scatter plots, the submodular cut setting wins in 31 more instances than the spit cut setting under this configuration.

In both configurations, the submodular cuts are better than the split cuts in terms of the closed root gap. Moreover, under the embedded configuration, the difference in the relative improvements between submodular cuts and split cuts is 48%. This is larger than 28% of MAX CUT benchmark under the same configuration. This divergence between degree-2 and degree-4 MUBO suggests that the submodular cuts are suitable for high-order Boolean multilinear constraints.

We recall that to solve the nonlinear equations, the hybrid discrete Newton algorithm needs oracle access to the value of the Boolean multilinear function. For some instances, a Boolean multilinear function may consist of thousands of multilinear terms. After a code timing analysis, we find that the separation of submodular cuts spends the most time computing the function value. Therefore, this is the main time performance bottleneck, which needs to be optimized in the future. Similar to MAX CUT results, non-maximal $\mathcal{S}$-free sets may yield stronger cuts. Thus, the geometrical relation between the $\mathcal{S}$-free sets and the optimal tableau cone matters.

In Sect. 8.3, we conduct a branch-and-bound test.

Configuration	Default		Submodular cut				Split cut			
	closed	time	closed	relative	time	cuts	closed	relative	time	cuts
standalone	0.008	8.46	0.053	6.039	28.03	68.83	0.032	2.170	10.56	20.48
embedded	0.051	13.60	0.079	1.979	46.20	28.20	0.067	1.491	20.43	9.43

Table 2: Summary of PSEUDO BOOLEAN MAXIMIZATION results

Experiment 3: BAYESIAN D-OPTIMAL DESIGN. As mentioned before, the BAYESIAN D-OPTIMAL DESIGN problem has a submodular maximization form (21). In particular, we can encode it as an

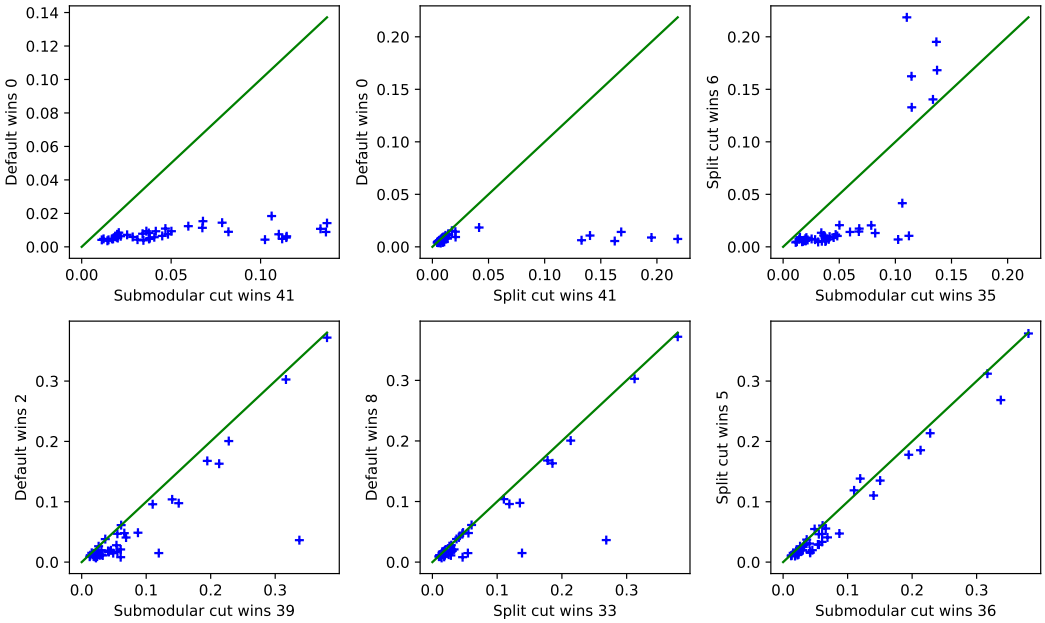

Fig. 3: Closed root gaps of PSEUDO BOOLEAN MAXIMIZATION instances in standalone (top) and embedded (bottom) configurations

extended formulation (22) in SCIP. SCIP generates gradient cuts for this convex MINLP. Therefore, we can obtain LP relaxations and simplicial conic relaxations.

Our benchmark consists of two sub-benchmarks. Recall that the binary vector x selects a subset of $\{M_j M_j{}^\top \in \mathbb{S}^m\}_{j\in[n]}$, and the information matrix $M(x)$ in (21) is the sum of matrices in the selected subset. Thus, the problem data of (21) are variable dimension n, matrix size m, cardinality number k, and matrices $M_j M_j{}^\top$. We next outline the procedure for generating them.

