Subset Selection Techniques for Deep/Machine Learning Applications: From Theory to Practice

Loay Mualem

A THESIS SUBMITTED FOR THE DEGREE OF "DOCTOR OF PHILOSOPHY"
DISSERTATION BY PUBLICATIONS

University of Haifa
Faculty of Social Sciences
Department of Computer Science

August 2025

Subset Selection Techniques for Deep/Machine Learning Applications: From Theory to Practice

By: Loay Mualem

Supervised by: Prof. Moran Feldman and Prof. Dan Feldman

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DOCTORAL'S DEGREE

University of Haifa
Faculty of Social Sciences
Department of Computer Science

August 2025

Approved by: Prof. Moran Feldman Moran Feldman Date: 25/08/25
(Supervisor)

Approved by: Prof. Dan Feldman [illegible] Date: 24/08/25
(Supervisor)

Approved by: Prof. Hagit Hel-Or Date: ______
(Chairperson of Advanced Studies Committee)

This research was carried out under the supervision of Prof. Moran Feldman and Prof. Dan Feldman, in the Faculty of Computer Science. The results in this thesis have been published as articles by the author and research collaborators in conferences during the course of the author's doctoral research period, the most up-to-date versions of which being:

- Murad Tukan, Loay Mualem, and Alaa Maalouf. "Pruning Neural Networks via Coresets and Convex Geometry: Towards No Assumptions." Advances in Neural Information Processing Systems 35 (NeurIPS). 2022; see [TMM22].
- Loay Mualem, and Moran Feldman. "Using Partial Monotonicity in Submodular Maximization." Advances in Neural Information Processing Systems 35 (NeurIPS). 2022; see [MF22].
- Loay Mualem, and Moran Feldman. "Resolving the Approximability of Offline and Online Non-monotone DR-Submodular Maximization over General Convex Sets." In International Conference on Artificial Intelligence and Statistics (AISTATS). PMLR, 2023; see [MF23].
- Loay Mualem, Murad Tukan, and Moran Feldman. "Bridging the Gap Between General and Down-Closed Convex Sets in Submodular Maximizaiton." In International Joint Conference on Artificial Intelligence (IJCAI). IEEE, 2024; see [MTF24].
- Loay Mualem, Ethan R. Elenberg, Moran Feldman and Amin Karbasi. "Submodular Minimax Optimization: Finding Effective Sets." In International Conference on Artificial Intelligence and Statistics (AISTATS). PMLR, 2024; see [MEFK24].

Beyond the main line of research described above, I also contributed to additional papers during the doctoral period:

- Murad Tukan, Loay Mualem, and Moran Feldman. "Practical 0.385-Approximation for Submodular Maximization Subject to a Cardinality Constraint." Advances in Neural Information Processing Systems, 37, pp.51223-51253. see [TMF24].

- Meir Goldenberg, Loay Mualem, Amit Shahar, Sagi Snir, and Adi Akavia, 2024. "Privacy-preserving biological age prediction over federated human methylation data using fully homomorphic encryption." In Genome Research, 34(9), pp.1324-1333. see [GMS$^+$24].

Acknowledgements

I am profoundly grateful to everyone who stood by me throughout my Ph.D. journey. Thanks to their unwavering support, I was able to overcome challenges, grow through hardships, and ultimately reach this milestone.

My first and foremost thanks go out to Prof. Moran Feldman and Prof. Dan Feldman. Prof. Moran Feldman was the one who introduced me to the world of submodular functions, sparking my curiosity and expanding my ambitions. Through the open problems he shared with me, I came to appreciate the practical power and significance of these functions. It was a privilege-and a truly transformative experience-to work alongside such a pioneer in the field. Thank you for believing in me during the most difficult times and for guiding me with insight and patience. Prof. Dan Feldman, although our collaboration was shorter, your influence was lasting. You taught me how to channel my theoretical skills into practical challenges, and your ambition reignited my hunger to solve meaningful problems. I'm grateful for the trust you placed in me and for the opportunities you provided for both academic and personal growth.

I am also thankful to my close friends and collaborators: Dr. Murad Tukan, Dr. Alaa Maalouf, and Dr. Ibrahim Jubran. Murad, you are among the most talented and sharp-minded researchers I've ever worked with. Our collaboration was not only productive but also enriching and full of laughter. Your friendship carried me through some of the hardest moments. Alaa, your teaching skills and leadership continue to inspire me. Beyond our collaboration, sharing an apartment with you was a true blessing—I learned a lot from you, both professionally and personally. Ibrahim, you were my first TA during my undergraduate studies, and since then, our bond grew into true brotherhood. Your guidance shaped my path more than you can imagine, and sharing an apartment with you helped me clarify who I want to become.

Beyond the University of Haifa, I wish to thank Prof. Amin Karbasi, Prof. Alan Kuhnle, Dr. Ethan Elenberg, and Yixin Xin. Prof. Amin, working with you was one of the most rewarding experiences of my life. You helped sharpen my thinking and expand my worldview—not only academically but also socially and professionally. Prof. Alan, I'm grateful for the clarity and elegance you brought to our theoretical discussions, which left a lasting mark on my approach to research. Ethan, I will always miss our spontaneous brainstorming sessions, where ideas flowed with energy and excitement. Yixin, your back-and-forth theoretical discussions helped me solidify ideas and see nuances I might've otherwise missed. I cherish all of it.

To my wife, Haneen Mualem, words will never suffice. Your unconditional love, support, and belief in me gave me the strength to pursue this path, even during the darkest and most exhausting times. You stood by me with grace, with calm, and with courage. Because of you, I was able to balance the chaos of research with peace at home. Thank you for your patience, your sacrifices, and your endless encouragement. I am the luckiest man to walk this journey with you.

To my beloved parents and brother—Raed, Ola, and Sari—thank you for being my foundation. Mom and Dad, it is only because of your selflessness that I was able to dream, to grow, and to pursue higher education. You gave me every opportunity without hesitation and without condition. You taught me what strength, resilience, and compassion look like. Sari, your encouragement and pride in me have always motivated me to aim higher and do better. I hope I've made you proud.

To my grandmother, Matheel, you have been my anchor since childhood. You cared for me, believed in me when I was unsure of myself, and offered a warm place to return to, no matter how far I strayed. Your love has been a constant reminder of where I came from and who I want to be. Thank you for everything—for your prayers, your stories, and your quiet strength.

Finally, to my entire extended family—thank you for celebrating my milestones and holding me through the storms. Each one of you played a role, whether through encouragement, care, or simple presence.

Contents

Subset Selection Techniques for Deep/Machine Learning Applications: From Theory to Practice

Loay Mualem

Abstract

We address a broad class of machine learning and optimization problems centered around selecting compact, informative subsets of data for efficient training, inference, and decision-making. In modern large-scale AI systems, both the computational and memory requirements of learning algorithms often render full-data processing impractical. To this end, our work explores both algorithmic and theoretical foundations for subset selection through the lens of coresets and (robust) submodular optimization.

Our contributions span five main directions:

1. We develop a coreset-inspired pruning framework for deep neural networks that identifies and removes redundant parameters with minimal impact on accuracy, enabling faster and more resource-efficient deployment.

2. We introduce a refined analysis of partially monotone submodular functions, showing that they admit improved approximation guarantees compared to general submodular functions. Our results provide a stronger theoretical foundations for submodular maximization in real-world settings.

3. We conducted an empirical analysis of state-of-the-art algorithms for maximizing continuous DR-submodular functions under general convex constraints, extended them to the online setting, and provided new bounds on the best possible approximation guarantees.

4. We propose a novel decomposition-based approach that bridges the gap between down-closed and general convex constraints in DR-submodular maximization, allowing smooth transitions between the two theoretical regimes. Our approach applies to both the offline and online settings.

5. We formulate and approximately solve minimax submodular optimization problems to model uncertainty and adversarial scenarios, providing scalable algorithms with

provable worst-case guarantees.

Throughout the thesis, we emphasize both theoretical rigor and practical applicability. Our methods combine tools from convex analysis, combinatorial optimization, and machine learning theory. The proposed frameworks contribute to improving the scalability, adaptability, and robustness of modern AI systems across a variety of domains.

Chapter 1

Introduction

Artificial Intelligence (AI) is rapidly transforming nearly every domain of modern life, from revolutionizing healthcare [Sha21, ERR$^+$19] and autonomous systems [WMX$^+$24] to enabling scientific discovery [WFD$^+$23], financial decision-making [ID23], and personalized learning [GBK$^+$24]. Advances in deep learning and large scale models—such as Large Language Models (LLMs), Graph Neural Networks (GNNs), and generative models have made these breakthroughs possible. However, alongside this progress lie critical challenges: these models are often **computationally expensive**, **data intensive**, and offer limited **theoretical understanding or guarantees** on their behavior, generalization, accuracy and efficiency, making their deployment in real world, resource constrained, or safety critical environments difficult.

My research focuses on building AI systems that are efficient and scalable through algorithms with strong theoretical guarantees. I aim to reduce the computational and data demands of deep learning models while maintaining their performance and generalization capabilities. To this end, I design optimization-driven methods that accelerate training and inference while offering provable guarantees on approximation quality. These methods provide a principled foundation for improving efficiency without relying solely on empirical heuristics, enabling more reliable and interpretable deployment across a variety of real-world tasks.

To achieve these goals, I leverage two key tools: **submodular optimization** and **coresets**. Submodularity captures the principle of diminishing returns and has proven to be a powerful framework for data selection, summarization, and diversity. Coresets provide small, weighted approximations of large datasets or models while preserving essential properties for learning and inference. While both techniques offer formal guarantees and scalability, they differ in applicability. Coresets often provide strong approximation bounds for specific optimization problems; however, they may not exist or be practical for many problems. In fact, many well-known coresets took years to develop, and designing a coreset for a new problem typically requires deep, problem-specific insight. On the other hand, submodular functions arise more naturally across many machine learning settings where the diminishing returns property offers a principled and often model-agnostic structure. Submodular objectives can thus capture essential structural properties of the data without being tailored to a specific model.

Coresets. Coresets were initially introduced to address problems in computational geometry [AHPV04] and have since gained considerable attention across various domains [BLK18, BLL18, BC08, BEL13, BJKW19, CIM$^+$19, JTMF20, MJTF21, FMSW10, FRVR14, KL19]; comprehensive surveys can be found in [Fel20, MSSW18, Phi16]. Informally, a coreset is a small, weighted subset of the input data (unlike sketches or dimensionality reduction techniques) that approximates the loss of the full dataset P for *every* feasible query x, with provable guarantees—typically within a factor of $1 \pm \varepsilon$ for an error parameter $\varepsilon \in (0, 1)$. The size of such coresets is usually polynomial in $1/\varepsilon$ and often independent of the input size or logarithmic in it. Because coresets approximate all possible queries (not just the optimal one), they support constrained optimization and enable efficient computational models such as merge-and-reduce trees [Fel20]. Importantly, coresets can often be computed in near-linear time, even for many NP-hard problems, allowing expensive or heuristic algorithms to be applied repeatedly on a compact representation of the data.

Coresets are Typically Problem Dependent. Most coresets in the literature are tailored to specific problems, loss functions, or data distributions. That is, the coreset is designed to preserve the structure and optimization landscape of the particular problem at hand. In this sense, the coreset encapsulates the statistical or geometric properties that the target task imposes on the data.

Submodular Functions. Submodular functions provide a powerful and flexible tool for data summarization and subset selection. A set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ is called *submodular* if it exhibits the diminishing returns property: for every $A \subseteq B \subseteq \mathcal{N}$ and $s \in \mathcal{N} \setminus B$, it holds that $f(s \mid A) \geq f(s \mid B)$, where $f(s \mid A) \triangleq f(\{s\} \cup A) - f(A)$ denotes the marginal gain of the element s with respect to the set A. This naturally models the intuition that the benefit of adding a new element to a smaller set is greater than to a larger one, due to overlapping or redundant information. Submodular functions arise in many applications, such as sensor placement, document summarization, influence maximization, and data selection. For example, in coverage functions, where f measures how well a subset represents or covers the full dataset, the marginal gain of adding a data point decreases as more of the dataset is already covered—which makes coverage functions submodular.

Submodular Functions for Problem-Agnostic Subset Selection. In contrast to coresets, which are typically tailored to specific optimization problems and loss functions, submodular optimization offers a general-purpose and modular framework for selecting effective subsets. Once an appropriate submodular objective is defined, it can be applied across a wide range of tasks and datasets, often with strong theoretical guarantees on the quality of the selected subset. This versatility makes submodular functions especially attractive in large-scale settings, where a single selection strategy can support multiple downstream applications without the need for task-specific redesign. In this way, submodular optimization serves as a bridge between problem-specific accuracy and broad applicability.

1.1 Overview

The thesis is composed of five papers, each contributing to the development of efficient and theoretically grounded AI systems through model compression techniques and submodular optimization methods for data and model selection. My main results are five-fold, and are detailed across the following chapters:

Chapter 2 introduces our coreset-inspired pruning method for deep neural networks. This work is driven by the need to accelerate and enhance the training and inference of large-scale models, enabling practical deployment without significant loss in performance.

Chapter 3 focuses on improving the theoretical guarantees for submodular maximization in discrete settings. Specifically, we study partially monotone functions—commonly arising in real-world applications—and demonstrate how leveraging their monotonicity structure enables stronger approximation bounds than those achievable in the general non-monotone case.

Chapters 4 and 5 are devoted to DR-submodular maximization. In Chapter 4, we investigate continuous DR-submodular functions under general convex constraints, and develop principled algorithms with provable guarantees. Building on this foundation, Chapter 5 presents a novel decomposition framework that bridges the gap between general and down-closed convex constraints, enabling smooth interpolation between existing guarantees for these types of constraints and extending the applicability of DR-submodular optimization in both offline and online settings.

