Why not both?
Exact continuous and discrete optimization with submodularity

Jonathan Bunton and Paulo Tabuada

Abstract— **Submodularity is a pivotal property of functions defined on lattices, as it permits their exact minimization and approximate maximization in polynomial time. In this work, we examine submodular function minimization over continuous and discrete lattices simultaneously. We identify a class of these submodular optimization problems which can be solved exactly by applying a combination of submodular and convex optimization. The utility of this approach is demonstrated via several examples in the proposed class of optimization problems.**

I. Introduction

Optimal decision-making generally consists of two components: selecting some optimal subset of available assets or actions and then determining the best way to use them. Well-informed decisions inherently reason about these two components *simultaneously*. This observation motivates the study of optimization problems which give equal consideration to subset selection and the usage of available assets or actions.

In problems of optimal subset selection, submodularity is ubiquitous, as set-functions exhibiting submodularity can be exactly minimized and approximately maximized in strongly polynomial time [1], [2]. Moreover, submodularity has proven to be a useful model for a variety of phenomena, leading to efficient solutions to problems in countless areas such as summarization [3], [4], dictionary learning [5], [6], sensor placement [7], [8], and even providing guarantees for non-submodular optimization [9]–[11].

While generally presented as a property of set-functions, submodularity extends to general lattice structures, and many of the associated optimization guarantees extend to these general lattices as well [12], [13]. Continuous domains can also be endowed with a lattice structure, and recent work has used submodularity on these lattices to identify new classes of tractable optimization problems. To this end, efficient algorithms for minimization [14] and approximate maximization [15], of continuous submodular functions have been established. Moreover, this class of optimization problems is neither included in nor includes traditional convex problems.

Optimization problems over continuous and discrete lattices have been previously considered in the field of *structured sparsity*, where discrete selections are encoded in the form of a regularization penalty [16]. Motivated by this approach, we examine problems of the form:

$$\underset{\mathbf{x}\in\mathcal{L}}{\text{minimize}}\ f(\mathbf{x}) + g(\eta(\mathbf{x})), \tag{1}$$

where $(\mathcal{L}, \leq)$ is a continuous lattice capturing desired use of assets, and $\eta : \mathcal{L} \to \mathcal{M}$ maps elements in the continous lattice $\mathcal{L}$ to their corresponding value on the discrete lattice $(\mathcal{M}, \sqsubseteq)$. This class of problems includes several well-known examples such as compressed sensing problems:

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}}\ \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \|\mathbf{x}\|_0,$$

that are obtained from (1) by letting $f(\mathbf{x})$ be $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$, g be the function returning the cardinality of a set, and η to be a function mapping $\mathbf{x} \in \mathbb{R}^n$ to the set of its nonzero elements. In Section II, we discuss in more detail how to recover compressed sensing and other optimization problems from (1).

Traditional approaches to solving these problems often rely on minimizing convex surrogates, whose minimizers do not necessarily correspond to the minimizer of the initial problem [16]–[18]. More recent approaches use algorithms which take advantage of continuous submodularity, but these are either suboptimal [9] or require discretizing the solution space, necessarily introducing discretization error [14].

In this paper, we leverage submodularity and results from comparative statics to provide conditions under which this joint continuous-discrete problem may be instead solved exactly. This result is then used to identify a class of optimization problems which can be solved exactly in polynomial time by an agnostic combination of submodular function minimization and convex optimization routines.

We validate our theory in several case studies with joint convex and submodular optimization, showing similar running times to current state-of-the art. Unlike existing algorithms however, our approach produces guaranteed optimal solutions in every scenario, with no discretization error or convergence issues.

The paper is organized as follows: in Section II the relevant background is detailed and the formal problem statement is presented. Next, Section III introduces a parameterized form the problem and uses results from comparative statics to link its solution to the original problem formulation. Section IV specifically considers the case when the two lattices are continuous and Boolean, and the resulting computational evaluations are shown in Section V. Finally, we present some concluding remarks and future directions in Section VI.

This research was supported in part the US Army Research Laboratory (ARL) under Cooperative Agreement W911NF-17-2-0196.

Jonathan Bunton and Paulo Tabuada are with the UCLA Electrical and Computer Engineering Department, Los Angeles, CA 90095 {j.bunton, tabuada} @ucla.edu

II. Problem setup and basic definitions

To motivate submodularity on more general structures than the subsets of a finite set, we briefly review some key concepts from lattice theory.

Consider the set $\mathcal{L}$ equipped with a partial ordering of its elements $\leq$, written together as $(\mathcal{L}, \leq)$. For brevity, we denote the partially ordered set by simply $\mathcal{L}$ when the ordering is clear from context. For any $\mathbf{x}, \mathbf{x}' \in \mathcal{L}$, we define the least upper bound (or *join*) of these elements under the partial ordering as $\mathbf{x} \vee \mathbf{x}'$. We similarly define the greatest lower bound (or *meet*) as $\mathbf{x} \wedge \mathbf{x}'$. When every $\mathbf{x}, \mathbf{x}' \in \mathcal{L}$ has a unique join and meet which is also in $\mathcal{L}$, the set and ordering define a *lattice*. We can similarly define a *sublattice* as a subset $S \subseteq \mathcal{L}$ such that for every $\mathbf{x}, \mathbf{x}' \in S$, their join, $\mathbf{x} \vee \mathbf{x}'$, and their meet, $\mathbf{x} \wedge \mathbf{x}'$, are also in S.