The first sub-benchmark consists of 15 block design instances. We follow the method in [78] to generate these instances. Let $x := (x_{1,2}, x_{1,3}, \ldots, x_{1,m+1}, \ldots, x_{m,m+1}) \in \{0,1\}^n$, where $n = \binom{m+1}{2}$. Let $H(x)$ be an undirected graph with $m+1$ vertices. If $x_{i,i'} = 1$, there is an edge between vertices i, i'; otherwise, no edge connects them. We have $M(x) = PL(x)P^\top$, where $L(x) := \sum_{i,i'} x_{i,i'}(\mathbf{1}_i - \mathbf{1}_{i'})(\mathbf{1}_i - \mathbf{1}_{i'})^\top \in \mathbb{S}^m \subseteq \mathbb{R}^{m\times m}$ is the Laplacian of $H(x)$, and $P \in \mathbb{R}^{m\times(m+1)}$ is the matrix that transforms an $(m+1)$-dimensional vector v to the vector obtained by keeping the first m entries of v. In other words, $M(x)$ is the submatrix of the Laplacian of $H(x)$ obtained by removing its last row and last column. Then an optimal solution to (20) corresponds to the graph with n nodes and k edges that has a maximum number of spanning trees. Note that $M_j = P(\mathbf{1}_i - \mathbf{1}_{i'}) \in \mathbb{R}^{m\times 1}$ is a single-column matrix, which is degenerated to an m-dimensional vector. We generate a block design instance for each combination of $m \in \{10, 11, 12\}, n = \binom{m+1}{2}, k \in \{m, m+1, m+2, m+3, m+4\}$. This results in a total of 15 combinations.

The second sub-benchmark consists of 30 random Gaussian instances. We generate a Guassian instance for each combination of $(n, m) \in \{(50, 20), (50, 30), (60, 24), (60, 36), (70, 28), (70, 42)\}$ and $k \in \{m, m+1, m+2, m+3, m+4\}$. This results in a total of 30 combinations. We still let each M_j be a single-column matrix (i.e., a vector), and its entries are drawn from a Gaussian distribution with zero mean and a variance of $1/\sqrt{n}$.

We set the regularization constant ϵ to 10^{-6}. SCIP can find primal feasible solutions at the root node using its internal heuristics. We select the best primal bound given by these solutions among all settings as the reference primal bound. Since SCIP's internal gradient cuts are important for linearizing convex nonlinear constraints, we keep the gradient cuts but disable all integer-oriented cuts (GMI cuts and mixed-integer rounding cuts etc.) in the standalone configuration.

In Table 3, we report the computational results. We divide the results of block design and Gaussian random instances, since the density of matrices are different. Looking at the default setting in different benchmarks, there is no difference between the standalone and embedded

configurations in terms of the closed root gap. This means that integer-oriented cuts do not improve the root node LP relaxations. We see the same problem for intersection cuts, which do not close the root gap but increase the computing time. In particular, the number of separated cuts is around one. Thereby, many intersection cuts are too weak to add to the cut pool.

We recall that intersection cuts and many integer-oriented cuts are LP-based cuts, i.e., derived from an LP relaxation of the extended formulation (22). Therefore, their strengths depend on the LP relaxation. Based on the types of MINLPs, there are two basic ways to construct initial LP relaxations. For nonconvex MINLPs, one way usually uses the factorable programming and term-wise envelopes [63]. Notable examples are Boolean multilinear constraints and continuous quadratic constraints [67]. The McCormick envelopes or Boolean linearization techniques are used to construct their LP relaxations, which have a finite number of constraints. In Fig. 4, the scatter plots also show that there are nearly no instance-wise difference between configurations.

For convex MINLPs, the other way linearizes nonlinear constraints, and the number of constraints in the LP relaxation can grow to infinity. This is because a convex nonlinear constraint is equivalent to an infinite number of linear constraints. Given that `SCIP` may incorporate numerous gradient cuts to approximate the convex MINLP (22), we can better understand its behavior in a simplified scenario. Consider a smooth convex body approximated by a polyhedral outer approximation, where each vertex and its associated faces define one or several simplicial cones, representing optimal tableau cones. As the polyhedron closely approximates the convex body, the vertex comes closer to the convex body, and the simplicial cones approach its tangent space at that vertex. Consequently, the cones become very flat, and in the most extreme case, they turns into a hyperplane defining the tangent space. When a hyperplane intersects an $\mathcal{S}$-free set, this results in the hyperplane itself. Therefore, it is highly likely that our separators will generate weak intersection cuts. In summary, the weakness of intersection cuts is due to the flatness of the optimal tableau cone.