Chapter 6 addresses robust submodular optimization under uncertainty. We formulate the problem as minimax optimization over submodular functions and develop scalable algorithms with theoretical approximation guarantees, with applications in adversarial learning scenarios and robustness to uncertainty.

Finally, Chapter 7 summarizes the key contributions and outlines promising directions for future research.

Statement of Authorship

Principal Author

Title of Paper	Pruning Neural Networks via Coresets and Convex Geometry: Towards No Assumption		
Publication Status	☑ Published ☐ Accepted for Publication ☐ Submitted for Publication		
Publication Details	Proceedings of the 36th International Conference on Neural Information Processing Systems: 38003-38019.		
Name of Principal Author (Candidate)	Loay Mualem		
Contribution to the Paper	Contributed to the implementation, theory and writing of the paper.		
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Name and Signature	Loay Mualem	Date	24/6/2025

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the dissertation

Name of Co-Author	Murad Tukan		
Contribution to the Paper	Contributed to the theory and implementation of the paper.		
Name and Signature	Morad Tukan [illegible]	Date	24/06/2025

Name of Co-Author	Alaa Maalouf		
Contribution to the Paper	Contributed to the theory and writing of the paper.		
Name and Signature	Alaa Maalouf [illegible]	Date	24/06/2025

Please cut and paste additional co-author panels here as required.

Chapter 2

Pruning Neural Networks via Coresets and Convex Geometry: Towards No Assumptions

In this chapter, we present a coreset-based framework for compressing deep neural networks, aimed at reducing training and inference costs while preserving model accuracy. Our approach leverages data summarization techniques to construct small, high-quality subsets that approximate the original training distribution.

The following paper [TMM22] was published at the *Conference on Neural Information Processing Systems (NeurIPS 2022).*

Pruning Neural Networks via Coresets and Convex Geometry: Towards No Assumptions

Murad Tukan[*][†]
muradtuk@gmail.com

Loay Mualem[*][†]
loaymua@gmail.com

Alaa Maalouf[*][†]
alaamaalouf12@gmail.com

Abstract

Pruning is one of the predominant approaches for compressing deep neural networks (DNNs). Lately, coresets (provable data summarizations) were leveraged for pruning DNNs, adding the advantage of theoretical guarantees on the trade-off between the compression rate and the approximation error. However, coresets in this domain were either data-dependent or generated under restrictive assumptions on both the model's weights and inputs. In real-world scenarios, such assumptions are rarely satisfied, limiting the applicability of coresets. To this end, we suggest a novel and robust framework for computing such coresets under mild assumptions on the model's weights and without any assumption on the training data. The idea is to compute the importance of each neuron in each layer with respect to the output of the following layer. This is achieved by a combination of Löwner ellipsoid and Caratheodory theorem. Our method is simultaneously data-independent, applicable to various networks and datasets (due to the simplified assumptions), and theoretically supported. Experimental results show that our method outperforms existing coreset based neural pruning approaches across a wide range of networks and datasets. For example, our method achieved a 62% compression rate on ResNet50 on ImageNet with 1.09% drop in accuracy.

1 Introduction and Backround

Deep neural networks (DNNs) achieved state-of-the-art (SOTA) performance on a large variety of tasks, e.g., in computer vision [33, 50] and natural language processing (NLP; [87, 20]). However, DNNs usually contain millions or even billions of parameters in order to achieve SOTA performances resulting in large storage requirements and long inference time. This is obstructive when, e.g., dealing with limited hardware or real-time systems such as autonomous cars and text/speech translation. To this end, a large body of research is dedicated to reducing the size and inference costs of DNNs.

Pruning. A dominant approach widely used for reducing the size of DNNs is to utilize a pruning algorithm to remove redundant parameters from the original, over-parameterized network. In general, pruning can be categorized into two main types: (i) Unstructured pruning [31, 6] reduces the number of non-zero parameters by inducing sparsity into weight parameters, which can achieve high compression rates but requires specialized software and/or hardware in order to achieve faster inference times. (ii) Structured pruning [35, 60, 77] modifies the structure of the underlying weight tensors, by removing filters/neurons from each layer, usually resulting in smaller compression rates while directly achieving faster inference times with no specialized software; see section 4

[*]These authors equally contributed to this paper.
[†]Department of Computer Science, University of Haifa

1.1 Coresets for Pruning

Notably, many recent papers focused on various types of filter pruning [88, 79] potentially due to the empirical observation that existing filter pruning approaches consistently yield impressive results. However, most pruning methods are based on heuristics, lacking theoretical guarantees on the trade-off between the compression rate and the approximation error. This was the motive for introducing coresets [82, 60] to the world of pruning.

Coresets. In machine learning, we are (usually) given an input set $P \subseteq \mathbb{R}^d$ of n points, its corresponding weights function $w : P \to \mathbb{R}$, a feasible set of queries X, and a loss function $\phi : P \times X \to [0, \infty)$. The tuple (P, w, X, ϕ) is called *query space*, and it defines the optimization problem at hand. For a given problem that is defined by its query space (P, w, X, ϕ), and an error parameter $\varepsilon \in (0, 1)$, an ε-coreset is is a small weighted subset of the input points that approximates the loss of the input set P for every feasible query x, up to a provable bound of $1 + \varepsilon$.

Since coresets approximate the cost of every query, traditional (possibly inefficient) algorithms/solvers can be applied on coresets to obtain an approximation of the optimal solution on the full data, using less time and memory; see Section B.2 in the appendix for more details.

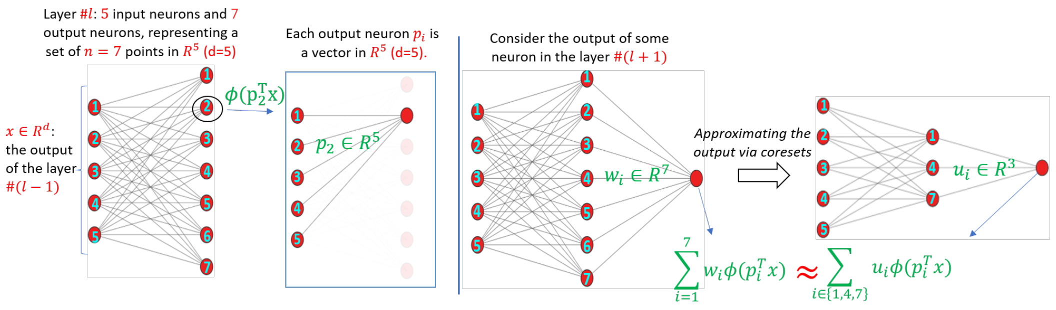

Layer $\#l$: 5 input neurons and 7 output neurons, representing a set of $n = 7$ points in R^5 (d=5)
$x \in R^d$: the output of the layer $\#(l-1)$
$\phi(p_2^T x)$
Each output neuron p_i is a vector in R^5 (d=5).
$p_2 \in R^5$
Consider the output of some neuron in the layer $\#(l+1)$
$w_i \in R^7$
Approximating the output via coresets
$u_i \in R^3$
$\sum_{i=1}^{7} w_i \phi(p_i^T x) \approx \sum_{i \in \{1,4,7\}} u_i \phi(p_i^T x)$

Figure 1: Illustration of our neuron coreset construction on a toy example.

Pruning via coresets. Recently, some inspiring innovative frameworks [82, 60, 5, 7] leveraged the idea of coresets for pruning DNNs. Any layer ℓ can be represented as a set $P = \{p_1, \cdots, p_n\}$ of $n > 1$ points in $\mathbb{R}^d$, where d is the number of input neurons, and n in the number of output neurons, i.e., each point $p \in P$, represents a specific neuron using its d weights (parameters). When the layer receives an input vector $x \in \mathbb{R}^d$, it outputs the vector $(\phi(p_1^T x), \cdots, \phi(p_n^T x))$, where $\phi : \mathbb{R} \to \mathbb{R}$ is an activation function which defines a non-linear mapping. Focusing on a single neuron η in the layer that follows ℓ, defined by its corresponding vector of weights $w = (w_1, \cdots, w_n)$, we set in the context of coresets, $w(p_i) := w_i$ for every $i \in \{1, \cdots, n\}$ - this is just a mapping from p_i to w_i to simplify the writing and reading. Note that the output of this neuron is $\sum_{p \in P} w(p)\phi\left(p^T x\right)$. Assuming that we are given an ε-coreset (C, u) for the query space $\left(P, w, \mathbb{R}^d, \phi\right)$ where $C \subseteq P$, and $u : C \to \mathbb{R}$, we have that (C, u) approximates the output of this specific neuron η for every query using less parameters; see Figure 1. To formalize the stated above, we now define coresets in the context of activation functions.

Definition 1.1 (Coreset for activation functions)**.** Let $\varepsilon \in (0, 1)$, and let $\left(P, w, \mathbb{R}^d, \phi\right)$ be a query space. Then the pair (C, u), is an ε-coreset for $\left(P, w, \mathbb{R}^d, \phi\right)$ if (i) $C \subseteq P$, (ii) $u : C \to [0, \infty)$, and (iii) for every $x \in X$, $\left|1 - \frac{\sum_{q \in C} u(q)\phi\left(q^T x\right)}{\sum_{p \in P} w(p)\phi\left(p^T x\right)}\right| \leq \varepsilon$.

Since C is a subset of P, we can remove (assign zero to) all weights from w that corresponds to points not chosen to be in C from P, and replace the weights of the chosen points in C with the new weights vector u; see Figure 1 in [82] for a visual illustration. **To prune neurons,** we refer the reader to Section 2.5 as it is a simple extension. Prior work showed that such approaches successfully result in high compression rates across a wide range of networks and datasets, and even achieves SOTA performance on a verity of them.

The main (strong) advantage of the coreset approach over others was the provided provable theoretical guarantees on the tradeoff between the compression rate and the approximation error, which supports worse case scenarios. In addition, coresets play an important role in improving the generalization properties of the trained networks [5, 78].

Sensitivity sampling for constructing pruning coresets. To compute such coresets, both [60, 82] utilised the known sensitivity sampling framework [9, 52]. In short the sensitivity of a point $p \in P$ in some query space (P, w, X, ϕ) corresponds to the importance of this point with respect to the query space at hand, and it is defined as $s(p) = \sup_{x \in X} \frac{w(p)\phi(p,x)}{\sum_{q \in P} w(q)\phi(q,x)}$ - where the denominator is not equal to zero. Once we bound these sensitivities, we can sample points (neurons) from P according to sensitivity bounds, and re-weight the sampled points to obtain a coreset. The size of the sample is proportional to the sum of these bounds. See Section B.1 and Theorem B.2 for more details in the Appendix.

1.2 Our contribution

Prior coreset methods for pruning DNNs either (i) imposed restrictive assumptions both on the model's weights and inputs [82], i.e., the input set P representing the neurons, and the query set X which represents the inputs of the layer, are enclosed in a ball in $\mathbb{R}^d$ of radius r_1 and r_2, respectively, or (ii) the methods are data-dependent, i.e., use a mini-batch of the input set to measure the influence of each parameter on the loss function [5, 60].

To this end, in this work, we take coresets a step further into the realm of pruning by introducing a unified framework with provable guarantees for pruning DNNs (weights and neurons/filters) while minimally affecting the generalization error. Our main improvement is that our framework is simultaneously (i) **data-independent**, (ii) **requires a single assumption** on the model's weights, and (iii) **provably guarantees a multiplicative factor approximation**, which is favourable upon additive approximations; see Theorem 2.6. The approach is based on the widely used theory of coresets allowing us to suggest a provable guarantee on the tradeoff between the approximation error and compression rate for each layer.

We conducted experimental results which established new SOTA benchmarks for structured pruning via coresets across a wide range of networks and datasets. We share all of our resulted models [14].

2 Method

In general, the coreset (for pruning) technique hinges upon the insight that any linear layer such as convolutions, can be casted as a matrix multiplication [82]. Hence, we focus in what follows on fully connected (FC) layers, while the details holds for any linear layer. Furthermore, for simplicity, we assume in what follows that the weights of P are all equal to 1 and thus our query space is denote by $\left(P, \mathbb{R}^d, \phi\right)$, and the sensitivity of a point $p \in P$ is simply $s(p) = \sup_{x \in X} \frac{\phi(p,x)}{\sum_{q \in P} \phi(q,x)}$. Note that our proofs are easily extended to the general case where we are given a weight function $w : P \to \mathbb{R}$ as discussed in Section 2.5.

2.1 Preliminaries

Notations. For a positive integer n, we use $[n]$ to denote the set $\{1, \ldots, n\}$. For $c \in \mathbb{R}^d$ and a symmetric positive definite matrix $G \in \mathbb{R}^{d \times d}$, we define $E\left(G, c\right) := \left\{x \in \mathbb{R}^d \middle| \left(x - c\right)^T G \left(x - c\right) \leq 1\right\}$ to be the ellipsoid defined by c and G. For an ellipsoid $E\left(G, c\right)$, each endpoint of a semi principal axis is called a vertex of $E\left(G, c\right)$. We define $\operatorname{rank}\left(P\right)$ for any set $P \subseteq \mathbb{R}^d$ to be the dimension of the affine subspace that P lies on. For a set $P \subset \mathbb{R}^d$ the convex hull of P is denoted by $\operatorname{Conv}(P)$. Finally, vectors are treated as column vectors.