A function $f : \mathcal{L} \to \mathbb{R}$ is *submodular* on the lattice $\mathcal{L}$ if for all $\mathbf{x}, \mathbf{x}' \in \mathcal{L}$:

$$f(\mathbf{x}) + f(\mathbf{x}') \geq f(\mathbf{x} \vee \mathbf{x}') + f(\mathbf{x} \wedge \mathbf{x}'). \tag{2}$$

Additionally, f is *monotone* if for all $\mathbf{x}, \mathbf{x}' \in \mathcal{L}$:

$$\mathbf{x} \leq \mathbf{x}' \implies f(\mathbf{x}) \leq f(\mathbf{x}'). \tag{3}$$

As an example, consider the power set of some finite set K, denoted by 2^K. The typical partial order endowed on 2^K is set inclusion, $\subseteq$, with the corresponding join and meet operations being set union and intersection, respectively. We then obtain the traditional definition of submodularity for set-functions, where $f : 2^K \to \mathbb{R}$ is submodular if for any $A, B \in 2^K$:

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B), \tag{4}$$

and is monotone if for any $A, B \in 2^K$:

$$A \subseteq B \implies f(A) \leq f(B). \tag{5}$$

Submodularity with respect to general lattices (2) also includes continuous domains with lattice structure. Lattice structure on these continuous domains simply means a partial ordering such that the join and meet are uniquely defined and in the domain. A common choice is to endow $\mathbb{R}^n$ with coordinate-wise ordering $\leq$, where for any $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$:

$$\mathbf{x} \leq \mathbf{x}' \Leftrightarrow \mathbf{x}_i \leq \mathbf{x}'_i, \ i = 1, 2, ..., n. \tag{6}$$

This ordering has join and meet operations defined by coordinate-wise minimum and maximum respectively, meaning for any $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$:

$$\begin{matrix} \left(\mathbf{x} \vee \mathbf{x}'\right)_i = \max\left(\mathbf{x}_i, \mathbf{x}'_i\right) \\ \left(\mathbf{x} \wedge \mathbf{x}'\right)_i = \min\left(\mathbf{x}_i, \mathbf{x}'_i\right) \end{matrix} \quad \text{for all } i = 1, 2, ..., n. \tag{7}$$

As in the subset lattice case, functions which are submodular on $\mathbb{R}^n$ also give rise to polynomial-time algorithms for exact minimization [14] and approximate maximization [15].

We now formalize the problem studied in this paper.

Problem 1: Let $\mathcal{L}$ and $\mathcal{M}$ be lattices. Consider the maps $f : \mathcal{L} \to \mathbb{R}$ and $g : \mathcal{M} \to \mathbb{R}$, and let $\eta : \mathcal{L} \to \mathcal{M}$ be a map between the two lattices. We seek a minimizer $\mathbf{x} \in \mathcal{L}$ for the problem:

$$\underset{\mathbf{x} \in \mathcal{L}}{\text{minimize}} \ f(\mathbf{x}) + g(\eta(\mathbf{x})). \tag{P}$$

More plainly, our objective is to minimize the function f over one lattice $\mathcal{L}$, while considering costs incurred on a second lattice $\mathcal{M}$ through the mapping η and the function g.

As an example, let the first lattice be $(\mathbb{R}^n, \leq)$, and let $f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$ with $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$. Let the second lattice be $(2^n, \subseteq)$ and define $\eta = supp : \mathbb{R}^n \to 2^n$ as:

$$supp\,(\mathbf{x}) = \{i \in \{1, 2, ..., n\} \mid \mathbf{x}_i \neq 0\}, \tag{8}$$

which maps a real-valued vector $\mathbf{x} \in \mathbb{R}^n$ to the set of its nonzero entries. Letting g be the cardinality function $|supp\,(\mathbf{x})|$, we have $g(supp\,(\mathbf{x})) = \|\mathbf{x}\|_0$, the ℓ_0 pseudo-norm counting the number of nonzero entries. The lattice optimization problem (P) becomes:

$$\underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} \ \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \|\mathbf{x}\|_0, \tag{CS}$$

which is a form of the well-studied compressed sensing problem.

The compressed sensing problem (CS) is known to be NP-Hard [18], but ties to submodularity on the lattice $(\mathbb{R}^n, \leq)$ can provide guarantees for simple algorithms to produce optimal solutions [9], [10]. In contrast, our work identifies conditions under which we may solve (CS) exactly through submodular function minimization over the power set lattice.

In the following, we let $\mathcal{L}$ and $\mathcal{M}$ be lattices, and denote their partial orderings by $\leq$ and $\sqsubseteq$, respectively. We write the join and meet operations for $\mathcal{L}$ as $\vee$ and $\wedge$, and similarly denote the join and meet for $\mathcal{M}$ as $\sqcup$ and $\sqcap$.

In this paper, we make two assumptions on the functions f and g, and one on the mapping η defining problem (P).

Assumptions:

1) f and g are submodular on the lattices $\mathcal{L}$ and $\mathcal{M}$, respectively;
2) g is monotone;
3) for all $\mathbf{x}, \mathbf{x}' \in \mathcal{L}$:

$$\eta(\mathbf{x} \vee \mathbf{x}') \sqsubseteq \eta(\mathbf{x}) \sqcup \eta(\mathbf{x}'),$$
$$\eta(\mathbf{x} \wedge \mathbf{x}') \sqsubseteq \eta(\mathbf{x}) \sqcap \eta(\mathbf{x}').$$

We next use these assumptions to prove the correctness of a parameterization-based approach to exactly solve problem (P).

III. Optimization on two lattices

In this section, we propose a parameterization of problem (P) to define a new function over $\mathcal{M}$. We show this new function is submodular on the lattice $\mathcal{M}$, and therefore can be minimized exactly in polynomial time. Finally, we show the minimizer of this parameterized problem may be used to compute a minimizer of the original problem.

A. Parameterization

We first identify an equivalent form of problem (P):

$$\underset{\mathbf{y}\in\mathcal{M}}{\text{minimize}} \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}}} f(\mathbf{x}) + g(\mathbf{y}). \tag{P1}$$

In this form, we cast the problem as an outer minimization over $\mathcal{M}$ and an inner minimization over all $\mathbf{x} \in \mathcal{L}$ satisfying $\eta(\mathbf{x}) = \mathbf{y}$, as opposed to all $\mathbf{x} \in \mathcal{L}$.