Benchmark	Configuration	Default		Submodular cut				Split cut			
		closed	time	closed	relative	time	cuts	closed	relative	time	cuts
Block design	standalone	0.59	20.46	0.59	1.0	18.71	1.84	0.59	1.0	11.62	1.77
	embedded	0.59	21.44	0.59	1.0	19.0	1.84	0.59	1.0	12.41	1.77
Gaussian	standalone	0.83	213.13	0.83	1.0	415.07	1.45	0.83	1.0	214.17	1.45
	embedded	0.83	214.77	0.83	1.0	426.33	1.45	0.83	1.0	214.14	1.45
All	standalone	0.75	98.47	0.75	1.0	149.54	1.57	0.75	1.0	82.6	1.55
	embedded	0.75	100.47	0.75	1.0	153.01	1.57	0.75	1.0	84.31	1.55

Table 3: Summary of Bayesian D-optimal design results

8.3 Supplementary branch-and-bound computational results

We also have an additional branch-and-bound test for max cut problem instances. This test was designed to assess the performance and properties of our cuts in a "production-level" environment, which presents more complex challenges compared to the root node experiment. As such, the parameter settings and analysis in this test are more intricate and require a detailed explanation, which we provide below.

We conducted our tests under the embedded configuration, where the branching rule, node selection rule, and primal heuristics adhere to `SCIP`'s defaults. In addition to the parameters mentioned in Appendix, we made adjustments to other parameters specifically to control the behavior of our cut separators in the branch-and-bound algorithm. These parameters are as follows:

- SEPA_FREQ: the default frequency for separating cuts. We set it to 0, meaning that our cut separators are called at the root node.
- SEPA_NCUTSLIMITROOT: the limit for the number of cuts generated at the root node. We set it to 60.
- SEPA_MAXBOUNDDIST: the default maximal relative distance from the current node's dual bound to primal bound compared to the best node's dual bound for applying separation. We set it to 1, meaning that separation is applied at all search nodes.

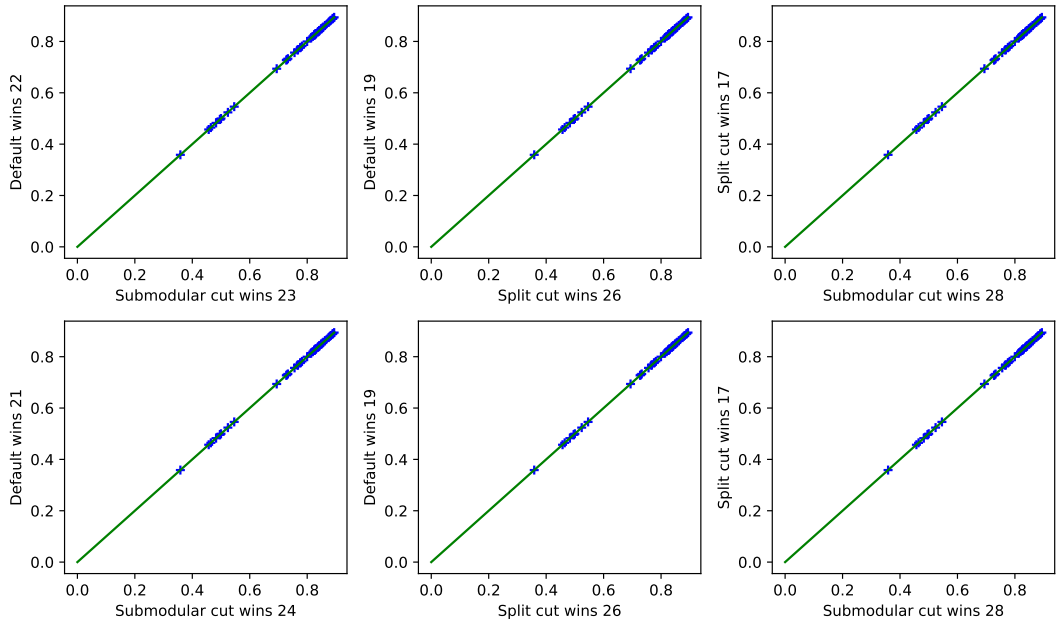

Fig. 4: Closed root gaps of Bayesian D-optimal design instances in standalone (top) and embedded (bottom) configurations

Due to the substantial number of parameter combinations, tuning the parameters for the branch-and-bound test is more challenging compared to the root node experiment. For instance, SCIP's internal Gomory mixed-integer cut separator [1,33] is limited to applying at most 30 cuts at the root node, while SCIP's quadratic intersection cut separator [3,24] employs at most 20 cuts at the root node and 2 cuts at each non-root node.