2.2 Novelty - Löwner ellipsoid meets Carathéodory

Our method hinges upon a combination of two known tools from convex geometry. The novelty of our approach exploits the following observation. Most activation functions ϕ are continuous non-decreasing functions, which indicate that for every query x and a set of points P, the maximal contribution to $\sum_{p \in P} \phi(p^T x)$ with respect to such activation function is associated to a point on the convex hull of P. By finding a geometrical body B of bounded number of vertices, that is (i)

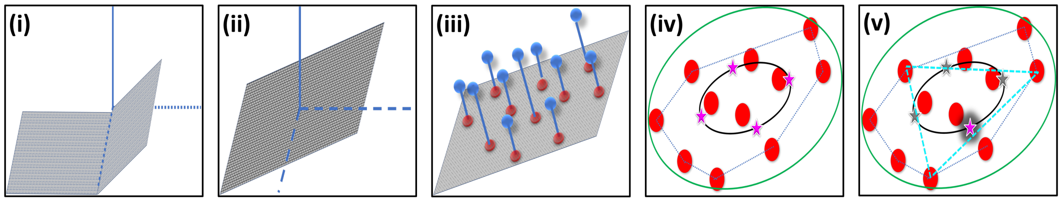


Figure 2: Our novelty in a nut-shell: The very first steps of our technique rely on bounding the ReLu activation function by the ℓ_1-regression loss function, e.g., for $\mathrm{ReLU}\left(p^T x\right)$, where $p = (1, 0)$ in this example (shown in (i)), we first bound it by the ℓ_1-regression loss function (shown in (ii)) using Definition 2.3. Following this step, a set of points P can be projected into a low dimensional subspace of dimension $\mathrm{rank}\,(P)$ using any dimensionality reduction algorithm as presented in (iii), resulting in the set P'. (iv) The convex hull (blue dashed lines) P' (red points) is enclosed by its Löwner ellipsoid (depicted in green). (v) Finally, for each vertex (magenta star) of the enclosed ellipsoid (black ellipsoid), its Carathédory set is found (red points connected by cyan dashed lines).

enclosed in $\mathrm{Conv}\,(P)$ and (ii) with some dilation factor (expanding) B enclose $\mathrm{Conv}\,(P)$, we will be able to represent each point p on the boundary of the convex hull of P as a convex combination of two points p_1, p_2, one of each (p_1) on B and the second (p_2) on its dilated form, which is formalized as the set $\{\alpha(x - c) + c \mid x \in B, \forall \alpha \in [0, 1]\}$, where c here denotes the center of B. For such task, Löwner ellipsoid is leveraged.

Theorem 2.1 (John-Löwner ellipsoid[42]). *Let $L \subseteq \mathbb{R}^d$ be a set of points such that the convex hull of L has a nonempty interior. Then, there exists an ellipsoid $E\,(G, c)$ (also known as the* MVEE*), where $G \in \mathbb{R}^{d\times d}$ is a positive definite matrix and $c \in \mathbb{R}^d$, of minimal volume such that $\frac{1}{d}\left(E\,(G, c) - c\right) + c \subseteq \mathrm{Conv}\,(L) \subseteq E\,(G, c)$. If L is symmetric around the center c, then the dilation factor can be reduced to $\frac{1}{\sqrt{d}}$.*

```
Algorithm 1: $\ell_\infty$-CORESET $(P, m)$
Input  : $P \subseteq \mathbb{R}^d$ of $n$ points with rank $r$
Output : A subset $S \subseteq P$ that satisfies Lemma 2.5
1  $(Y, z)$ := affine subspace that $P$ lies on,
     i.e. $P \subseteq \left\{xYY^T + z \middle| x \in \mathbb{R}^d\right\}$
2  $P' := \left\{p' := pY^T\right\}$
3  Let $\mathrm{map} : P' \to P$ a function that maps
     every $p' \in P'$ to its corresponding point
     $p \in P$
4  $S := \emptyset; K := \emptyset$
5  $E\,(G, c) := \mathrm{MVEE}\,(\mathrm{Conv}\,(P'))$
6  $V$ := the vertices of ellipsoid
     $\frac{1}{r}\left(E\,(G, c) - c\right) + c$
7  for every $v \in V$ do
8      $K :=$
         $K \cup$ CARATHEODORY-SET $(v, P')$
         // See Algorithm 4 in the
         supplementary material.
9  end
10 $S := \{\mathrm{map}\,(q) | q \in K\}$
11 return $S$
```

```
Algorithm 2: CORESET $(P, m)$
input  : A set $P \subseteq \mathbb{R}^d$ of $n$ points, and a
         sample size $m$
output : A weighted set $(C, u)$
1  $Q := P, i := 1, C := \emptyset$
2  while $|Q| \geq 2\mathrm{rank}\,(Q)^2$ do
3      $S_i := \ell_\infty$-CORESET $(Q)$
4      for every $p \in S_i$ do
5          $s(p) := \frac{2\mathrm{rank}(Q)^{1.5}}{i}$
6      end
7      $Q := Q \setminus S_i, i := i + 1$
8  end
9  for every $p \in Q$ do
10     $s(p) := \frac{2\mathrm{rank}(Q)^{1.5}}{i}$
11 end
12 $t := \sum_{p \in P} s(p)$
13 $C$ := an i.i.d sample of $m$ points from $P$,
     where each $p \in P$ is sampled with
     probability $\frac{s(p)}{t}$.
14 $u(p) := \frac{t}{m \cdot s(p)}$ for every $p \in C$
15 return $(C, u)$
```

Afterwards, p_1 and p_2 should be represented by points from P. Each point on B (specifically, p_1) can be represented via a convex combination of $d + 1$ points from P. The same holds for points on the dilated form of B (e.g., p_2) but via a conical combination (linear combination where the weights are non-negative and the sum of weights is not necessarily 1). This problem is solved by invoking Carathéodory theorem.

Theorem 2.2 ([10, 95])**.** *For any $A \subset \mathbb{R}^d$ and $p \in \operatorname{Conv}(A)$, there exists $m \leq d+1$ points $p_1, \ldots, p_m \in A$ (denoted by a Carathéodory set of p) such that $p \in \operatorname{Conv}(\{p_1, \ldots, p_m\})$.*

Finally, it is known that some functions, including the ReLU function, do not admit an ε-coreset of size $o(n)$ [82, 81]. Thus, we use a generalized form of what is known as the complexity measure of a set of points, which was first introduced in [81] and later leveraged in [76]. This measure is used to determine the complexity of a given set P with respect to ReLU, and the coreset size theoretically.

Definition 2.3 (Regression Complexity Measure)**.** Let $P \subseteq \mathbb{R}^d \times \{1\}$, the regression complexity measure of P is defined as $\mu(P) = \sup_{x \in \mathbb{R}^{d+1}} \frac{\sum_{q \in \{p \in P | p^T x \leq 0\}} |q^T x|}{\sum_{q \in \{p \in P | p^T x > 0\}} |q^T x|}$, where the denominator is ≥ 0, and the last entry of every $p \in P$ is 1, reserved for the bias/intercept term.

2.3 Our Pruning Scheme

In what follows, we present our data summarization technique for ReLU on the dot product function. Then in Section 2.5, we discuss that our results can be easily extended to a wide family of activation functions including the Sigmoid function, as recently shown in [76]. First we present Algorithm 1, which serves as a stepping stone towards bounding the sensitivities.

Overview of Algorithm 1. The algorithm receives as input a set $P \subset \mathbb{R}^d$ whose rank is $r \in [d]$ and deterministically finds a subset $S \subseteq P$ which satisfies that for every $j \in [d]$, $X \in \mathbb{R}^{d \times j}$ and $v \in \mathbb{R}^d$, $\frac{\max_{p \in P} \|(p-v)X\|_1}{\max_{p \in S} \|(p-v)X\|_1} \leq r^{1.5}$. To do so, first, we find the affine hyperplane that P lies on, followed by computing the low dimensional representation of P, denoted by P'; see Lines 1–2. Note that if $\operatorname{rank}(P) = d$, then we can either keep P as it is (i.e., $P' := P$), or use dimensionality reduction tricks as detailed in Section 2.5. To compute the output S, we first bound the convex hull of P' by its Löwner ellipsoid $E(G, c)$ in Line 5, followed by computing the dilated ellipsoid of $E(G, c)$, namely, $\frac{1}{r}(E - c) + c$. Let V be the set of vertices of such ellipsoid; see Line 6. Now, for each point $v \in V$, we represent it as a convex combination of $r+1$ points from P' via Theorem 2.2, and store the union of such sets (each of size at most $r+1$) of points into K as done in Lines 7–9. For each point in K, we map it back to $\mathbb{R}^d$ to satisfy Lemma 2.5. To sum up Algorithm 1, we observe that the vertices can be used via canonical combinations with their dilated form to describe every point on the convex hull of the input data (in our end, it would the network's weights). Hence the Carathéodory set of these vertices from the input points lying on the convex hull can be further used to also represent points lying on the convex hull. This is the core idea which enable us in forming our ℓ_∞ coreset for any ℓ_ρ regression problem where $\rho \in (0, \infty)$.

We now discuss Algorithm 2 which is responsible for constructing an ε-coreset with respect to activation functions. Its input is a set $P \subset \mathbb{R}^d$ and a sample size $m \geq 1$.

Overview of Algorithm 2. First set $Q := P$. At each iteration i, the algorithm obtains a subset $S_i \subseteq Q$ as an output to a call to ℓ_∞-$\textsc{Coreset}(Q)$ as stated in Line 3 of Algorithm 1 such that for every $x \in \mathbb{R}^d$, it holds that $\frac{\max_{p \in P} |p^T x|}{\max_{p \in S_i} |p^T x|} \leq r^{1.5}$ with r being the rank of Q. The sensitivity of each point in S_i is bounded from above by $\frac{2d^{1.5}}{i}$ as stated in Lines 4–6. The idea behind these bounds lies in our proof of Theorem 2.6. The set S_i is removed from Q, and this procedure is repeated with respect to Q until the size of Q is small enough. The obtained sensitivities are the ones needed for computing the pruning coresets. Finally, we utilize the sensitivity sampling framework of [9] to obtain the desired coreset; see Lines 13–14.

2.4 Analysis

In this section, we prove the correctness of our algorithms. The following lemma shows that for each point p that is inside some convex hull S, its ℓ_1 distance to any affine subspace is always bounded from above by ℓ_1 distance from the same affine subspace, of some other point $q \in S$.

Lemma 2.4. *Let $d, \ell, m \geq 1$ be integers. Let $p \in \mathbb{R}^d$ and $A \subseteq \mathbb{R}^d$ be a set of m points with $p \in \operatorname{Conv}(A)$ so that there exists $\alpha : A \to [0,1]$ such that $\sum_{q \in A} \alpha(q) = 1$ and $\sum_{q \in A} \alpha(q) \cdot q = p$. Then for every $Y \in \mathbb{R}^{d \times \ell}$ and $v \in \mathbb{R}^\ell$, $\|(p-v)Y\|_1 \leq \max_{q \in A} \|(q-v)Y\|_1$.*

The following states the provable guarantees of Algorithm 1.

Lemma 2.5 (ℓ_∞-coreset for ℓ_1-regression)**.** *Let $P \subseteq \mathbb{R}^d$ be a set of points, and r be the rank of P. Let $j \in [d-1]$ and let S be the output of a call to ℓ_∞-*CORESET(P). *Then (i) $|S| \in O\left(r^2\right)$, and (ii) for every $X \in \mathbb{R}^{d\times j}$ and $v \in \mathbb{R}^d$, $\frac{\max_{q\in P}\|(q-v)X\|_1}{\max_{q\in S}\|(q-v)X\|_1} \in \left[1, 2r^{1.5}\right]$.*

The following theorem states our main result.

Theorem 2.6 (ReLU ε-coreset)**.** *Let $P \subseteq \mathbb{R}^d$, $\varepsilon, \delta \in (0,1)$, $r = \operatorname{rank}(P)$, $\mu(P)$ be as in Definition 2.3, and let $m \in O\left(\frac{\mu(P)r^{3.5}\log n}{\varepsilon^2}\left(d\left(\log\left(\mu\left(P\right)r\log n\right)\right) + \log\left(\frac{1}{\delta}\right)\right)\right)$. Let (C, w) be the output of a call to* CORESET(P, m)*; see Algorithm 2. Then, with probability at least $1-\delta$, (C, w) is an ε-coreset for $\left(P, \mathbb{R}^d, \operatorname{ReLU}(\cdot)\right)$; see Definition 1.1.*

Time analysis. Letting r be the rank of P, the time complexity of Algorithm 1 can be dissected to two main parts: (i) Computing the Löwner ellipsoid in $O(nr^2\log n)$ time using the method proposed in [98] and (ii) computing the Carathédory set in $O(nr + r^4\log n)$ time via [70]. Since V can contain up to $O(r^2)$ points, the overall time for Algorithm 1 is $O(nr^2\log n + nr^3 + r^6\log n)$. As for Algorithm 2, it takes $O(n^2r^2\log n + nr^4\log n)$. Indeed, as explained in Section 2.5, a dimensionality reduction algorithm may be applied to improve the run time (reducing the r^6 factor). Furthermore, the run time of our algorithm can be improved, using the merge-and-reduce tree from the literature of coresets to reduce the n^2 terms to $n\log n$, i.e., the running time can be reduced to $O(n\log^2(n)r^2 + nr^4)$. For a data-independent provable method, this running time is reasonable.

Our advantages over previous results. Our coreset supports different activation functions without the need to change the sensitivity that much. Specifically, it will only be multiplied by some scalar, unlike previous coresets where different losses impose drastically different sensitivities/leverage scores and algorithms. This is since our coreset unlike other coresets is in its essence a framework of coresets for different ℓ_ρ losses, as it can be used as is for different ℓ_ρ losses and yet still attain ϵ-approximation. In addition, when the rank of the input points is small, then our method outperforms previous methods.If the input data is of full rank, previous methods obtain faster coresets construction.