While the parameterization (P1) is equivalent to (P), the inner minimization still involves searching over all $\mathbf{x} \in \mathcal{L}$ such that $\eta(\mathbf{x}) = \mathbf{y}$. Instead, we consider searching over a larger space and define the function:

$$H(\mathbf{y}) = \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})\sqsubseteq\mathbf{y}}} f(\mathbf{x}) + g(\mathbf{y}). \tag{9}$$

We consider this relaxation because the constraint $\eta(\mathbf{x}) \leq \mathbf{y}$ makes for more tractable minimization in (9) than $\eta(\mathbf{x}) = \mathbf{y}$. For example, in problem (CS), for an $A \in 2^n$ the relaxed constraint in (9) becomes $supp(\mathbf{x}) \subseteq A$, an affine equality constraint forcing some entries to be zero and *allowing* others to be nonzero, as opposed to the stricter $supp(\mathbf{x}) = A$, which sets some entries to be zero and *forces* others to be nonzero.

We propose solving (P) by instead solving the surrogate problem:

$$\underset{\mathbf{y}\in\mathcal{M}}{\text{minimize}}\ H(\mathbf{y}). \tag{P2}$$

In the following, we show conditions under which solving the parameterized problem (P2) is tractable, then connect its minimizer to the minimizer of our original problem (P).

B. Tractable parameterized optimization

Problem (P2) is only useful when H can be efficiently minimized. We show this to be the case by first proving that H is submodular on $\mathcal{M}$.

Lemma 3.1: Consider the set:

$$S = \{(\mathbf{x},\mathbf{y}) \in \mathcal{L}\times\mathcal{M} \mid \eta(\mathbf{x}) \sqsubseteq \mathbf{y}\}, \tag{10}$$

and the mapping $\eta : \mathcal{L} \to \mathcal{M}$. If η satisfies assumption (3), then S is a sublattice of $\mathcal{L}\times\mathcal{M}$.

Proof: Take $(\mathbf{x},\mathbf{y}) \in S$ and $(\mathbf{x}',\mathbf{y}') \in S$. The join operation on S is:

$$(\mathbf{x},\mathbf{y}) \vee (\mathbf{x}',\mathbf{y}') = (\mathbf{x}\vee\mathbf{x}', \mathbf{y}\sqcup\mathbf{y}').$$

We then simply verify that the join of two elements is still in S:

$$\eta(\mathbf{x}\vee\mathbf{x}') \sqsubseteq \eta(\mathbf{x}) \sqcup \eta(\mathbf{x}') \sqsubseteq \mathbf{y}\sqcup\mathbf{y}'$$
$$\Rightarrow (\mathbf{x}\vee\mathbf{x}', \mathbf{y}\sqcup\mathbf{y}') \in S,$$

where in this step we have used assumption (3) and that $\mathbf{x},\mathbf{x}' \in S$. An identical proof holds for $(\mathbf{x},\mathbf{y}) \wedge (\mathbf{x}',\mathbf{y}')$ defined by:

$$(\mathbf{x},\mathbf{y}) \wedge (\mathbf{x}',\mathbf{y}') = (\mathbf{x}\wedge\mathbf{x}', \mathbf{y}\sqcap\mathbf{y}').$$

■

We next note that the objective function in problem (P) is submodular on this sublattice S.

Lemma 3.2: Consider the functions $f : \mathcal{L} \to \mathbb{R}$ and $g : \mathcal{M} \to \mathbb{R}$. If f is submodular on $\mathcal{L}$ and g is submodular on $\mathcal{M}$, then the function $f + g : \mathcal{L}\times\mathcal{M} \to \mathbb{R}$ defined by:

$$(\mathbf{x},\mathbf{y}) \mapsto f(\mathbf{x}) + g(\mathbf{y}),$$

restricted to the sublattice $S \subseteq \mathcal{L}\times\mathcal{M}$, as defined in (10), is submodular.

Proof: Take $(\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}') \in S$. It follows:

$$\begin{aligned} &f(\mathbf{x}) + g(\mathbf{y}) + f(\mathbf{x}') + g(\mathbf{y}') \\ &\quad \geq f(\mathbf{x}\vee\mathbf{x}') + g(\mathbf{y}\sqcup\mathbf{y}') + f(\mathbf{x}\wedge\mathbf{x}') + g(\mathbf{y}\sqcap\mathbf{y}') \end{aligned}$$

where in the second line we have used the submodularity of f and g and that S is a sublattice and therefore the meet and join elements are still in S. ■

We now identify a result from [19] which connects our previous two lemmas to the submodularity of H. We provide its proof here for completeness.

Theorem 3.3: (From [19].) Consider the functions $f : \mathcal{L} \to \mathbb{R}$ and $g : \mathcal{M} \to \mathbb{R}$, and the mapping $\eta : \mathcal{L} \to \mathcal{M}$. If f and g are submodular and η satisfies asssumption (3), then the function H defined by the parameterized optimization problem (9) is submodular on $\mathcal{M}$.

Proof: Take $(\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}') \in S$, such that:

$$\mathbf{x} = \underset{\substack{\mathbf{z}\in\mathcal{L} \\ \eta(\mathbf{z})\sqsubseteq\mathbf{y}}}{\operatorname{argmin}}\ f(\mathbf{z}) + g(\mathbf{y}),$$
$$\mathbf{x}' = \underset{\substack{\mathbf{z}\in\mathcal{L} \\ \eta(\mathbf{z})\sqsubseteq\mathbf{y}'}}{\operatorname{argmin}}\ f(\mathbf{z}) + g(\mathbf{y}').$$