In a preliminary branch-and-bound test, we find that even the default setting can solve the "pw" instances of density 0.1 within 100 seconds, while for all the other instances, all settings time out (gaps are not closed after 3600 seconds). To have an unbiased result, we remove "pw" instances of density 0.1 and create a sub-benchmark called max cut-sub.

For the following branch-and-bound test, we measure the closed duality gap (abbreviated as gap), the relative improvement of the closed duality gap to the default setting, the number of search nodes (abbreviated as nodes), and the number of applied cuts. The aggregated results are summarized in Table 4.

Benchmark	Default		Submodular cut				Split cut			
	gap	nodes	gap	relative	nodes	cuts	gap	relative	nodes	cuts
max cut-sub	0.605	231372	0.596	0.981	220176	59.87	0.618	1.026	207078	42.2

Table 4: Summary of max cut-sub results in the embedded branch-and-bound test

The results indicate that the submodular cut setting performs slightly worse than the default setting, while the split cut setting performs marginally better than the default setting. As shown in Fig. 5, the difference in closed duality gaps between the submodular/split cut and default settings is no more than 2%. This shows that the optimization landscape of max cut problems is very complicated. As for our parameter settings, intersection cuts cannot have a significant impact on the branch-and-bound algorithm.

In contrast to the results in the root node experiment, we find that the split cut setting outperforms the submodular cut setting in the branch-and-bound test. Detailed cut information obtained during debugging reveals that the condition number of submodular cuts can be thousands of times larger than that of split cuts, and the submodular cuts can be denser as

well. Consequently, these numerical properties make the submodular cuts less stable and efficient compared to the split cuts.

When considering the approximation of the Boolean-hypograph $\text{hypo}_{\mathcal{B}}(g)$, where g represents any function over $\mathcal{B}$, we can deduce from Thm. 2 that the splits define a class of maximal $\mathcal{S}$-free sets. Although the split cuts are independent of the values of g, we can use the split cuts to approximate the Boolean-hypograph of g. While one can find other $\mathcal{S}$-free sets based on the values of g, the resulting cuts will likely exhibit the same numerical properties as our submodular cuts.

As a result, future research should consider this finding when exploring intersection cuts. However, it is also worthwhile to investigate the performance of submodular cuts in other problems and algorithms, such as PSEUDO BOOLEAN MAXIMIZATION problems and the diving heuristic.

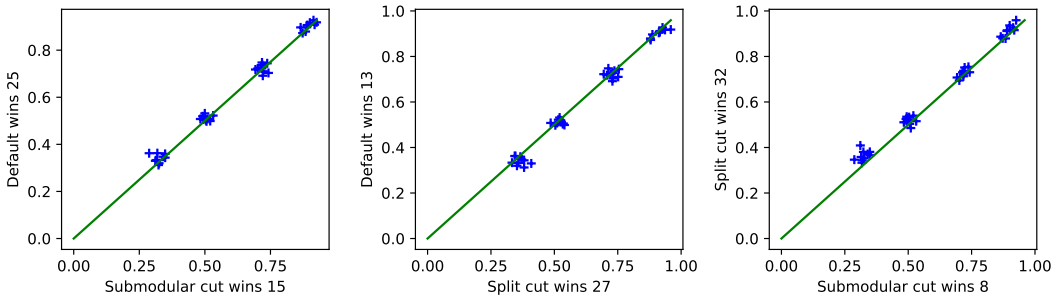

Fig. 5: Closed root gaps of MAX CUT-sub instances in the embedded branch-and-bound test

9 Conclusion

We construct $\mathcal{S}$-free sets for the Boolean-hypographs of submodular functions. Our construction relies on a new continuous extension of submodular functions. We characterize maximal $\mathcal{S}$-free sets, generalize our results to sets involving submodular-supermodular functions. These yield intersection cuts for Boolean multilinear constraints. We exploit the submodular structure in an extended formulation of the D-OPTIMAL DESIGN problem. We propose a hybrid discrete Newton algorithm that can compute intersection cuts efficiently and exactly. The computational results show that intersection cuts derived from the submodularity are better than those derived from split cuts for MAX CUT and PSEUDO BOOLEAN MAXIMIZATION problems in the root-node experiments. For convex MINLPs, our computational results on the BAYESIAN D-OPTIMAL DESIGN problem suggest that simplicial conic relaxations given by gradient cuts can be flat, which makes intersection cuts weak.

Acknowledgements

We thank Antonia Chmiela for information about the implementation of intersection cuts in `SCIP`. We also thank two anonymous reviewers for many helpful comments.

Statements and Declarations

We have no conflict of interest with the journal or the funding agencies.