On the boundness of the regression complexity measure. First of all, there exists an example where the complexity measure is unbounded, e.g., consider a set of points distributed evenly on a unit ball. In this case, you can always find a point where a hyperplane separating it from the rest of the points can be found such that the one half-space of this hyperplane contains only this point while the other half-space contains the rest of the points. This leads to an infinite complexity measure. Such an example is also mentioned in [82], when assessing the hardness of generating multiplicative-approximation coresets for ReLU functions.

Theoretically, the complexity measure is influenced by how free can the bias term be (the last entry of x); see Definition 2.3. This term is the only thing that can ensure that one point can be separated from the rest in the sense of finding a separating hyperplane, leading to an infinite complexity measure. Bounding on this term, leads to bounded complexity measure from a theoretical point of view.

In the context of model pruning, from the perspective of the complexity measure, the model's weights are the input denoted by a matrix $P \in \mathbb{R}^{n\times(d+1)}$, while the query is now $\mathbb{R}^d \times \{1\}$. Thus the complexity measure is now $\mu(P) := \sup_{x\in\mathbb{R}^d\times\{1\}} \frac{\sum_{q\in\{p\in P|p^Tx\leq 0\}}|q^Tx|}{\sum_{q\in\{p\in P|p^Tx>0\}}|q^Tx|}$. With this in mind, we observe that the complexity measure is now an instance of the complexity measure used in [76]. The complexity measure now relies entirely on the structure of the model's weights, where the goal is to find the largest ratio between the sum of the absolute of the values inside the rectified neurons prior to applying the rectification, and the sum values of non-rectified neurons. To bound this measure, we can use a variant of the algorithm described in the proof of Theorem 3 in [81].

2.5 Extensions

Our suggested scheme can be extended to support many other variants of the pruning problem.

Various activation functions. Our result can be extended to a family of activation functions called "Nice hinge functions"; see Definition D.1. Let (P, X, ϕ) be a query space, where ϕ is a "Nice hinge functions". To bound the sensitivity of a point p, we first bound the nominator of $s(p)$ by proving that $\forall x \in X : \phi(p^Tx) < \left|p^Tx\right|$. For bounding the denominator from below, recently [76] proved that $\forall x \in X : \sum_{p\in P}\operatorname{ReLU}\left(p^Tx\right) \lessapprox \sum_{p\in P}\phi(p^Tx)$; see full detail in Section D.3.

Weighted Input. In the context of deep learning, the output of each neuron is multiplied with a scalar which brings the necessity of having the ability to deal with weighted set of points. Algorithm 2 can be extended easily to the case as generously detailed in Section D.1 in the Appendix.

Dimensionality reduction. All coreset-based pruning methods rely heavily on the dimensionality of the model's layers, as well as our method. To ensure sufficient pruning ratio, we apply either PCA, TSNE, MDS, or the JL transform on the weights of each layer prior to generating its coreset.

From weight to neuron pruning. Most coreset-based pruning methods, e.g., [5, 82], first provide a scheme for (provable) weight pruning, which is then used as a stepping stone towards pruning neurons as follows. [5] first suggested coreset-based neuron pruning via the use of a generated controled set of queries to evaluate the importance of weights. Any neuron that has a maximal activation value lower or equal to zero, will be pruned from the network as its impact on the rest of the neurons is minimal. On the other hand [82] altered the definition of sensitivity such that it takes into account the sensitivity of a neuron in a layer ℓ with respect to all the neurons in the layer $\ell + 1$, which basically means that the sensitivity of each neuron is taken be the maximal sensitivity over every weight function (neuron in the next layer) defined by the layer. Hence, we follow the same logic for such method; see Section D.2 in the supplementary material.

From neuron to filter pruning. Convolutions can be expressed as matrix multiplications, which enables our method to prune filters from any model as done by [60, 82, 61].

3 Experimental Results

In this section, we study various widely used network architectures and benchmark data-sets. Following [82], to test the robustness of our methods on each of the neuron and filter pruning tasks independently, two sets of experiments are conducted. The first focuses on pruning neurons (Section 3.1) whereas the second focuses on pruning filters (Section 3.2), both via our coreset method.

The setting. In all experiments we report the *Pruning ratio* – the percentage of the parameters that were removed from the original mode. Here, *PR* stands for pruning ratio, *FR* stands for floating-point reduction ratio and *Err* – the percentage of misclassified test instances of our method compared to coreset-based pruning methods and more. *Baseline Err* is the error of the original uncompressed network, while *Pruned Err* is the classification error of the compressed model. In our experiment we compress and fine-tune the network once, no iterative pruning was applied, thus, the compared methods also satisfy this setting. Each experiment was conducted 5 times, in the tables, we report for our method the best error achieved and we highlight in parentheses next to it the average error and standard deviation across the 5 trails. In all of our experiments, the models are fine-tuned till convergence (after pruning). Implementation details are given in Section E in the Appendix.

Software/Hardware. Our algorithms were implemented in Python 3.6 [108] using Numpy [83], and Pytorch [84]. Tests were performed on NVIDIA DGX A100 servers with 8 NVIDIA A100 GPUs each, fast InfiniBand interconnect and supporting infrastructure.

Baselines. Our results are compared to (i) PFP [60], (ii) FT [58], (iii) SoftNet [34], (iv) ThiNet [68], and (v) PvC [82], (vi) Soft Pruning [34], (vii) CCP [85], (viii) FPGM [36], (ix) ThiNet-70, (x) ThiNet-50 [68], (xi) Pruning via Coresets (PvC) [82], (xii) Pruning from Scratch (PfS) [111], and (xiii) Rethinking the value of network pruning (Rethink) [65].

3.1 Neuron Pruning

We test our method on VGG16 [92] using CIFAR10 [49], and LeNet300-100 using MNIST [55].

Discussion. Table 1 present the results of LeNet300-100 and VGG16. Observe that in both architectures, our method outperformed the competing methods under the same compression scenarios. For example, we pruned roughly 90% of the parameters of the LeNet-300-100 model while improving the accuracy of the original model. We witness a similar phenomena on the VGG16 model, where we pruned roughly 90% of the parameters of the dense layers resulting in accuracy improvement. This confirms the insights in [5] that coresets help in improving the generalization properties of DNNs.

Table 1: Pruning of LeNet-300-100 architecture on the MNIST dataset and of VGG-16 on the CIFAR-10 dataset. Here, we report the compression with respect to the fully connected layers only.

Model	Method	Baseline Err. (%)	Pruned Err. (%)	PR (%)
LeNet-300-100	PFP	1.59	2.00	84.32
	FT	1.59	1.94	81.68
	SoftNet	1.59	2.00	81.69
	ThiNet	1.59	12.17	75.01
	PvC	2.16	2.03	90
	Our method (90)	$\mathbf{2.07}$	$\mathbf{1.98}$ $(\mathbf{2.02 \pm 0.04})$	$\mathbf{90}$
	Our method (92.6)	$\mathbf{2.07}$	$\mathbf{2.64}$ $(\mathbf{2.74 \pm 0.1})$	$\mathbf{92.6}$
	Our method (94.6)	$\mathbf{2.07}$	$\mathbf{3.51}$ $(\mathbf{3.58 \pm 0.07})$	$\mathbf{94.6}$
VGG-16	PvC	8.95	8.16	75
	Our method	$\mathbf{5.9}$	$\mathbf{5.9}$ $(\mathbf{6.2 \pm 0.3})$	$\mathbf{90}$

3.2 Filter pruning

We compressed the convolutional layers of (i) ResNet50 [33] on ILSVRC-2012 [18], (ii) ResNet56 [33], (iii) VGG19 [92] on CIFAR10 and (iv) VGG16 [92] on CIFAR10.

Table 2: Filter pruning results on different neural networks with respect to the CIFAR10 dataset.

Model	Method	Baseline Err. (%)	Pruned Err. (%)	PR (%)	FR (%)
VGG-19	PfS	6.4	6.29	52	NA
	Rethink	6.5	6.22	80	NA
	Structured Pruning	6.33	6.20	88	NA
	PvC	6.33	6.02	88	NA
	Our method (88)	$\mathbf{6.33}$	$\mathbf{5.85}$ $(\mathbf{6.03 \pm 0.18})$	$\mathbf{88}$	NA
	Our method (91.28)	$\mathbf{6.33}$	$\mathbf{6.23}$ $(\mathbf{6.35 \pm 0.12})$	$\mathbf{91.28}$	NA
VGG-16	ThiNet	7.11	9.24	63.95	64.02
	FT	7.11	8.22	80.09	80.14
	SoftNet	7.11	7.92	63.95	63.91
	PFP (94)	7.11	7.61	94.32	85.03
	PFP (87)	7.11	7.17	87.06	70.32
	PFP (80)	7.11	7.06	80.02	59.21
	Our method (95.32)	$\mathbf{7.11}$	$\mathbf{7.31}$ $(\mathbf{7.55 \pm 0.24})$	$\mathbf{95.32}$	$\mathbf{85.09}$
	Our method (87)	$\mathbf{7.11}$	$\mathbf{6.63}$ $(\mathbf{6.76 \pm 0.13})$	$\mathbf{87}$	$\mathbf{68.2}$
	Our method (79.53)	$\mathbf{7.11}$	$\mathbf{6.3}$ $(\mathbf{6.38 \pm 0.08})$	$\mathbf{79.53}$	$\mathbf{59.14}$
ResNet56	ThiNet	7.05	8.433	49.23	49.74
	Channel Pruning	7.2	8.2	N/A	50
	AMC	7.2	8.1	64.78	50
	CCP	6.5	6.42	57	52.6
	PvC	6.21	7.0	55	N/A
	Our method	$\mathbf{6.61}$	$\mathbf{7.26}$ $(\mathbf{7.56 \pm 0.3})$	$\mathbf{63.95}$	$\mathbf{50}$

Table 3: Filter pruning of ResNet50 on ImageNet (ILSVRC-2012).

Method	Baseline Err. (%)	Pruned Err. (%)	PR (%)	FR (%)
PFP	23.87	24.79	44.04	30.05
Soft Pruning	23.85	25.39	49.35	41.80
CCP	23.85	24.5	56	48.8
FPGM	23.85	25.17	62	53.5
ThiNet-70	27.72	26.97	33.72	36.78
ThiNet-50	27.72	28.0	51.5	55.82
PvC	23.78	25.11	62	N/A
Our method	$\mathbf{23.78}$	$\mathbf{24.87}$ $(\mathbf{25.07 \pm 0.2})$	$\mathbf{62}$	$\mathbf{61.5}$
Our method (18.01)	$\mathbf{23.78}$	$\mathbf{24.1}$ $(\mathbf{24.2 \pm 0.1})$	$\mathbf{18.01}$	$\mathbf{10.82}$

Filter pruning of DNNs on CIFAR10 and ImageNet (ILSVRC-2012). For Cifar10, we used PyTorch implementations of VGG19 and VGG16. We compressed both models using our approach by different compression rates as shown in Table 2 with a comparison to other methods. For ImageNet, we compressed the baseline model of ResNet50 [33] roughly by 62% in terms of number of parameters. Table 3 provides comparison between our method and other baselines.

Discussion. As can be seen in Tables 1, 2, and 3, our method either outperforms the competing methods or achieves comparable results. As for the coreset methods, our algorithm achieves better result than all of them in this setting, e.g., we compressed 62% of ResNet50 trained on ImageNet while incurring 1.09% drop in accuracy, improving the recent coreset result of PvC [82] for the same compression ratio, while PFP [60] compressed 44.04% to achieve comparable results.

4 Related work

DNNs can be compressed before training [97, 110, 57], during training [118, 115, 51], or after training [93]. Furthermore, such procedures may also be repeated iteratively [88]. As previously noted pruning can be categorized into structured and unstructured pruning.

Unstructured pruning. Weight pruning [56] techniques aim to reduce the number of weights in a layer while approximately preserving its output. Approaches of this type include the works of [54, 21, 39, 1, 62], where the desired sparsity is embedded as a constraint or via a regularizer into the training pipeline, and those of [31, 88, 30], where weights with absolute values below a threshold are removed. The approaches of [5, 6] use a mini-batch of data points to approximate the influence of each parameter on the loss function. Other data-informed techniques include [28, 63, 80, 79, 116]. A thorough overview of recent pruning approaches is given by [27, 8]. However, unlike our approach, weight-based pruning approaches generate sparse models instead of smaller ones thus requiring specialized hardware and sparse linear algebra libraries in order to speed up inference.

Structured pruning. Pruning entire neurons and filters directly shrinks the network leading to smaller storage requirements and improved inference-time performance on any hardware [59, 67]. Lately, these approaches were investigated in many papers [64, 59, 11, 36, 22, 46, 114, 113]. Usually, filters are pruned by assigning an importance score to each neuron/filter, either solely weight-based [37, 35] or data-informed [116, 60], and removing those with a score below a threshold. The procedure can be embedded into an iterative pruning scheme [88] that requires potentially expensive retrain cycles.

Tensor decomposition. Some of the work in DNN compression entails decomposing the layer into multiple smaller ones, e.g., via low-rank tensor decomposition [19, 41, 74, 48, 96, 40, 2, 105, 117, 53, 61]. Other approaches to tensor decomposition include weight sharing, random projections, and feature hashing [112, 3, 91, 12, 13, 107]. However, such techniques usually require expensive approximation algorithms or use heuristics since tensor decomposition is generally NP-hard.