Because S is a sublattice (Lemma 3.1), $(\mathbf{x},\mathbf{y}) \vee (\mathbf{x}',\mathbf{y}')$ and $(\mathbf{x},\mathbf{y}) \wedge (\mathbf{x}',\mathbf{y}')$ are also in S. It follows:

$$\begin{aligned} H(\mathbf{y}) + H(\mathbf{y}') &= f(\mathbf{x}) + g(\mathbf{y}) + f(\mathbf{x}') + g(\mathbf{y}') \\ &\geq f(\mathbf{x}\vee\mathbf{x}') + g(\mathbf{y}\sqcup\mathbf{y}') + f(\mathbf{x}\wedge\mathbf{x}') + g(\mathbf{y}\sqcap\mathbf{y}') \\ &\geq \min_{\substack{\mathbf{u}\in\mathcal{L} \\ \eta(\mathbf{u})\sqsubseteq\mathbf{y}\sqcup\mathbf{y}'}} f(\mathbf{u}) + g(\mathbf{y}\sqcup\mathbf{y}') \\ &\qquad + \min_{\substack{\mathbf{v}\in\mathcal{L} \\ \eta(\mathbf{v})\sqsubseteq\mathbf{y}\sqcap\mathbf{y}'}} f(\mathbf{v}) + g(\mathbf{y}\sqcap\mathbf{y}') \\ &= H(\mathbf{y}\sqcup\mathbf{y}') + H(\mathbf{y}\sqcap\mathbf{y}'). \end{aligned}$$

In the second line we used the submodularity of $f + g$ on S (Lemma 3.2). Because S is a sublattice, the pairs $(\mathbf{x}\vee\mathbf{x}', \mathbf{y}\sqcup\mathbf{y}')$ and $(\mathbf{x}\wedge\mathbf{x}', \mathbf{y}\sqcap\mathbf{y}')$ are also in S, and therefore are feasible points in the minimization performed in the third line, producing the submodular inequality. ■

The submodularity of H on $\mathcal{M}$ implies H may be efficiently minimized, and even approximately maximized. We then seek a connection between this ability and a solution to our initial optimization problem.

In the following theorem we compare solutions of the problem (P1) to the solutions of relaxed problem (P2) and show that under assumptions (1)-(3), we may construct minimizers of (P1) and (P) from a minimizer of (P2). The essence of our proof lies in recognizing that (P) and (P2) differ only when the inequality constraint is not tight. We then use the monotonicity of g to show that when this difference occurs we can still easily construct another minimizer for

(P2) such that the inequality is tight and the problems are again equivalent.

Theorem 3.4: Consider the functions $f : \mathcal{L} \to \mathbb{R}$, $g : \mathcal{M} \to \mathbb{R}$, and $\eta : \mathcal{L} \to \mathcal{M}$. Let f, g, and η satisfy assumptions (1)-(3). If $\mathbf{y}_2^*$ is a minimizer for the relaxed problem (P2) and:

$$\mathbf{x}_2^* \in \underset{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})\sqsubseteq\mathbf{y}_2^*}}{\operatorname{argmin}}\ f(\mathbf{x}) + g\left(\mathbf{y}_2^*\right),$$

then $\eta\left(\mathbf{x}_2^*\right)$ is a minimizer for (P1), and $\mathbf{x}_2^*$ is a minimizer for (P).

Proof: Let $\mathbf{y}_1^*$ and $\mathbf{y}_2^*$ be such that:

$$\mathbf{y}_1^* \in \underset{\mathbf{y}\in\mathcal{M}}{\operatorname{argmin}} \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}}} f(\mathbf{x}) + g(\mathbf{y}), \tag{11}$$

$$\mathbf{y}_2^* \in \underset{\mathbf{y}\in\mathcal{M}}{\operatorname{argmin}}\ H(\mathbf{y}), \tag{12}$$

i.e., $\mathbf{y}_1^*$ is a minimizer for (P1) and $\mathbf{y}_2^*$ is a minimizer for (P2). By (12), we have:

$$\begin{aligned} H(\mathbf{y}_2^*) &\le H(\mathbf{y}_1^*) \\ &= \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})\sqsubseteq\mathbf{y}_1^*}} f(\mathbf{x}) + g\left(\mathbf{y}_1^*\right) \\ &\le \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}_1^*}} f(\mathbf{x}) + g\left(\mathbf{y}_1^*\right), \end{aligned} \tag{13}$$

where in the third line we shrank the feasible set for the minimization. Moreover, noting (11):

$$\min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}_1^*}} f(\mathbf{x}) + g\left(\mathbf{y}_1^*\right) \le \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}_2^*}} f(\mathbf{x}) + g\left(\mathbf{y}_2^*\right). \tag{14}$$

Recall that $\mathbf{x}_2^*$ is defined as:

$$\mathbf{x}_2^* \in \underset{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})\sqsubseteq\mathbf{y}_2^*}}{\operatorname{argmin}}\ f(\mathbf{x}) + g\left(\mathbf{y}_2^*\right),$$

and identify two cases:

Case 1: $\eta\left(\mathbf{x}_2^*\right) = \mathbf{y}_2^*$.

It then holds that:

$$\begin{aligned} H(\mathbf{y}_2^*) &= \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})\sqsubseteq\mathbf{y}_2^*}} f(\mathbf{x}) + g\left(\mathbf{y}_2^*\right) \\ &= \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}_2^*}} f(\mathbf{x}) + g\left(\mathbf{y}_2^*\right). \end{aligned} \tag{15}$$

By combining (15) and (13) we have:

$$\min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}_2^*}} f(\mathbf{x}) + g\left(\mathbf{y}_2^*\right) \le \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}_1^*}} f(\mathbf{x}) + g\left(\mathbf{y}_1^*\right).$$

Noting (14), the left hand side also upper bounds the right hand side, and hence we have equality:

$$\min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}_1^*}} f(\mathbf{x}) + g\left(\mathbf{y}_1^*\right) = \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})=\mathbf{y}_2^*}} f(\mathbf{x}) + g\left(\mathbf{y}_2^*\right).$$

Therefore $\mathbf{y}_2^*$ has the same objective value for (P1) and is also minimizer. Clearly then, $\mathbf{x}_2^*$ is also a minimizer for (P).

Case 2: $\eta\left(\mathbf{x}_2^*\right) = \mathbf{y}' <_2 \mathbf{y}_2^*$.