Appendix A Basic experimental settings

For all the test problems presented in Sect. 8, we have utilized a consistent set of parameters for our cut separators. These parameters for the cut separators in `SCIP` are detailed in [2]. Notably, during our experiments, we observed that the cut separators are predominantly influenced by the following parameters:

- SEPA_PRIORITY: the priority of the intersection cut separator. We set it to 100000 (the separators are called in a predefined order, which is given by the priorities of the separators).
- SEPA_DELAY: the default for whether the separation method should be delayed, if other separators found cuts. We set it to TRUE, i.e., delayed. (If the separator's separation method is marked to be delayed, it is only executed after no other separator found a cut during the price-and-cut loop).
- SEPA_MINVIOL: the minimal violation a cut must fulfill such that the cut can be added. We set it to 10^{-4}.
- SEPA_NCUTSLIMITROOT: the limit for the number of cuts generated at the root node. We set it to -1, meaning that the separation is unlimited.

Most cut separators in `SCIP` have priorities lower than 15, leading us to assign the highest priority to our cut separators. Consequently, `SCIP` calls our cut separators before the others during the optimization process.

The proposed cuts in this paper are represented by the expression $\alpha x + \mu t \leq \beta$. When constructing a cut of this form to separate a point $(\tilde{x}, \tilde{t})$, it is considered numerically *ill-conditioned*, if the condition number $\max(\alpha, \mu)/\min(\alpha, \mu)$ becomes too large.

The objective of the proposed cuts is to approximate the constraint $f(x) \geq \ell t$, where f represents either a submodular function or an SS function. During our analysis, we observed that the magnitude of $f(x)$ can be significantly larger than 1. For submodular cuts, this leads to a numerically ill-conditioned cut, where the magnitude of μ is much smaller than the magnitudes of the entries in α.

A similar issue arises in the numerical optimization of finite sums of nonlinear functions, such as problems of the form $\min g(x) := \min \sum_{j\in[k]} g_j(x)$. To enhance numerical stability during optimization, it is more favorable to optimize the average $g(x)/k$ rather than $g(x)$ itself. Therefore, we adopt a similar pre-processing step to scale our test problems.

Specifically, we scale the constraint $f(x) \geq \ell t$ into $f(x)/\chi \geq \ell t$, where χ represents a positive scaling factor. The purpose of this step is to ensure that the magnitude of μ becomes similar to that of α and β. The factor χ is selected as follows:

- For MAX CUT problems, χ is the number of edges of the graph.
- For PSEUDO BOOLEAN MAXIMIZATION problems, χ is the number of degree-4 monomials in the polynomial.
- For D-OPTIMAL DESIGN problems, χ is 1.

References

1. SCIP Doxygen Documentation: examples/GMI/src/sepa_gmi.c Source File (2023). URL `https://scipopt.org/doc-8.0.3/html/sepa__gmi_8c_source.php`. [Online; accessed 11. Jul. 2023]
2. SCIP Doxygen Documentation: How to add separators (2023). URL `https://www.scipopt.org/doc-8.0.3/html/SEPA.php`. [Online; accessed 6. Jul. 2023]
3. SCIP Doxygen Documentation: nlhdlr_quadratic.c Source File (2023). URL `https://www.scipopt.org/doc-8.0.3/html/nlhdlr__quadratic_8c_source.php`. [Online; accessed 14. Jul. 2023]
4. Adams, W.P., Sherali, H.D.: A tight linearization and an algorithm for zero-one quadratic programming problems. Management Science **32**(10), 1274–1290 (1986)
5. Ahmed, S., Atamtürk, A.: Maximizing a class of submodular utility functions. Mathematical programming **128**(1), 149–169 (2011)
6. Andersen, K., Louveaux, Q., Weismantel, R.: An analysis of mixed integer linear sets based on lattice point free convex sets. Mathematics of Operations Research **35**(1), 233–256 (2010)
7. Andersen, K., Louveaux, Q., Weismantel, R., Wolsey, L.A.: Inequalities from two rows of a simplex tableau. In: M. Fischetti, D.P. Williamson (eds.) Integer Programming and Combinatorial Optimization, pp. 1–15. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
8. ApS, M.: Mosek modeling cookbook (2020)