Coresets. In the recent years, coresets got increasing attention, and where leveraged to compress the input datasets of many machine learning algorithms, improving there performance, e.g., regression [72, 38, 81, 47, 103], decision trees [44], matrix approximation [26, 70, 25, 89, 73], data discretization [75], clustering [24, 29, 66, 4, 45, 90, 106], ℓ_z-regression [16, 17, 94], *SVM* [32, 101, 99, 100, 102], deep learning models [69, 5, 60] and even for path planning in the field of robotics [104]. For extensive surveys on coresets, we refer the reader to [23, 86, 43, 71].

5 Conclusions and Future Work

In this paper, we provided a coreset-based pruning technique that hinges upon a combination of tools from convex geometry, while achieving SOTA results with respect to coreset-based structured pruning approaches on a variety of networks. Our main improvement is that our coreset is (training) data-independent and assumes a single assumption on the models weights.

Future work includes (i) suggesting a coreset based budget allocation framework, to determine the (optimal) per layer prune ratio while achieving an overall desired compression rate, (ii) extending our coreset technique to other layers such as attention layers, and (iii) bridging the gap between coreset based pruning approaches and tensor-decomposition methods, as both techniques are theoretically supported by bounding the approximation error given specific compression rate, we can leverage these bounds to formulate the compression problem as an optimization problem which iterates between the two approaches to search for the local minimum.

References

[1] Alireza Aghasi, Afshin Abdi, Nam Nguyen, and Justin Romberg. Net-trim: Convex pruning of deep neural networks with performance guarantee. In *Advances in Neural Information Processing Systems*, pages 3180–3189, 2017.

[2] Jose M Alvarez and Mathieu Salzmann. Compression-aware training of deep networks. In *Advances in Neural Information Processing Systems*, pages 856–867, 2017.

[3] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *International Conference on Machine Learning*, pages 254–263, 2018.

[4] Olivier Bachem, Mario Lucic, and Silvio Lattanzi. One-shot coresets: The case of k-clustering. In *International conference on artificial intelligence and statistics*, pages 784–792. PMLR, 2018.

[5] Cenk Baykal, Lucas Liebenwein, Igor Gilitschenski, Dan Feldman, and Daniela Rus. Data-dependent coresets for compressing neural networks with applications to generalization bounds. In *International Conference on Learning Representations*, 2019.

[6] Cenk Baykal, Lucas Liebenwein, Igor Gilitschenski, Dan Feldman, and Daniela Rus. Sipping neural networks: Sensitivity-informed provable pruning of neural networks. *arXiv preprint arXiv:1910.05422*, 2019.

[7] Gantavya Bhatt and Jeff Bilmes. Tighter m-DPP Coreset Sample Complexity Bounds. In *ICML 2021 Workshop: SubSetML: Subset Selection in Machine Learning: From Theory to Practice*, Virtual, July 2021.

[8] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? In *Proceedings of Machine Learning and Systems 2020*, pages 129–146. 2020.

[9] Vladimir Braverman, Dan Feldman, and Harry Lang. New frameworks for offline and streaming coreset constructions. *arXiv preprint arXiv:1612.00889*, 2016.

[10] Constantin Carathéodory. Über den variabilitätsbereich der koeffizienten von potenzreihen, die gegebene werte nicht annehmen. *Mathematische Annalen*, 64(1):95–115, 1907.

[11] Jianda Chen, Shangyu Chen, and Sinno Jialin Pan. Storage efficient and dynamic flexible runtime channel pruning via deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[12] Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *International conference on machine learning*, pages 2285–2294, 2015.

[13] Wenlin Chen, James T. Wilson, Stephen Tyree, Kilian Q. Weinberger, and Yixin Chen. Compressing convolutional neural networks. *CoRR*, abs/1506.04449, 2015.

[14] Code. All resulted pruned models presented in this paper, 2022. The authors commit to publish upon acceptance of this paper or reviewer request.

[15] Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. *Journal of the ACM (JACM)*, 68(1):1–39, 2021.

[16] Michael B Cohen and Richard Peng. Lp row sampling by lewis weights. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 183–192, 2015.

[17] Anirban Dasgupta, Petros Drineas, Boulos Harb, Ravi Kumar, and Michael W Mahoney. Sampling algorithms and coresets for \ell_p regression. *SIAM Journal on Computing*, 38(5):2060–2078, 2009.

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[19] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in neural information processing systems*, pages 1269–1277, 2014.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[21] Xin Dong, Shangyu Chen, and Sinno Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In *Advances in Neural Information Processing Systems*, pages 4860–4874, 2017.

[22] Xuanyi Dong, Junshi Huang, Yi Yang, and Shuicheng Yan. More is less: A more complicated network with less inference complexity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5840–5848, 2017.

[23] Dan Feldman. Core-sets: An updated survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, https://arxiv.org/abs/2011.09384*, 10(1):e1335, 2020.

[24] Dan Feldman, Matthew Faulkner, and Andreas Krause. Scalable training of mixture models via coresets. In *Advances in neural information processing systems*, pages 2142–2150, 2011.

[25] Dan Feldman, Morteza Monemizadeh, Christian Sohler, and David P Woodruff. Coresets and sketches for high dimensional subspace approximation problems. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 630–649. Society for Industrial and Applied Mathematics, 2010.

[26] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1434–1453. SIAM, 2013.

[27] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.

[28] Noah Gamboa, Kais Kudrolli, Anand Dhoot, and Ardavan Pedram. Campfire: Compressible, regularization-free, structured sparse training for hardware accelerators. *arXiv preprint arXiv:2001.03253*, 2020.

[29] Lei Gu. A coreset-based semi-supverised clustering using one-class support vector machines. In *Control Engineering and Communication Technology (ICCECT), 2012 International Conference on*, pages 52–55. IEEE, 2012.

[30] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances In Neural Information Processing Systems*, pages 1379–1387, 2016.

[31] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. *CoRR*, abs/1510.00149, 2015.

[32] Sariel Har-Peled, Dan Roth, and Dav Zimak. Maximum margin coresets for active and noise tolerant learning. In *IJCAI*, pages 836–841, 2007.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[34] Yang He, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yi Yang. Soft filter pruning for accelerating deep convolutional neural networks.

[35] Yang He, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yi Yang. Soft filter pruning for accelerating deep convolutional neural networks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 2234–2240. AAAI Press, 2018.

[36] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2019.

[37] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1389–1397, 2017.

[38] Jonathan Huggins, Trevor Campbell, and Tamara Broderick. Coresets for scalable bayesian logistic regression. *Advances in Neural Information Processing Systems*, 29:4080–4088, 2016.

[39] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[40] Yani Ioannou, Duncan Robertson, Jamie Shotton, Roberto Cipolla, and Antonio Criminisi. Training cnns with low-rank filters for efficient image classification. *arXiv preprint arXiv:1511.06744*, 2015.

[41] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Speeding up convolutional neural networks with low rank expansions. In *Proceedings of the British Machine Vision Conference. BMVA Press*, 2014.

[42] Fritz John. Extremum problems with inequalities as subsidiary conditions. In *Traces and emergence of nonlinear programming*, pages 197–215. Springer, 2014.

[43] Ibrahim Jubran, Alaa Maalouf, and Dan Feldman. Introduction to coresets: Accurate coresets. *arXiv preprint arXiv:1910.08707*, 2019.

[44] Ibrahim Jubran, Ernesto Evgeniy Sanches Shayda, Ilan Newman, and Dan Feldman. Coresets for decision trees of signals. *Advances in Neural Information Processing Systems*, 34, 2021.

[45] Ibrahim Jubran, Murad Tukan, Alaa Maalouf, and Dan Feldman. Sets clustering. In *International Conference on Machine Learning*, pages 4994–5005. PMLR, 2020.

[46] Minsoo Kang and Bohyung Han. Operation-aware soft channel pruning using differentiable masks. In *International Conference on Machine Learning*, pages 5122–5131. PMLR, 2020.

[47] Zohar Karnin and Edo Liberty. Discrepancy, coresets, and sketches in machine learning. In *Conference on Learning Theory*, pages 1975–1993. PMLR, 2019.

[48] Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. *arXiv preprint arXiv:1511.06530*, 2015.

[49] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[50] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[51] Aditya Kusupati, Vivek Ramanujan, Raghav Somani, Mitchell Wortsman, Prateek Jain, Sham Kakade, and Ali Farhadi. Soft threshold weight reparameterization for learnable sparsity. *arXiv preprint arXiv:2002.03231*, 2020.

[52] Michael Langberg and Leonard J Schulman. Universal ε-approximators for integrals. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 598–607. SIAM, 2010.

[53] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan V. Oseledets, and Victor S. Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. In *ICLR (Poster)*, 2015.

[54] Vadim Lebedev and Victor Lempitsky. Fast convnets using group-wise brain damage. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 2554–2564. IEEE, 2016.

[55] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[56] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

[57] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019.

[58] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

[59] Yawei Li, Shuhang Gu, Luc Van Gool, and Radu Timofte. Learning filter basis for convolutional neural network compression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5623–5632, 2019.

[60] Lucas Liebenwein, Cenk Baykal, Harry Lang, Dan Feldman, and Daniela Rus. Provable filter pruning for efficient neural networks. In *International Conference on Learning Representations*, 2020.

[61] Lucas Liebenwein, Alaa Maalouf, Dan Feldman, and Daniela Rus. Compressing neural networks: Towards determining the optimal layer-wise decomposition. *Advances in Neural Information Processing Systems*, 34:5328–5344, 2021.

[62] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. In *Advances in Neural Information Processing Systems*, pages 2178–2188, 2017.

[63] Tao Lin, Sebastian U. Stich, Luis Barba, Daniil Dmitriev, and Martin Jaggi. Dynamic model pruning with feedback. In *International Conference on Learning Representations*, 2020.

[64] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3296–3305, 2019.

[65] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019.

[66] Mario Lucic, Olivier Bachem, and Andreas Krause. Strong coresets for hard and soft bregman clustering with applications to exponential family mixtures. In Arthur Gretton and Christian C. Robert, editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 1–9, Cadiz, Spain, 09–11 May 2016. PMLR.

[67] Jian-Hao Luo and Jianxin Wu. Autopruner: An end-to-end trainable filter pruning method for efficient deep model inference. *arXiv preprint arXiv:1805.08941*, 2018.

[68] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[69] Alaa Maalouf, Gilad Eini, Ben Mussay, Dan Feldman, and Margarita Osadchy. A unified approach to coreset learning. *arXiv preprint arXiv:2111.03044*, 2021.

[70] Alaa Maalouf, Ibrahim Jubran, and Dan Feldman. Fast and accurate least-mean-squares solvers. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 8307–8318, 2019.

[71] Alaa Maalouf, Ibrahim Jubran, and Dan Feldman. Introduction to coresets: Approximated mean. *arXiv preprint arXiv:2111.03046*, 2021.

[72] Alaa Maalouf, Ibrahim Jubran, and Danny Feldman. Fast and accurate least-mean-squares solvers for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[73] Alaa Maalouf, Ibrahim Jubran, Murad Tukan, and Dan Feldman. Coresets for the average case error for finite query sets. *Sensors*, 21(19):6689, 2021.

[74] Alaa Maalouf, Harry Lang, Daniela Rus, and Dan Feldman. Deep learning meets projective clustering. In *International Conference on Learning Representations*, 2020.

[75] Alaa Maalouf, Murad Tukan, Eric Price, Daniel M Kane, and Dan Feldman. Coresets for data discretization and sine wave fitting. In *International Conference on Artificial Intelligence and Statistics*, pages 10622–10639. PMLR, 2022.

[76] Tung Mai, Anup B Rao, and Cameron Musco. Coresets for classification–simplified and strengthened. *arXiv preprint arXiv:2106.04254*, 2021.

[77] Zelda Mariet and Suvrit Sra. Diversity networks: Neural network compression using determinantal point processes. *arXiv preprint arXiv:1511.05077*, 2015.

[78] Baharan Mirzasoleiman, Kaidi Cao, and Jure Leskovec. Coresets for robust training of deep neural networks against noisy labels. *Advances in Neural Information Processing Systems*, 33, 2020.

[79] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11264–11272, 2019.

[80] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

[81] Alexander Munteanu, Chris Schwiegelshohn, Christian Sohler, and David P Woodruff. On coresets for logistic regression. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 6562–6571, 2018.

[82] Ben Mussay, Dan Feldman, Samson Zhou, Vladimir Braverman, and Margarita Osadchy. Data-independent structured pruning of neural networks via coresets. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[83] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[84] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

[85] Hanyu Peng, Jiaxiang Wu, Shifeng Chen, and Junzhou Huang. Collaborative channel pruning for deep networks. In *International Conference on Machine Learning*, pages 5113–5122. PMLR, 2019.

[86] Jeff M Phillips. Coresets and sketches. *arXiv preprint arXiv:1601.00617*, 2016.

[87] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

[88] Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing fine-tuning and rewinding in neural network pruning. In *International Conference on Learning Representations*, 2020.

[89] Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 143–152. IEEE, 2006.

[90] Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair coresets and streaming algorithms for fair k-means. In *International Workshop on Approximation and Online Algorithms*, pages 232–251. Springer, 2019.

[91] Qinfeng Shi, James Petterson, Gideon Dror, John Langford, Alex Smola, and SVN Vishwanathan. Hash kernels for structured data. *Journal of Machine Learning Research*, 10(Nov):2615–2637, 2009.

[92] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[93] Sidak Pal Singh and Dan Alistarh. Woodfisher: Efficient second-order approximations for model compression. *arXiv preprint arXiv:2004.14340*, 2020.

[94] Christian Sohler and David P Woodruff. Subspace embeddings for the l1-norm with applications. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 755–764, 2011.

[95] Ernst Steinitz. Bedingt konvergente reihen und konvexe systeme. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1913(143):128–176, 1913.