By the monotonicity of g, we have:

$$g\left(\mathbf{y}'\right) \le g\left(\mathbf{y}_2^*\right),$$

and therefore:

$$\begin{aligned} H(\mathbf{y}_2^*) &= f(\mathbf{x}_2^*) + g(\mathbf{y}_2^*) \ge f(\mathbf{x}_2^*) + g(\mathbf{y}') \\ &\ge \min_{\substack{\mathbf{x}\in\mathcal{L} \\ \eta(\mathbf{x})\sqsubseteq\mathbf{y}'}} f(\mathbf{x}) + g(\mathbf{y}') = H(\mathbf{y}'). \end{aligned} \tag{16}$$

But by (12), we also have:

$$H(\mathbf{y}_2^*) \le H(\mathbf{y}'). \tag{17}$$

By (16) and (17), it holds that $H(\mathbf{y}_2^*) = H(\mathbf{y}')$ and $\mathbf{y}'$ is also a minimizer for (P2).

For the minimizer $\mathbf{y}'$, $\eta\left(\mathbf{x}_2^*\right) = \mathbf{y}'$ and Case 1 holds, indicating $\mathbf{y}'$ is a minimizer of (P1) and $\mathbf{x}_2^*$ is a minimizer of (P). ■

We note that the proofs in this section follow identically for constrained optimization, provided the constraint set $\mathcal{C} \subseteq \mathcal{L}$ is a sublattice.

Theorems 3.3 and 3.4 together give conditions for flexibility in selecting which lattice to optimize over (i.e. $\mathcal{M}$ or $\mathcal{L}$) to solve the same problem, provided the function H is easy to evaluate.

IV. The case of $\mathbb{R}^n$ and 2^n

In this section, we briefly discuss submodularity in the context of continuous lattices. We then use Theorems 3.3 and 3.4 to identify a class of optimization problems defined on $\mathbb{R}^n$ and the power set lattice 2^n that can be exactly solved in polynomial time.

A. Submodularity on $\mathbb{R}^n$

We recall the continuous lattice $\mathbb{R}^n$ with element-wise ordering $\le$ as expressed in (6), resulting in the element-wise minimum and maximum as the join and meet operations expressed in (7). It is then natural to consider functions which are submodular on $(\mathbb{R}^n, \le)$, as done in [14], [15].

As shown in [14], [19] for twice-differentiable functions $f : \mathbb{R}^n \to \mathbb{R}$, submodularity on $(\mathbb{R}^n, \le)$ is equivalent to the condition:

$$\frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j} \le 0, \text{ for all } i \ne j,$$

for all $\mathbf{x} \in \mathbb{R}^n$. The class of submodular functions on $(\mathbb{R}^n, \le)$ is neither a subset nor a superset of convex functions, despite both classes of functions permitting polynomial time minimization [14].

B. Tractable evaluations

A key factor in the efficient minimization of H in (P2) is the ability to tractably evaluate H. Evaluating H is itself an optimization problem over a sublattice of $\mathcal{L} \times \mathcal{M}$. In the case of $(\mathbb{R}^n_{\ge 0}, \le)$ and $(2^n, \subseteq)$ letting η be $supp$, this corresponds to solving, for an arbitrary $I \in (2^n, \subseteq)$:

$$\begin{aligned} &\underset{\mathbf{x}\in\mathbb{R}^n_{\ge 0}}{\text{minimize}}\ f(\mathbf{x}) + g(I) \\ &\text{subject to}\ \ supp(\mathbf{x}) \subseteq I, \end{aligned} \tag{18}$$

which is an optimization problem subject to the constraint that the entries $\mathbf{x}_i$ with $i \in I$ be zero. We can equivalently express this notion as the affine equality constraint:

$$supp(\mathbf{x}) \subseteq I \iff \mathbf{x}_i = 0 \text{ for all } i \notin I.$$

Rather than using generic lattice optimization algorithms to solve (18), we can instead seek functions $f : \mathbb{R}^n \to \mathbb{R}$ which are both submodular, and permit efficient minimization subject to an affine equality constraint.

A simple choice is to require the function f to be both submodular and convex, as convex functions allow polynomial time minimization subject to affine equality constraints. Under this requirement, H may be evaluated in polynomial time by solving a convex optimization problem.

C. Polynomial Time Joint Optimization

We formalize the advantage in requiring f to be convex in the following corollary.

Corollary 4.1: Let $f : \mathbb{R}^n_{\geq 0} \to \mathbb{R}$ be a convex and submodular function on $(\mathbb{R}^n_{\geq 0}, \leq)$ and $g : 2^n \to \mathbb{R}$ be a monotone submodular set function. Further, let $supp : \mathbb{R}^n_{\geq 0} \to 2^n$ be the support map as defined in (8). Then the optimization problem:

$$\underset{\mathbf{x} \in \mathbb{R}^n_{\geq 0}}{\text{minimize}} \; f(\mathbf{x}) + g(supp(\mathbf{x})), \tag{19}$$

can be solved in polynomial time.

Proof: The function $supp : \mathbb{R}^n_{\geq 0} \to 2^n$ satisfies Assumption (3) as for any $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n_{\geq 0}$:

$$\begin{aligned} supp\left(\mathbf{x} \vee \mathbf{x}'\right) &= supp(\mathbf{x}) \cup supp\left(\mathbf{x}'\right) \\ supp\left(\mathbf{x} \wedge \mathbf{x}'\right) &= supp(\mathbf{x}) \cap supp\left(\mathbf{x}'\right). \end{aligned}$$

Then, by Theorem 3.3, the function

$$H(A) = \min_{\substack{\mathbf{x} \in \mathbb{R}^n_{\geq 0} \\ supp(\mathbf{x}) \subseteq A}} f(\mathbf{x}) + g(A) \tag{20}$$

is submodular on 2^n. Moreover, its evaluation is a convex minimization problem subject to an affine equality constraint, and therefore is a polynomial time operation. By its submodularity, we can also minimize H in time polynomial in both evaluations and dimension n. Finally, by Theorem 3.4, this produces a minimizer of (19). ■

Given arbitrary functions f and g, a naive method to solve (19) exactly would be to exhaustively search through all possible sets of nonzero entries. In doing this, we select a set of nonzero entries $A \in 2^n$, identify the minimizing argument with these entries nonzero, continuing for all sets to find the minimum. Clearly this exhaustive search quickly becomes intractable. In light of Corollary 4.1, however, any combination of convex optimization and submodular optimization routines can *efficiently* search through all possible sets of nonzero entries in strongly polynomial time, with guaranteed optimality.