9. Atamtürk, A., Gómez, A.: Submodularity in conic quadratic mixed 0–1 optimization. Operations Research **68**(2), 609–630 (2020)
10. Atamtürk, A., Gómez, A.: Supermodularity and valid inequalities for quadratic optimization with indicators. Mathematical Programming pp. 1–44 (2022)
11. Atamtürk, A., Narayanan, V.: Submodular function minimization and polarity. Mathematical Programming (2021)
12. Balas, E.: Intersection cuts—a new type of cutting planes for integer programming. Operations Research **19**(1), 19–39 (1971)
13. Basu, A., Conforti, M., Cornuéjols, G., Zambelli, G.: Maximal lattice-free convex sets in linear subspaces. Mathematics of Operations Research **35**(3), 704–720 (2010)
14. Basu, A., Conforti, M., Cornuéjols, G., Zambelli, G.: Maximal lattice-free convex sets in linear subspaces. Mathematics of Operations Research **35**(3), 704–720 (2010)
15. Basu, A., Dey, S.S., Paat, J.: Nonunique lifting of integer variables in minimal inequalities. SIAM Journal on Discrete Mathematics **33**(2), 755–783 (2019)
16. Belotti, P., Góez, J.C., Pólik, I., Ralphs, T.K., Terlaky, T.: A conic representation of the convex hull of disjunctive sets and conic cuts for integer second order cone optimization. In: Numerical Analysis and Optimization, pp. 1–35. Springer (2015)
17. Bestuzheva, K., Besançon, M., Chen, W.K., Chmiela, A., Donkiewicz, T., van Doornmalen, J., Eifler, L., Gaul, O., Gamrath, G., Gleixner, A., et al.: The scip optimization suite 8.0. arXiv preprint arXiv:2112.08872 (2021)
18. Bestuzheva, K., Chmiela, A., Müller, B., Serrano, F., Vigerske, S., Wegscheider, F.: Global optimization of mixed-integer nonlinear programs with scip 8. arXiv preprint arXiv:2301.00587 (2023)
19. Bienstock, D., Chen, C., Munoz, G.: Outer-product-free sets for polynomial optimization and oracle-based cuts. Mathematical Programming **183**(1), 105–148 (2020)
20. Billionnet, A., Minoux, M.: Maximizing a supermodular pseudoboolean function: A polynomial algorithm for supermodular cubic functions. Discrete Applied Mathematics **12**(1), 1–11 (1985)
21. Bouhtou, M., Gaubert, S., Sagnol, G.: Submodularity and randomized rounding techniques for optimal experimental design. Electronic Notes in Discrete Mathematics **36**, 679–686 (2010)
22. Bussieck, M.R., Drud, A.S., Meeraus, A.: Minlplib—a collection of test models for mixed-integer nonlinear programming. INFORMS Journal on Computing **15**(1), 114–119 (2003)
23. Chen, R., Dash, S., Günlük, O.: Multilinear sets with two monomials and cardinality constraints. Discrete Applied Mathematics **324**, 67–79 (2023)
24. Chmiela, A., Muñoz, G., Serrano, F.: On the implementation and strengthening of intersection cuts for qcqps. Mathematical Programming pp. 1–38 (2022)
25. Chmiela, A., Muñoz, G., Serrano, F.: Monoidal strengthening and unique lifting in miqcps. In: Integer Programming and Combinatorial Optimization: 24th International Conference, IPCO 2023 (2023). Accepted for publication
26. Coey, C., Lubin, M., Vielma, J.P.: Outer approximation with conic certificates for mixed-integer convex problems. Mathematical Programming Computation **12**(2), 249–293 (2020)
27. Conforti, M., Cornuéjols, G.: Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem. Discrete applied mathematics **7**(3), 251–274 (1984)
28. Conforti, M., Cornuéjols, G., Zambelli, G.: Integer programming. Springer International Publishing, Cham (2014)
29. Conforti, M., Cornuéjols, G., Zambelli, G., et al.: Integer programming, vol. 271. Springer (2014)
30. Conforti, M., Cornuéjols, G., Daniilidis, A., Lemaréchal, C., Malick, J.: Cut-Generating Functions and S-Free Sets. Mathematics of Operations Research **40**(2), 276–391 (2015). DOI 10.1287/moor.2014.0670
31. Conforti, M., Cornuéjols, G., Zambelli, G.: Corner polyhedron and intersection cuts. Surveys in Operations Research and Management Science **16**(2), 105–120 (2011)
32. Coniglio, S., Furini, F., Ljubić, I.: Submodular maximization of concave utility functions composed with a set-union operator with applications to maximal covering location problems. Mathematical Programming pp. 1–48 (2022)
33. Cornuéjols, G., Margot, F., Nannicini, G.: On the safety of gomory cut generators. Mathematical Programming Computation **5**, 345–395 (2013)
34. Cornuéjols, G., Wolsey, L., Yıldız, S.: Sufficiency of cut-generating functions. Mathematical Programming **152**(1), 643–651 (2015)
35. Crama, Y.: Concave extensions for nonlinear 0–1 maximization problems. Mathematical Programming **61**(1), 53–60 (1993)
36. Crama, Y., Hammer, P.L.: Boolean functions: Theory, algorithms, and applications. Cambridge University Press (2011)
37. Del Pia, A., Khajavirad, A.: A polyhedral study of binary polynomial programs. Mathematics of Operations Research **42**(2), 389–410 (2017)
38. Del Pia, A., Khajavirad, A.: The multilinear polytope for acyclic hypergraphs. SIAM Journal on Optimization **28**(2), 1049–1076 (2018)
39. Del Pia, A., Khajavirad, A., Sahinidis, N.V.: On the impact of running intersection inequalities for globally solving polynomial optimization problems. Mathematical programming computation **12**(2), 165–191 (2020)
40. Del Pia, A., Walter, M.: Simple odd-cycle inequalities for binary polynomial optimization. In: International Conference on Integer Programming and Combinatorial Optimization, pp. 181–194. Springer (2022)
41. Del Pia, A., Weismantel, R.: Relaxations of mixed integer sets from lattice-free polyhedra. 4OR **10**(3), 221–244 (2012)