[96] Cheng Tai, Tong Xiao, Yi Zhang, Xiaogang Wang, et al. Convolutional neural networks with low-rank regularization. *arXiv preprint arXiv:1511.06067*, 2015.

[97] Hidenori Tanaka, Daniel Kunin, Daniel LK Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *arXiv preprint arXiv:2006.05467*, 2020.

[98] Michael J Todd and E Alper Yıldırım. On khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids. *Discrete Applied Mathematics*, 155(13):1731–1744, 2007.

[99] Ivor W Tsang, James T Kwok, and Pak-Ming Cheung. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, 6(Apr):363–392, 2005.

[100] Ivor W Tsang, James Tin-Yau Kwok, and Pak-Ming Cheung. Very large svm training using core vector machines. In *AISTATS*, 2005.

[101] IW-H Tsang, JT-Y Kwok, and Jacek M Zurada. Generalized core vector machines. *IEEE Transactions on Neural Networks*, 17(5):1126–1140, 2006.

[102] Murad Tukan, Cenk Baykal, Dan Feldman, and Daniela Rus. On coresets for support vector machines. *Theoretical Computer Science*, 2021.

[103] Murad Tukan, Alaa Maalouf, and Dan Feldman. Coresets for near-convex functions. *Advances in Neural Information Processing Systems*, 33:997–1009, 2020.

[104] Murad Tukan, Alaa Maalouf, Dan Feldman, and Roi Poranne. Obstacle aware sampling for path planning. *arXiv preprint arXiv:2203.04075*, 2022.

[105] Murad Tukan, Alaa Maalouf, Matan Weksler, and Dan Feldman. No fine-tuning, no cry: Robust svd for compressing deep networks. *Sensors*, 21(16):5599, 2021.

[106] Murad Tukan, Xuan Wu, Samson Zhou, Vladimir Braverman, and Dan Feldman. New coresets for projective clustering and applications. In *International Conference on Artificial Intelligence and Statistics*, pages 5391–5415. PMLR, 2022.

[107] Karen Ullrich, Edward Meeds, and Max Welling. Soft weight-sharing for neural network compression. *arXiv preprint arXiv:1702.04008*, 2017.

[108] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[109] Kasturi Varadarajan and Xin Xiao. A near-linear algorithm for projective clustering integer points. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1329–1342. SIAM, 2012.

[110] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*, 2020.

[111] Yulong Wang, Xiaolu Zhang, Lingxi Xie, Jun Zhou, Hang Su, Bo Zhang, and Xiaolin Hu. Pruning from scratch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12273–12280, 2020.

[112] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 1113–1120, 2009.

[113] Jianbo Ye, Xin Lu, Zhe Lin, and James Z. Wang. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. In *International Conference on Learning Representations*, 2018.

[114] Mao Ye, Chengyue Gong, Lizhen Nie, Denny Zhou, Adam Klivans, and Qiang Liu. Good subnetworks provably exist: Pruning via greedy forward selection. In *International Conference on Machine Learning*, pages 10820–10830. PMLR, 2020.

[115] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. *arXiv preprint arXiv:1812.08928*, 2018.

[116] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. Nisp: Pruning networks using neuron importance score propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9194–9203, 2018.

[117] Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7370–7379, 2017.

[118] Michael Zhu and Suyog Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See section 5
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 2
   (b) Did you include complete proofs of all theoretical results? [Yes] See Sections 2.4 and C
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes]

   (b) Did you mention the license of the assets? [Yes]

   (c) Did you include any new assets either in the supplemental material or as a URL? [No]

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Computing the Carathéodory set

Overview of Algorithm 4. First, a convex combination of v with respect to P is formulated as a linear programming problem. This is done by reformulating the input set of points P as a matrix denoted as $A \in \mathbb{R}^{(d+1)\times n}$ (see Line 1). We then formulate the goal vector $b \in \mathbb{R}^{d+1}$ to be v concatenated with an entry of 1 which serves to make sure that the solution to

$$\begin{aligned} \underset{x\in\mathbb{R}^{n+1}}{\text{minimize}} \quad & \mathbf{1}_{n+1}^T x \\ \text{subject to} \quad & Ax = b, \\ & x_i \in [0,1] \quad \forall i \in [d] \end{aligned}$$

satisfies that $\sum\limits_{i\in[n]} x_i P_i = v$ and $\sum\limits_{i\in[n]} x_i = 1$. Solving this problem takes roughly $O^*\left(n^{\omega+o(1)}\right)$ where ω is the matrix multiplication exponent as elaborated in [15]; see Lines 2–3. We observe that x from Line 3 might be dense, i.e., the number of non-zero entries exceeds $d+1$. To ensure that we have at max $d+1$ non-zero entries, we use Algorithm 1 of [70] which aims to find a set of $d+1$ points, where their weighted average is the desired v given the initial weight vector x; see Line 4.

Algorithm 3: CARATHEODORY-SET (v, P)

```
Input   : A point $v \in \mathbb{R}^d$ and a set $P \subseteq \mathbb{R}^d$ of $n$ points
Output  : A subset $C \subseteq P$ of at max $d+1$ points such that $p \in \text{Conv}(C)$
1 $A := \left[\begin{bmatrix}P_1\\1\end{bmatrix}, \begin{bmatrix}P_2\\1\end{bmatrix}, \ldots, \begin{bmatrix}P_n\\1\end{bmatrix}\right]$ /* $P_i$ here denotes the $i$th point in $P$ */
2 $b := \begin{bmatrix}v\\1\end{bmatrix}$
3 $x := \underset{\substack{x\in[0,\infty)^d\\Ax=b}}{\arg\min}\ \mathbf{1}_d^T x$ // $\mathbf{1}_d$ denotes a $d$ dimensional vector of 1s
4 $C :=$ FAST-CARATHEODORY-SET $\left(P, x, d^2+2\right)$ /* See Algorithm 1 of [70] */
5 return $C$
```

B Coreset-Related Technical Details

Definition B.1 (VC-dimension [9]). For a query space $(P, w, \mathbb{R}^d, f)$ and $r \in [0,\infty)$, we define

$$\text{ranges}(x, r) = \{p \in P \mid w(p)f(p,x) \leq r\},$$

for every $x \in \mathbb{R}^d$ and $r \geq 0$. The dimension of $(P, w, \mathbb{R}^d, f)$ is the size $|S|$ of the largest subset $S \subset P$ such that

$$\left|\left\{S \cap \text{ranges}(x,r) \mid x \in \mathbb{R}^d, r \geq 0\right\}\right| = 2^{|S|},$$

where $|A|$ denotes the number of points in A for every $A \subseteq \mathbb{R}^d$.

B.1 Sensitivity Sampling Missing Details

We want to use the sensitivity sampling framework to compute a coreset for a set of points P in $\mathbb{R}^d$.

First, we need to bound the sensitivity of each point $p \in P$. The sensitivity pf a point $p \in P$ is defined as $s(p) = \sup_{x\in X} \frac{\phi(p,x)}{\sum_{q\in P}\phi(q,x)}$ where the denominator is not zero.

Hence, for every $p \in P$, we wish to compute a number $s'(p)$, such that $s'(p) \geq s(p)$. Once the bound $s'(p)$ on the sensitivity $s(p)$ of each point p is computed, we define $T = \sum_{p\in P} s'(p)$ as the total sensitivity. Now, to obtain a coreset, we can sample points according to the distribution $s'(p)/T$, i.e., we sample $m > 0$ points from P, where at each sample, the point $p \in P$ is sampled i.i.d with probability $s'(p)/T$. We also re-weight the sampled points to obtain a coreset.

As the bound $s'(p)$ (on $s(p)$) is tighter, the total sensitivity T gets smaller, and then the coreset size (required number of sampled points) gets smaller, and vice versa.

Theorem B.2 (Restatement of Theorem 5.5 in [9]). *Let $P \subseteq \mathbb{R}^d$ be a set of n points, $w : P \to [0, \infty)$ be a weight function , and let $f : P \times \mathbb{R}^d \to [0, \infty)$ be a loss function. For every $p \in P$ define the* sensitivity *of p as*

$$\sup_{x \in \mathbb{R}^d} \frac{w(p)f(p,x)}{\sum_{q \in P} w(q)f(q,x)},$$

where the sup is over every $x \in \mathbb{R}^d$ such that the denominator is non-zero. Let $s : P \to [0, 1]$ be a function such that $s(p)$ is an upper bound on the sensitivity of p. Let $t = \sum_{p \in P} s(p)$ and d' be the VC dimension *of the quadruple $\left(P, w, \mathbb{R}^d, f\right)$; see Definition B.1. Let $c \geq 1$ be a sufficiently large constant, $\varepsilon, \delta \in (0, 1)$, and let S be a random sample of $|S| \geq \frac{ct}{\varepsilon^2}\left(d' \log t + \log \frac{1}{\delta}\right)$ i.i.d points from P, such that every $p \in P$ is sampled with probability $\frac{s(p)}{t}$. Let $v(p) = \frac{tw(p)}{s(p)|S|}$ for every $p \in S$. Then, with probability at least $1 - \delta$, (S, v) is an ε-coreset for $\left(P, w, \mathbb{R}^d, f\right)$.*

B.2 From Coresets to Approximating the Optimal Solution

In optimization problems (or machine learning in general), the goal is usually to find a query that minimizes (or maximizes) some cost function. In the context of coresets, the goal is to find a small weighted subset such that for a given cost function, the cost of applying any solution (hypotheses/query) on the coreset approximates the cost of applying the same solution on the whole data. Since a coreset approximates the cost of every query, we do note that in many cases, coresets are applied for approximating the optimal solution. Specifically, solving the desired optimization problem on the whole data can be a hard problem when the time needed for such a solution is either polynomial or exponential in the size of the whole data, or when the required memory is too high. In this case, coresets can be leveraged, by computing the the optimal solution of fitting an ε-coreset and applying it on the original data. If the computed coresets gives worst-case $(1 + \varepsilon)$-approximation error, then we provably $(1 + 4\varepsilon)$-approximation towards the optimal cost of solving the optimization on the whole data (the proof is very easy, it is done by applying the triangle inequality few times). In other words, we can solve the problem on the coreset to obtain a solution x^*, and then apply x^* to the whole data giving a good approximation for solving the problem from the beginning on the whole data.

C Proofs of Technical Results

C.1 Proof of Lemma 2.4

Lemma C.1. *Let $d, \ell, m \geq 1$ be integers. Let $p \in \mathbb{R}^d$ and $A \subseteq \mathbb{R}^d$ be a set of m points with $p \in \mathrm{Conv}(A)$ so that there exists $\alpha : A \to [0, 1]$ such that $\sum_{q \in A} \alpha(q) = 1$ and $\sum_{q \in A} \alpha(q) \cdot q = p$. Then for every $Y \in \mathbb{R}^{d \times \ell}$ and $v \in \mathbb{R}^\ell$, $\|(p - v)Y\|_1 \leq \max_{q \in A} \|(q - v)\, Y\|_1$.*

Proof. Since we can write p as the convex combination of points $q \in A$ with weight $\alpha(q)$, we have

$$\left\|p^T Y - v\right\|_1 = \left\|\left(\sum_{q \in A} \alpha(q) q^T\right) Y - v\right\|_1 .$$

Moreover, we have $\sum_{q \in A} \alpha(q) = 1$, so we can decompose v into

$$\left\|p^T Y - v\right\|_1 = \left\|\sum_{q \in A} \alpha(q) \left(q^T Y - v\right)\right\|_1 .$$

By triangle inequality (or Jensen's inequality),

$$\left\|p^T Y - v\right\|_1 \leq \sum_{q \in A} \alpha(q) \left\|q^T Y - v\right\|_1 \leq \max_{q \in A} \left\|q^T Y - v\right\|_1 .$$

□

C.2 Proof of Lemma 2.5

Lemma C.2 (ℓ_∞-coreset for ℓ_1-regression). *Let $P \subseteq \mathbb{R}^d$ be a set of points, and r be the rank of P. Let $j \in [d-1]$ and let S be the output of a call to ℓ_∞-*CORESET*(P). Then (i) $|S| \in O\left(r^2\right)$, and (ii) for every $X \in \mathbb{R}^{d \times j}$ and $v \in \mathbb{R}^d$, $\frac{\max_{q \in P} \|(q-v)X\|_1}{\max_{q \in S} \|(q-v)X\|_1} \in \left[1, 2r^{1.5}\right]$.*

Proof. Let Y, z, P', K, S and map be defined as in Algorithm 1. Since P lies on a r-dimensional affine subspace, it holds that for every $p \in P$, $p = (p-z)YY^T + z$. Note that $Y \in \mathbb{R}^{d \times r}$ is an orthogonal matrix (i.e., Y^TY is the identity matrix in $\mathbb{R}^{r \times r}$) and $z \in \mathbb{R}^d$ denotes the translation of the affine subspace that P lies on.

Claim (i). $E(G,c)$ is the Löwner ellipsoid of P', which has $2r$ vertices. Since V is the set of vertices of the shrunk form of $E(G,c)$ that is contained in $\mathrm{Conv}\left(P'\right)$, each point from V can be represented as convex combination of $r+1$ points from P' by Carathéodory's Theorem. Then the number of points in K is at most $2r\left(r+1\right)$, i.e., $|K| \in O\left(r^2\right)$. Thus, $|S| \in O\left(r^2\right)$ due to the fact that it can be constructed from K through the use of map.