Our proposed method does not algorithmically depend on the continuous lattice ordering for $\mathbb{R}^n$, as we instead resort to convex optimization tools to evaluate the minimization in (20). We simply require that there *exists* a partial ordering on $\mathbb{R}^n$ such that f is submodular, satisfying Assumption (1), and that the mapping $supp : \mathbb{R}^n \to 2^n$ satisfies Assumption (3). As long as such an ordering exists, the parameterization method is still guaranteed to produce an optimal solution.

V. Illustrative Examples and Computational Evaluation

In this section we present a few numerical examples for the lattices $(\mathbb{R}^n, \leq)$ and $(2^n, \subseteq)$, and validate our proposed method. We compare our approach to two others, namely the Pairwise-Franke-Wolfe algorithm coupled with discretization to solve the continuous submodular problem directly (denoted *Cont Submodular* and plotted with blue in figures) [14], and a generic projected subgradient descent method (denoted *Projected (Sub)Gradient* and plotted with red in figures) to minimize H which is shown in [10] to possess optimality guarantees even in the non-submodular case.

For our approach, we use the minimum-norm point algorithm as implemented by [20], coupled with IBM's CPLEX 12.8 quadratic program solver in MATLAB. We run the minimum-norm point algorithm until convergence (certified optimality gap less than 10^{-4}), and allow the continuous submodular and projected gradient descent algorithms to run until either similar convergence or a maximum of 1000 iterations. Each algorithm is presented with the same cost function to minimize, meaning any variations in objective value are results of purely algorithmic differences. All experiments were run on a mid-2012 Macbook Pro equipped with 2.9 GHz Intel Core i7 and 16GB 1600 MHz DDR3 RAM.

A. Regularized Least-Squares

Similarly to (CS), we consider a regularized least-squares problem. We let $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$, $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{y} \in \mathbb{R}^m$, with $g : 2^n \to \mathbb{R}$ a monotone submodular set function. We set $\lambda \in \mathbb{R}_{\geq 0}$, and consider the problem:

$$\underset{\mathbf{x} \in \mathbb{R}^n_{\geq 0}}{\text{minimize}} \; \frac{1}{2} \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda g(supp(\mathbf{x})). \tag{LS}$$

On $(\mathbb{R}^n, \leq)$, submodularity requires that $\left(\mathbf{A}^T \mathbf{A}\right)_{ij} \leq 0$, for all $i \neq j$. We generate synthetic problem instances with $m = n$ by generating $\mathbf{A}^T \mathbf{A} \in \mathbb{R}^{n \times n}$ with entries drawn uniformly from the interval $[-1, 0]$. We then add n times the identity matrix, making this matrix positive semidefinite (and diagonally dominant). The resulting $\mathbf{A} \in \mathbb{R}^{n \times n}$ may then be extracted by the Cholesky decomposition. For the data points $\mathbf{y} \in \mathbb{R}^n$, we use the signal shown in Fig. 1a.

As an example submodular set function, we consider the modified range function augmented with cardinality:

$$g(S) = \begin{cases} n + \max(S) - \min(S) - 1 + |S|, & S \neq \varnothing \\ 0, & S = \varnothing, \end{cases}$$

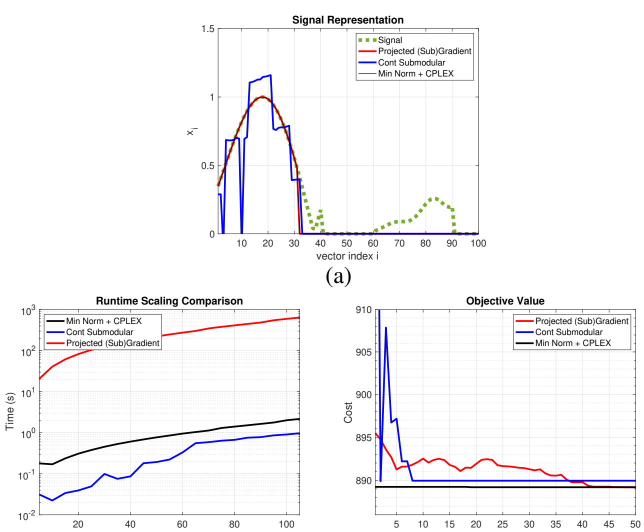
where $\min(S)$ $(\max(S))$ is the minimum (maximum) index element in S. This particular function assigns a high penalty for large sets of very wide-spread index selections, while giving a small penalty to sets containing few elements which are nearby in index. For experiments, we set the regularization strength to $\lambda = 0.05$ so this set function plays a nontrivial role in the overall cost function. For the continuous submodular approach, we divide the solution space of $\mathbf{x} \in \mathbb{R}^n$ (taken to be $[-1, 1]^n$) into a uniform grid of 51 discrete points per axis.

A comparison of runtime scaling for each approach is shown in Fig. 1b, while the sequence of achieved values of the cost in (LS) for a problem instance with $n = 100$ is shown in Fig. 1c. As the solution to (LS) is a representation of the signal $\mathbf{y}$ with structured sparse columns of $\mathbf{A}$, we also show the resulting signal representations generated from each algorithm in Fig. 1a. Analysis of these results is covered in Section V-C.

B. Signal Denoising

Consider a discrete time signal $\mathbf{x} \in \mathbb{R}^n$ which is assumed to have arrived in some discrete window of length $p \leq n$. We know this signal to be smooth, meaning variations between adjacent time indices are relatively small. Moreover, we know that the signal occurred in a small number of consecutive time steps within the recorded measurements. However, this signal is corrupted by noise so that the received signal is instead $\mathbf{y} = \mathbf{x} + \mathbf{w}$, with $\mathbf{w} \in \mathbb{R}^n$, $\mathbf{w} \sim \mathcal{N}(0, \mathbf{I})$.