42. Dey, S.S., Wolsey, L.A.: Lifting integer variables in minimal inequalities corresponding to lattice-free triangles. In: A. Lodi, A. Panconesi, G. Rinaldi (eds.) Integer Programming and Combinatorial Optimization, pp. 463–475. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
43. Edmonds, J.: Submodular functions, matroids, and certain polyhedra. In: Combinatorial Optimization—Eureka, You Shrink!, pp. 11–26. Springer (2003)
44. en:User:Cburnett: Hamming distance 3 bit binary (2007). URL `https://commons.wikimedia.org/wiki/File:Hamming_distance_3_bit_binary.svg`. The image is licensed under CC BY-SA 3.0
45. Fischetti, M., Ljubić, I., Monaci, M., Sinnl, M.: On the use of intersection cuts for bilevel optimization. Mathematical Programming **172**(1), 77–103 (2018)
46. Fischetti, M., Monaci, M.: A branch-and-cut algorithm for mixed-integer bilinear programming. European Journal of Operational Research **282**(2), 506–514 (2020)
47. Fortet, R.: Applications de l'algebre de boole en recherche opérationelle. Revue Française de Recherche Opérationelle **4**(14), 17–26 (1960)
48. Glover, F.: Convexity cuts and cut search. Operations Research **21**(1), 123–134 (1973). DOI 10.1287/opre.21.1.123
49. Goemans, M.X., Gupta, S., Jaillet, P.: Discrete newton's algorithm for parametric submodular function minimization. In: F. Eisenbrand, J. Koenemann (eds.) Integer Programming and Combinatorial Optimization, pp. 212–227. Springer International Publishing, Cham (2017)
50. Gomory, R.E.: Some polyhedra related to combinatorial problems. Linear algebra and its applications **2**(4), 451–558 (1969)
51. Gomory, R.E.: Outline of an algorithm for integer solutions to linear programs and an algorithm for the mixed integer problem. In: M. Jünger, T.M. Liebling, D. Naddef, G.L. Nemhauser, W.R. Pulleyblank, G. Reinelt, G. Rinaldi, L.A. Wolsey (eds.) 50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art, pp. 77–103. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
52. Han, S., Gómez, A., Prokopyev, O.A.: Fractional 0–1 programming and submodularity. Journal of Global Optimization pp. 1–17 (2022)
53. Hiriart-Urruty, J.B., Lemaréchal, C.: Fundamentals of convex analysis. Springer Science & Business Media (2004)
54. Horst, R., Tuy, H.: Global Optimization: Deterministic Approaches. Springer, Berlin (1990)
55. Khamisov, O.: On optimization properties of functions, with a concave minorant. Journal of Global Optimization **14**(1), 79–101 (1999)
56. Kılınç-Karzan, F., Küçükyavuz, S., Lee, D.: Joint chance-constrained programs and the intersection of mixing sets through a submodularity lens. Mathematical Programming pp. 1–44 (2021)
57. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM review **51**(3), 455–500 (2009)
58. Kılınç-Karzan, F., Yıldız, S.: Two-term disjunctions on the second-order cone. Mathematical Programming **154**(1-2), 463–491 (2015)
59. Kılınç-Karzan, F.: On minimal valid inequalities for mixed integer conic programs. Mathematics of Operations Research **41**(2), 477–510 (2016). DOI 10.1287/moor.2015.0737
60. Liberti, L.: Spherical cuts for integer programming problems. International Transactions in Operational Research **15**(3), 283–294 (2008)
61. Liers, F., Marinari, E., Pagacz, U., Ricci-Tersenghi, F., Schmitz, V.: A non-disordered glassy model with a tunable interaction range. Journal of Statistical Mechanics: Theory and Experiment **2010**(05), L05003 (2010)
62. Lovász, L.: Submodular functions and convexity. In: Mathematical programming the state of the art, pp. 235–257. Springer (1983)
63. McCormick, G.P.: Computability of global solutions to factorable nonconvex programs: Part i—convex underestimating problems. Mathematical programming **10**(1), 147–175 (1976)
64. Modaresi, S., Kilinç, M.R., Vielma, J.P.: Split cuts and extended formulations for Mixed Integer Conic Quadratic Programming. Operations Research Letters **43**(1), 10–15 (2015)
65. Modaresi, S., Kılınç, M.R., Vielma, J.P.: Intersection cuts for nonlinear integer programming: convexification techniques for structured sets. Mathematical Programming **155**(1-2), 575–611 (2016)
66. Muñoz, G., Paat, J., Serrano, F.: Towards a characterization of maximal quadratic-free sets. arXiv preprint arXiv:2211.05185 (2022)
67. Muñoz, G., Serrano, F.: Maximal quadratic-free sets. Mathematical Programming **192**(1), 229–270 (2022)
68. Murota, K.: Discrete convex analysis. Mathematical Programming **83**(1), 313–371 (1998)
69. Nemhauser, G., Wolsey, L.: Matroid and Submodular Function Optimization (1988)
70. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions—i. Mathematical programming **14**(1), 265–294 (1978)
71. Nesterov, Y., et al.: Lectures on convex optimization, vol. 137. Springer (2018)
72. Pagacz, U.: POLIP: Library for polynomially constrained mixed-integer programming (2023). URL `https://polip.zib.de/`. Online accessed
73. Rendl, F., Rinaldi, G., Wiegele, A.: Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. Mathematical Programming **121**, 307–335 (2010)
74. Rhys, J.M.: A selection problem of shared fixed costs and network flows. Management Science **17**(3), 200–207 (1970)
75. Richard, J.P.P., Dey, S.S.: The group-theoretic approach in mixed integer programming. In: 50 Years of Integer Programming 1958-2008, pp. 727–801. Springer (2010)
76. Rinaldi, G.: Rudy. http://www-user. tu-chemnitz. de/helmberg/rudy. tar. gz (1998)
77. Sagnol, G.: Approximation of a maximum-submodular-coverage problem involving spectral functions, with application to experimental designs. Discrete Applied Mathematics **161**(1-2), 258–276 (2013)
78. Sagnol, G., Harman, R.: Computing exact d-optimal designs by mixed integer second-order cone programming. The Annals of Statistics **43**(5), 2198–2224 (2015)