Claim (ii). First put $X \in \mathbb{R}^{d \times j}$ and $v \in \mathbb{R}^d$, and let $p \in \arg\sup\limits_{q \in P} \|(p-v)X\|_1$. Since $S \subseteq P$, it holds that $\frac{\max_{q \in P} \|(q-v)X\|_1}{\max_{q \in S} \|(q-v)X\|_1} \geq 1$. Let $a := zYY^T + z - v$, we have

$$\|(p-v)X\|_1 = \left\|\left((p-z)YY^T + z - v\right)X\right\|_1^T = \left\|\left(pYY^T + a\right)X\right\|_1 = \left\|\left(p'Y^T + a\right)X\right\|_1, \tag{1}$$

where the first equality holds since $\mathrm{rank}(P) = r$, the second holds by definition of a, and the last equality holds by the construction of $p' = pY$ at Line 2 of Algorithm 1.

Note that since V is the set of vertices of $\frac{1}{r}\left(E(G,c) - c\right) + c$, by the definition of the Löwner ellipsoid,

$$V \subseteq \mathrm{Conv}\left(V\right) \subseteq \frac{1}{r}(E(G,c) - c) + c \subseteq \mathrm{Conv}\left(P'\right) \subseteq E(G,c) \subseteq \mathrm{Conv}\left(r^{1.5}\left(V - c\right) + c\right).$$

Since $\mathrm{Conv}\left(P'\right)$ enclose $\mathrm{Conv}\left(V\right)$, and is enclosed by $\mathrm{Conv}\left(r^{1.5}\left(V-c\right)+c\right)$, then there exists a point $q \in \mathrm{Conv}\left(V\right)$ and $\gamma \in [0,1]$ such that $p'Y^T = \gamma qY^T + (1-\gamma)\left(r^{1.5}\left(q-c\right)+c\right)Y^T$, where by definition it holds that $r^{1.5}\left(q-c\right)+c \in \mathrm{Conv}\left(r^{1.5}\left(V-c\right)+c\right)$, and $p' = pY$.

By invoking Lemma 2.4, we obtain that

$$\left\|\left(p'Y^T + a\right)X\right\|_1 \leq \max\left\{\left\|\left(qY^T + a\right)X\right\|_1, \left\|\left(r^{1.5}\left(q-c\right)+c+a\right)Y^TX\right\|_1\right\}. \tag{2}$$

We note that

$$\left\|\left(qY^T + a\right)X\right\|_1 \leq \max_{\tilde{q} \in V}\left\|\left(\tilde{q}Y^T + a\right)X\right\|_1 \leq \max_{\tilde{q} \in K}\left\|\left(\tilde{q}Y^T + a\right)X\right\|_1, \tag{3}$$

where the first inequality follows from plugging $p := q$ and $A := V$ into Lemma 2.4, and the second inequality holds similarly since every point V lies in $\mathrm{Conv}\left(K\right)$. By invoking triangle inequality, we obtain that

$$\begin{aligned}
&\left\|\left(r^{1.5}\left(\left(q-c\right)+c\right)Y^T + a\right)X\right\|_1 = \left\|\left(r^{1.5}qY^T + \left(1 - r^{1.5}\right)cY^T + a\right)X\right\|_1 \qquad (4)\\
&= \left\|\left(r^{1.5}qY^T + r^{1.5}a + \left(1-r^{1.5}\right)cY^T + \left(1-r^{1.5}\right)a\right)X\right\|_1\\
&\leq r^{1.5}\left\|\left(qY^T + a\right)X\right\|_1 + \left(r^{1.5} - 1\right)\left\|\left(cY^T + a\right)X\right\|_1,
\end{aligned}$$

where the first equality follows by a simple rearrangement, and the second holds since $r^{1.5}a + \left(1 - r^{1.5}\right)a = a$. Observe that $c \in \mathrm{Conv}\left(V\right)$. Hence, by Lemma 2.4,

$$\left\|\left(cY^T + a\right)X\right\|_1 \leq \max_{\tilde{q} \in K}\left\|\left(\tilde{q}Y^T + a\right)X\right\|_1. \tag{5}$$

By the construction of S, it holds that for every $p \in S$, $pY \in K$. Thus, combining (2), (3), (4) and (5) yields $\frac{1}{2r^{1.5}}\left\|\left(p'Y^T + a\right)X\right\|_1 \leq \max_{\tilde{q} \in K}\left\|\left(\tilde{q}Y^T + a\right)X\right\|_1 = \max_{\tilde{q} \in S}\left\|\left(\tilde{q}YY^T + a\right)X\right\|_1 = \max_{\tilde{q} \in S}\left\|\left(\tilde{q} - v\right)X\right\|_1$ where the last equality holds by (1). This concludes Lemma 2.5. □

C.3 Proof of Theorem 2.6

Proof. For space constraints let ϕ denote the ReLU function. To obtain a coreset, we first need to bound the sensitivity of each $p \in P$. Put $p \in P$ and let $x \in \arg\sup\limits_{x' \in \mathbb{R}^d} \frac{\phi\left(p^T x'\right)}{\sum_{q \in P} \phi(q^T x')}$ where the supremum is over every $x' \in \mathbb{R}^d$ such that the denominator is not zero. Observe that

$$\frac{\sum\limits_{q \in P} \left|q^T x\right|}{\sum\limits_{q \in P} \phi\left(q^T x\right)} = \frac{\sum\limits_{q \in P} \phi\left(q^T x\right) + \sum\limits_{q \in P} \phi\left(-q^T x\right)}{\sum\limits_{q \in P} \phi\left(q^T x\right)} = 1 + \frac{\sum\limits_{q \in P} \phi\left(-q^T x\right)}{\sum\limits_{q \in P} \phi\left(q^T x\right)} \leq 1 + \mu\left(P\right),$$

where the last inequality follows from Definition 2.3. Thus

$$\sum_{q \in P} \phi\left(q^T x\right) \geq \frac{1}{1 + \mu\left(P\right)} \sum_{q \in P} \left|q^T x\right|. \tag{6}$$

Let $\beta = \left(1 + \mu\left(P\right)\right)$. We next observe that $\frac{\phi\left(p^T x\right)}{\sum_{q \in P} \phi(q^T x)} \leq \frac{\left|p^T x\right|}{\sum_{q \in P} \phi(q^T x)} \leq \beta \frac{\left|p^T x\right|}{\sum_{q \in P} \left|q^T x\right|}$, where the first inequality holds by properties of ϕ, and the second is by (6). Hence the sensitivity of p is bounded by

$$s(p) = \frac{\phi\left(p^T x\right)}{\sum_{q \in P} \phi\left(q^T x\right)} \leq \beta \frac{\left|p^T x\right|}{\sum_{q \in P} \left|q^T x\right|}. \tag{7}$$

Let i be the iteration counter from Algorithm 2 as defined in Line 1, used in Line 5 and incremented in Line 7. The idea follows that of [109] where points being discarded from P at lower levels (smaller i's) have higher sensitivity. This notion also resembles that of the "Onion sampling" of [45]. Now, assume that $p \in S_i$ at iteration i of the while loop (Line 3 of Algorithm 2). In this case, observe that by plugging $P := Q \setminus \bigcup_{\hat{i}=1}^{i-1} S_{\hat{i}}$, $j = 1$ and $v = -b \frac{x}{\|x\|_2}$ into Lemma 2.5, we obtain a subset $S_i \subseteq Q$ such that

$$\max_{q \in P} \left|q^T x\right| \leq \max_{q \in S} 2r^{1.5} \left|q^T x\right|. \tag{8}$$

Thus for every $q \in S_i$,

$$\frac{s(p)}{\left(1 + \mu\left(P\right)\right)} \leq \frac{\left|p^T x\right|}{\sum_{q \in P} \left|q^T x\right|} \leq \frac{\left|p^T x\right|}{\sum_{\hat{i}=1}^{i} \max_{q \in S_{\hat{i}}} \left|q^T x\right|} \leq 2r^{1.5} \frac{\left|p^T x\right|}{\sum_{\hat{i}=1}^{i} \left|p^T x\right|} = \frac{2r^{1.5}}{i},$$

where the first inequality is by (7), the second inequality follows from the observation that $\left\{\arg\max_{q \in S_{\hat{i}}} \left|q^T x\right|\right\}_{\hat{i}=1}^{i} \subseteq P$ and the last inequality holds by (8). Hence, we have obtained a bound on the sensitivity of each point $p \in P$. As for the total sensitivity, we observe that $t = \sum_{p \in P} s(p) \in O\left(\left(1 + \mu\left(P\right)\right) r^{3.5} \log n\right)$. Theorem B.2 states that to obtain an ε-coreset with probability at least $1 - \delta$, the sample size m must be $O\left(\frac{\mu(P) r^{3.5} \log n}{\varepsilon^2} \left(d\left(\log\left(\mu\left(P\right) r \log n\right)\right) + \log\left(\frac{1}{\delta}\right)\right)\right)$. □

D Extension

D.1 Handling Weighted Sets of Points

Similarly to [5, 60], we split the input data P into two sets $P_+, P_- \subseteq P$ such that $P_+ = \{p \in P | w(p) \geq 0\}$ while $P_- = \{p \in P | w(p) < 0\}$. Following this step we call Algorithm 4 for each of the two sets with corresponding weights and corresponding sample sizes. To account for proper sample sizes, we split our theoretical bound of the required sample size for generating ε-coreset into two terms for both P_+ and P_- respectively, i.e., we formulate $m = m_+ + m_-$ where $m_+ = \frac{|P_-|}{|P|} m$ (similarly for m_-). Hence, we obtain an ε-coreset for each of the query spaces $\left(P_+, w, \mathbb{R}^d, \phi\right)$ and $\left(P_-, w, \mathbb{R}^d, \phi\right)$.

D.2 From Weight to Neuron Pruning

Most coreset-based pruning methods, e.g., [5, 82], first provide a scheme for (provable) weight pruning, which is then used as a stepping stone towards pruning neurons as follows. To prune neurons from a layer, post to computing the coreset-based weight pruning for each neuron, ideally we would have that at certain layer, for all neurons, the generated coreset contains the same set of neurons from previous layers, which in this case we can remove the neurons which are not in the coreset. However, such scenario is almost implausible. To deal with such problem, we discuss two ways to do so. The first method to deal with such problem is inspired by the technique used in [82] which alters the definition of sensitivity such that it takes into account the sensitivity of a neuron in a layer ℓ with respect to all the neurons in the layer $\ell + 1$, basically the sensitivity of each neuron is taken be the maximal sensitivity over every weight function (neuron in the next layer) defined by the layer. Hence, we follow the same logic for such method, more details .

Algorithm 4: GENERALIZED-CORESET (P, w, m)

```
input  : A set $P \subseteq \mathbb{R}^d$ of $n$ points, a weight function $w(p) : P \to [0, \infty)$ and a sample size $m$
output : A weighted set $(C, u)$
1  $Q := P, i := 1, C := \emptyset$
2  while $|Q| \geq 2\text{rank}(Q)^2$ do
3      $Q' := \{w(q)q | q \in Q\}$
4      $\text{map}_w : Q' \to Q$ a map that maps from $Q'$ to $Q$
5      $S_i := \ell_\infty$-CORESET $(Q')$
6      for every $p \in S_i$ do
7          $s(\text{map}_w(p)) := \frac{2r^{1.5}}{i}$
8      end
9      $Q := Q \setminus \{\text{map}_w(q) | q \in S_i\}, i := i + 1$
10 end
11 for every $p \in Q$ do
12     $s(p) := \frac{2r^{1.5}}{i}$
13 end
14 $t := \sum_{p \in P} s(p)$
15 $C :=$ an i.i.d sample of $m$ points from $P$, where each $p \in P$ is sampled with probability $\frac{s(p)}{t}$.
16 $u(p) := \frac{tw(p)}{m \cdot s(p)}$ for every $p \in C$
17 return $(C, u)$
```

D.3 Other Activation Functions

[76] recently showed that there exists a family of functions $\mathcal{F}$ called "Nice hinge functions" such that for any query $x \in \mathbb{R}^d$ and a set of points $P \subseteq \mathbb{R}^d$, for any $\phi \in \mathcal{F}$, it holds that $\frac{\sum\limits_{p \in P} \phi(p^T x)}{\sum\limits_{q \in P} \text{ReLU}(q^T x)}$ is bounded from below. Formally speaking, below is the definition of a "nice hinge function".

Definition D.1 (Restatement of Definition 7 of [76]). We call $f : \mathbb{R} \to [0, \infty)$ an (L, a_1, a_2)-nice hinge function if for a fixed constant L, a_1 and a_2,

(i) f is L-Lipschitz,

(ii) $|f(z) - \text{ReLU}(z)| \leq a_1$ for all z, and

(iii) $f(z) \geq a_2$ for all $z \geq 0$.

As noted by [76], the hinge and log losses are $(1, 1, 1)$-nice and $(1, \ln 2, \ln 2)$-nice hinge functions respectively. Similarly, it is easy to show that the activation function $\phi(x) = \ln(1 + e^x)$ is

$(1, \ln 2, \ln 2)$-nice hinge functions. Following the same steps applied by [76], we obtain that

$$\sum_{p \in P} \phi\left(p^T x\right) \geq \min\left\{\frac{a_2}{2a_1}, \frac{1}{2}\right\} \frac{\sum\limits_{q \in P} \left|q^T x\right|}{\mu\left(P\right) + 1},$$

where $\phi\left(\cdot\right)$ is a $\left(L, a_1, a_2\right)$ where a_2 is assumed to be positive.

Unlike the ReLU activation function, to support for other activation functions, we need to restrict our query space to contain queries such that $\forall p \in P : \phi\left(p^T x\right) \leq r \left|p^T x\right|$ where r denotes the rank of P. Let X' denote the set of all such queries.