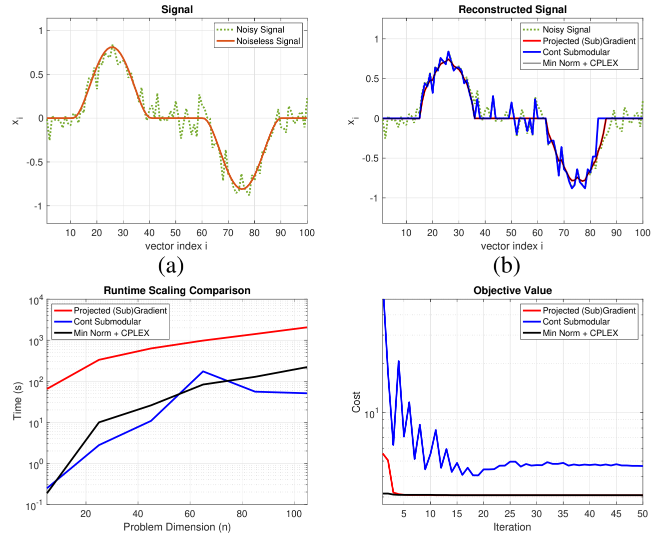
We can encode our prior knowledge of a smooth signal in the continuous, convex, and submodular function $f(\mathbf{x}) = \sum_{i=1}^{n-1} (\mathbf{x}_i - \mathbf{x}_{i+1})^2$. This function promotes smoothness by penalizing the variation of the signal in adjacent time steps with a quadratic function. We can also consider the monotone submodular function $g(S) = |S| + \#Int(S)$,



(a)

(b) (c)

Fig. 1: Results from the least-squares (LS) simulations. Each algorithm's reconstructed signal representations using structured sparse columns of $\mathbf{A}$ are shown in (a). Note the solution produced by projected subgradient and the minimum-norm point algorithm coincide with the signal for a large segment. A small window of runtime scaling comparisons is shown in (a), while (c) compares objective function values as the algorithms iterate.



(a) (b)

(c) (d)

Fig. 2: Results from the smooth signal reconstruction problem. The true signal and its noisy counterpart are shown in (a), while (b) shows the reconstruction performed by the algorithms. For a small window of problem sizes, (c) shows the running times, and (d) shows the objective value of the solution at each iteration for $n = 100$.

where $\#Int(S)$ counts the number of sets of adjacent nonzero elements in S. This set function assigns a high penalty to vectors $\mathbf{x} \in \mathbb{R}^n$ with many disjoint segments of nonzero entries.

The corresponding optimization problem is then:

$$\begin{aligned} \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2 + \mu \sum_{i=1}^{n-1} (\mathbf{x}_i - \mathbf{x}_{i+1})^2 \\ & + \lambda\left(|supp(\mathbf{x})| + \#Int(supp(\mathbf{x}))\right). \end{aligned} \tag{DN}$$

We generate the signal $\mathbf{y} \in \mathbb{R}^n$ shown in Figs. 2a and 2b and let $\lambda = 0.05$ and $\mu = 0.8$, so the set function regularization plays a nontrivial role in the overall objective value. The noise vector w is drawn from iid normal distributions with zero mean and standard deviations 0.1. For the continuous submodular algorithm, we again represent the space of the decision variable $\mathbf{x} \in [-1, 1]^n$ as 51 discrete values. The problem dimension n is varied across a small range of values to capture runtime comparisons in Fig. 2c, and the objective value across iterations is examined in Fig. 2d. The implications of these results are discussed in the following section.

C. Discussion

The solutions produced by each algorithm vary somewhat in quality. The continuous submodular optimization approach can often reach near-optimality, but suffers a discretization error. This error is visually clear in Fig. 2b, but is also evident from the sizeable gap between the continuous approach's achieved costs and the optimal in Figs. 1c and 2d. Naturally, as the problem size grows larger, the discretization error also

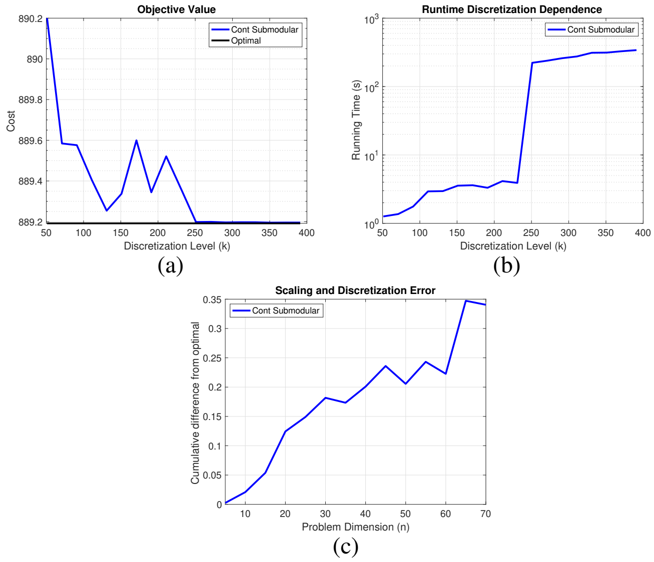


Fig. 3: The effects of the discretization level on the continuous submodular algorithm's optimality (a) and runtime (b) for problem (LS) with $n = 100$. The growth of discretization error with n for proble (DN) is shown in (c).

accumulates. We analyzed this behavior with instances of problem (DN), as shown in Fig. 3c.

This discretization error can be mitigated by using a finer resolution, but the running times increase accordingly. Fig 3 highlights this behavior in an instance of the problem (LS) with $n = 100$ by varying the discretization resolution from 50 to 400. Notably, once the discretization error becomes considerably small, the algorithm's running time jumps by an order of magnitude.