79. Saxena, A., Bonami, P., Lee, J.: Convex relaxations of non-convex mixed integer quadratically constrained programs: projected formulations. Mathematical programming **130**(2), 359–413 (2011)
80. Schrijver, A., et al.: Combinatorial optimization: polyhedra and efficiency, vol. 24. Springer (2003)
81. Serrano, F.: Intersection cuts for factorable MINLP. In: A. Lodi, V. Nagarajan (eds.) Integer Programming and Combinatorial Optimization, pp. 385–398. Springer International Publishing, Cham (2019)
82. Shamaiah, M., Banerjee, S., Vikalo, H.: Greedy sensor selection: Leveraging submodularity. In: 49th IEEE conference on decision and control (CDC), pp. 2572–2577. IEEE (2010)
83. Shi, X., Prokopyev, O.A., Zeng, B.: Sequence independent lifting for a set of submodular maximization problems. Mathematical Programming pp. 1–46 (2022)
84. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Mathematical programming **103**(2), 225–249 (2005)
85. Topkis, D.M.: Supermodularity and complementarity. Princeton university press, Princeton (2011)
86. Towle, E., Luedtke, J.: Intersection disjunctions for reverse convex sets. Mathematics of Operations Research **47**(1), 297–319 (2022)
87. Tuy, H.: Concave programming under linear constraints. Soviet Mathematics **5**, 1437–1440 (1964)
88. Vigerske, S.: MINLPLib: A Library of Mixed-Integer and Continuous Nonlinear Programming Instances (2022). URL `https://www.minlplib.org/`. Online accessed
89. Wiegele, A.: Biq mac library—a collection of max-cut and quadratic 0-1 programming instances of medium size. Preprint **51** (2007)
90. Xu, L., D'Ambrosio, C., Liberti, L., Vanier, S.H.: On cutting planes for signomial programming (2022). DOI 10.48550/ARXIV.2212.02857
91. Yu, Q., Küçükyavuz, S.: Strong valid inequalities for a class of concave submodular minimization problems under cardinality constraints. Mathematical Programming pp. 1–59 (2023)