As projected subgradient descent on H comes with optimality guarantees, it is unsurprising that it reaches the optimal cost in both scenarios. Moreover, the effectiveness of projected subgradient descent illustrates the correctness of our theory, as this particular algorithm becomes effectively another choice of submodular function minimization routine in our framework.

The minimum-norm point approach converges to its guaranteed optimality in considerably fewer iterations than its competitors. Comparing Figs. 3b and 1b, the continuous submodular algorithm must discretize the space with much higher resolution to produce competitive solutions, resulting in *higher* running times than our approach.

Despite their rapid convergence, the projected gradient and minimum-norm point approaches involve costly iterations. In particular, each iteration of these algorithms involve computing the Lovàsz Extension of H, which has complexity $O(n \log n + nEO)$, where EO denotes the cost of evaluating H. However, each evaluation of H is a convex minimization problem. If, for example, solving this problem had complexity $EO = O(n^3)$ (typical for many interior-point methods), then each iteration would have complexity $O(n \log n + n^4)$. Nevertheless, our approach produces exact optimal solutions in time competitive with the current state-of-the-art.

The benefit to our approach, however, is its flexibility in the choice of convex optimization routine to evaluate H. Depending on the problem structure, specialized algorithms may solve the equality-constrained minimization problem more rapidly. Another natural thought would be to solve each convex minimization suboptimally, making for faster evaluations of H at the price of exactness. While projected subgradient descent still gives approximation guarantees in this case [10], there may exist problems for which an agnostic combination still produces optimal solutions.

VI. Conclusion

This work posed optimal decision making as an optimization problem on continuous and discrete lattice structures, showing that submodular functions on both lattices give rise to a submodular discrete parameterization. We then leveraged this result to identify a class of joint continuous and discrete optimization problems which can be solved exactly by minimizing a purely discrete parameterization.

We applied this theory to continuous and Boolean domains, showing how an agnostic combination of convex optimization and submodular function minimization routines can produce guaranteed optimal solutions to joint continuous and discrete optimization problems in polynomial time. This theory was then validated against current state-of-the-art algorithms for continuous submodular function optimization, and gave comparable performance with provable optimality of the produced solution.

Future work might examine extending this framework to more general constrained optimization problems than sublattices. Moreover, many convex functions may not be exactly submodular, and it would be useful to investigate conditions under which the proposed techniques can be extended to functions only approximately satisfying this requirement.

References

[1] S. Fujishige and S. Isotani, "A submodular function minimization algorithm based on the minimum-norm base", *Pacific Journal of Optimization*, vol. 7, no. 1, pp. 3–17, 2011.

[2] A. Schrijver, "A combinatorial algorithm minimizing submodular functions in strongly polynomial time", *Journal of Combinatorial Theory, Series B*, vol. 80, no. 2, pp. 346–355, 2000.

[3] A. Badanidiyuru, B. Mirzasoleiman, A. Karbasi, and A. Krause, "Streaming submodular maximization: Massive data summarization on the fly", in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 671–680.

[4] H. Lin and J. Bilmes, "A class of submodular functions for document summarization", in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT '11, Portland, Oregon: Association for Computational Linguistics, 2011, pp. 510–520, ISBN: 9781932432879.

[5] A. Krause and V. Cevher, "Submodular dictionary selection for sparse representation", in *International Conference on Machine Learning (ICML)*, 2010.

[6] F. R. Bach, "Structured sparsity-inducing norms through submodular functions", in *Advances in Neural Information Processing Systems*, 2010, pp. 118–126.

[7] A. Krause, J. Leskovec, C. Guestrin, J. VanBriesen, and C. Faloutsos, "Efficient sensor placement optimization for securing large water distribution networks", *Journal of Water Resources Planning and Management*, vol. 134, no. 6, pp. 516–526, 2008.

[8] A. Krause, R. Rajagopal, A. Gupta, and C. Guestrin, "Simultaneous optimization of sensor placements and balanced schedules", *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2390–2405, Oct. 2011, ISSN: 0018-9286. DOI: 10.1109/TAC.2011.2164010.

[9] A. Das and D. Kempe, "Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection", *arXiv preprint arXiv:1102.3975*, 2011.

[10] M. El Halabi and S. Jegelka, "Optimal approximation for unconstrained non-submodular minimization",

[11] E. R. Elenberg, R. Khanna, A. G. Dimakis, S. Negahban, *et al.*, "Restricted strong convexity implies weak submodularity", *The Annals of Statistics*, vol. 46, no. 6B, pp. 3539–3568, 2018.

[12] S. Fujishige, *Submodular functions and optimization*. Elsevier, 2005, vol. 58.

[13] A. Schrijver, *Combinatorial optimization: polyhedra and efficiency*. Springer Science & Business Media, 2003, vol. 24.

[14] F. Bach, "Submodular functions: From discrete to continuous domains", *Mathematical Programming*, vol. 175, no. 1-2, pp. 419–459, 2019.

[15] A. Bian, K. Y. Levy, A. Krause, and J. M. Buhmann, "Non-monotone continuous dr-submodular maximization: Structure and algorithms", *CoRR*, vol. abs/1711.02515, 2017. arXiv: 1711.02515. [Online]. Available: http://arxiv.org/abs/1711.02515.

[16] F. Bach, R. Jenatton, J. Mairal, G. Obozinski, *et al.*, "Structured sparsity through convex optimization", *Statistical Science*, vol. 27, no. 4, pp. 450–468, 2012.

[17] F. Bach *et al.*, "Learning with submodular functions: A convex optimization perspective", *Foundations and Trends® in Machine Learning*, vol. 6, no. 2-3, pp. 145–373, 2013.

[18] H. Rauhut, "Compressive sensing and structured random matrices", *Theoretical foundations and numerical methods for sparse recovery*, vol. 9, pp. 1–92, 2010.

[19] D. M. Topkis, *Supermodularity and complementarity*. Princeton university press, 1998.

[20] A. Krause, "Sfo: A toolbox for submodular function optimization", *Journal of Machine Learning Research*, vol. 11, no. Mar, pp. 1141–1144, 2010.