Hence, under this additional assumption, we obtain that for every $p \in P$ and $x \in X'$

$$\frac{\phi\left(p^T x\right)}{\sum\limits_{q \in P} \phi\left(q^T x\right)} \leq \frac{\left(1 + \mu\left(P\right)\right) \phi\left(p^T x\right)}{\min\left\{\frac{a_2}{2a_1}, \frac{1}{2}\right\} \sum\limits_{q \in P} \left|q^T x\right|} \leq \frac{\left(1 + \mu\left(P\right)\right) r \left|p^T x\right|}{\min\left\{\frac{a_2}{2a_1}, \frac{1}{2}\right\} \sum\limits_{q \in P} \left|q^T x\right|}.$$

Following the same steps done at the proof of Theorem 2.6, we can generate an ε-coreset with respect to $\left(P, \phi, X'\right)$.

E Implementation Details

First observe that the *map* function in Line 3 of Algorithm 1 is hard to implement if all we have is Y and P' (see Lines 1–2) due to the fact that when the rank of P is not d, then Y becomes a singular matrix. A way around such problem (practically speaking yet also theoretically sound) is to reformulate P and P' as matrices, where our ℓ_∞-coreset will now be regarded as a set of indices of the rows selected as the desired coreset. **Note** that while we relied on an "accurate" measure of the rank of points in Algorithm 1, in our experiments, we used the rank function from *Numpy*, and still produced favorable results. Furthermore, our algorithms work also when the input has full rank. In addition, we can still obtain an ε-coreset when using approximated algorithms for the rank computation problem, where the error associated with our coreset may increase. In this case, we can increase our coreset size to reduce our approximation error to be the original desired error.

F Complexity Measure - Clarification

First, Note that while the complexity measure was first defined for construction of coresets with respect to the logistic regression problem, it also has been used for the ReLU regression problem (minimizing the sum of ReLU losses) [76].

The complexity is expected to be small other than in some cases [81, 76]. We also operate under the same assumption, i.e., the complexity measure is reasonably small.

Specifically speaking, when given a set P (expressing our neurons) containing points in $\mathbb{R}^3$ (for example), such that point in P lie on a 2-dimensional affine subspace parallel to the xy-plane, notice that in our setting, the complexity measure is defined as the maximal value of an optimization problem involving our vectors and a set of queries $X := \mathbb{R}^2 \times \{1\}$.

The existing of a hyperplane whose normal in X such that one point from P can be separated from the rest of the points in P, leads to large complexity measure.

In Figure 3, a clear separation can be made between one point and the rest leading to two sets of points: The first set contains one single point that has a positive dot product with the normal x to the separating hyperplane, while the other set contains the remaining point each with negative dot product with x. This leads to a large complexity measure, and as the separating hyperplane gets closer and closer to the set containing the single point, the complexity measure increases, as it can tend to infinity.

On the other hand, when one can not separate a single point or minimal set of points from the rest of the data, we expect the complexity measure to be small; see Figure 4.

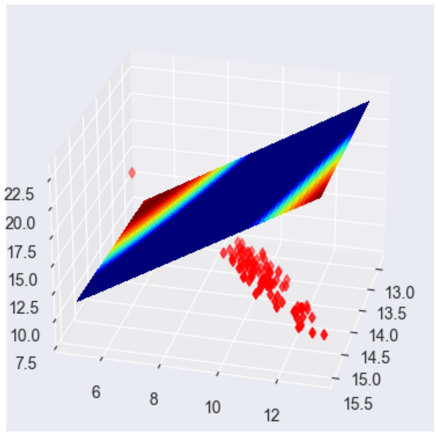


Figure 3: Linearly separable data leading to sufficiently large complexity measure.

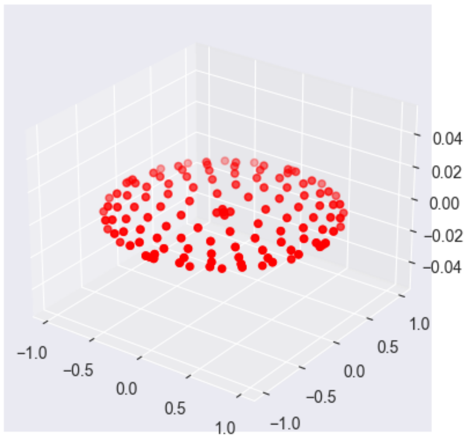


Figure 4: Non-linearly separable data leading to sufficiently small complexity measure.

Notice that in Figure 4, the input data is centered around the origin, which means that the data is not linearly separable. Thus leading to small complexity measure, since x represents the normal to a hyperplane emerging from the origin.

In fact, during our experiments, the complexity measure in the case of LeNet300-100 was around 15 when the input data (matrix representing the neurons) had 300 rows (points).

It is common in coreset literature from a practical point of view, sample sizes that are smaller than the bound on the coreset size are being used. This is due to the fact that such bounds are pessimistic in nature.

This motivated the choice of not incorporating the complexity measure in our sample size nor the sensitivity sampling since such a term will be eliminated when computing the sampling probability; see Theorem B.2. Our experiments confirmed such an observation, i.e., our coreset lead to favorable results when the complexity measure was not incorporated in our computations, or when the sample size was much smaller than the bound on the coreset size.

The appendix in the supplementary material has been modified in light of this.

G Additional Experiments

In all of our experiments, our hyper-parameters were drawn from [60].

G.1 The effect of fine-tuning

In this experiment, we aim to show the effect of fine-tuning on our compressed model. Specifically, Figure 5 shows VGG19's network accuracy over fine-tuning, where we start better than previous methods, followed by a slow incline in accuracy until we outperform previous models (around epoch 18).

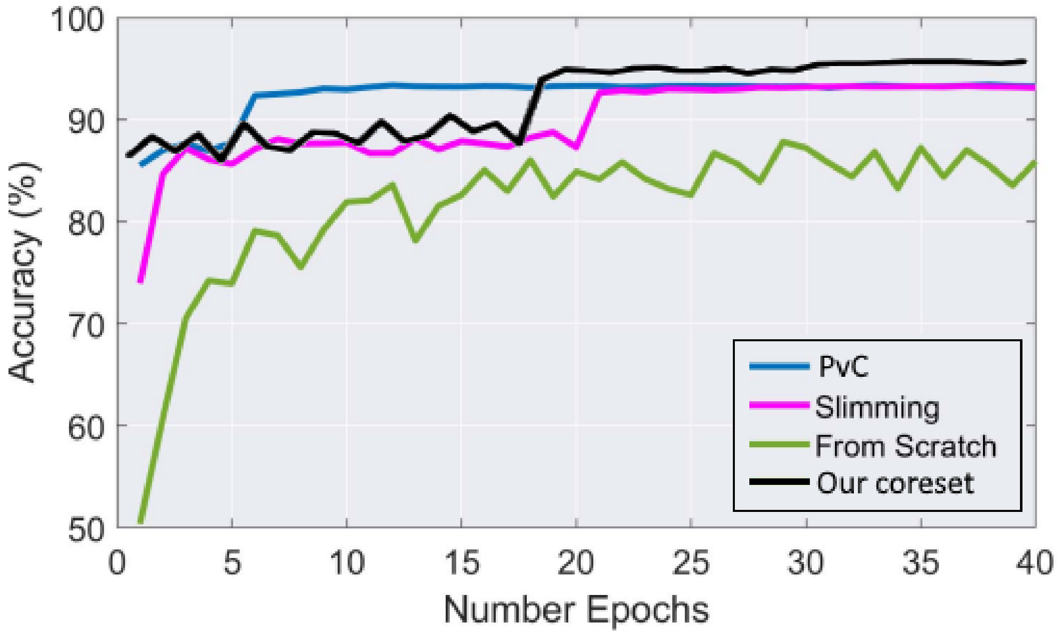


Figure 5: Accuracy of our proposed framework in comparison to previous methods and training the pruned network from scratch. The results above reflect the accuracy of VGG19 on CIFAR10.

G.2 Sensitivity based distribution

At Figure 6, we plot the sensitivity distribution of our sampling method in comparison to the sampling probabilities achieved by [82]. Our advantage lies in the observation that our induced probability distribution entails longer tails, i.e., important point are scarce.

G.3 Comparison with PvC

In this experiment, we aim to show the efficacy of our approach against that of [82]. We considered a single neuron in LeNet-300-100 where we computed the average additive error of the cost of the coreset from the cost of taking all the samples (neurons from previous layer), over set of 1000 queries. As shown in Figure 7, for very small coreset sizes, PvC [82] attains smaller error, however as we increase the sample size, our coreset outperforms that of [82].

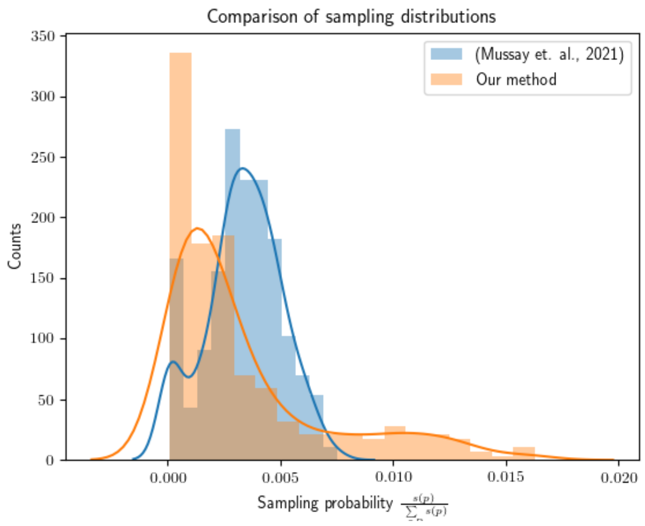


Figure 6: A comparison against [82] with respect to the distribution of the sampling probabilities of weights of a single neuron at some layer of LeNet-300-100. Here the x-axis denotes the sampling probability of points, while the y-axis presents the number of points with certain probability.

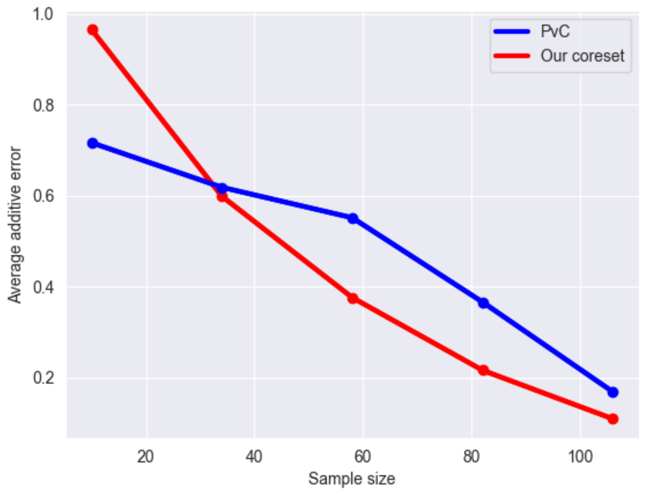


Figure 7: Average additive error of our coreset and that of [82] on LeNet-300-100.

Statement of Authorship

Principal Author

Title of Paper	Using partial monotonicity in submodular maximization.		
Publication Status	[✓ Published] ☐ Accepted for Publication ☐ Submitted for Publication		
Publication Details	*Advances in Neural Information Processing Systems*, *35*, pp.2723-2736.		
Name of Principal Author (Candidate)	Loay Mualem		
Contribution to the Paper	Contributed to the theory, implementation and writing of the paper.		
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Name and Signature	Loay Mualem [illegible]	Date	24/6/2025

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the dissertation

Name of Co-Author	Moran Feldman		
Contribution to the Paper	Contributed to the theory and writing of the paper.		
Name and Signature	Moran Feldman	Date	30.6.25

Please cut and paste additional co-author panels here as required.

Chapter 3

Using Partial Monotonicity in Submodular Maximization

In this chapter, we explore the role of partial monotonicity in submodular functions and demonstrate how it can be leveraged to achieve improved approximation guarantees in practical machine learning settings. Our work introduces a principled framework for identifying and exploiting near-monotone structure in otherwise non-monotone objectives, leading to more efficient optimization strategies.

The following paper [MF22] was published at the *Conference on Neural Information Processing Systems (NeurIPS 2022)*.

Using Partial Monotonicity in Submodular Maximization

Loay Mualem
Department of Computer Science
University of Haifa
Haifa 3303221, Israel
loaymua@gmail.com

Moran Feldman
Department of Computer Science
University of Haifa
Haifa 3303221, Israel
moranfe@cs.haifa.ac.il

Abstract

Over the last two decades, submodular function maximization has been the workhorse of many discrete optimization problems in machine learning applications. Traditionally, the study of submodular functions was based on *binary* function properties, but recent works began to consider *continuous* function properties such as the submodularity ratio and the curvature. The monotonicity property of set functions plays a central role in submodular maximization. Nevertheless, no continuous version of this property has been suggested to date (as far as we know), which is unfortunate since submoduar functions that are almost monotone often arise in machine learning applications. In this work we fill this gap by defining the *monotonicity ratio*, which is a continuous version of the monotonicity property. We then show that for many standard submodular maximization algorithms one can prove new approximation guarantees that depend on the monotonicity ratio; leading to improved approximation ratios for the common machine learning applications of movie recommendation, quadratic programming, image summarization and ride-share optimization.

1 Introduction

Over the last two decades, submodular function maximization has been the workhorse of many discrete optimization problems in machine learning applications such as data summarization [17, 19, 31, 32, 41, 50], social graph analysis [45], adversarial attacks [36], dictionary learning [15], sequence selection [42, 51], interpreting neural networks [18] and many more. Traditionally, the study of submodular functions was based on *binary* properties of functions. A function can be either submodular or non-submodular, monotone or non-monotone, etc. Such properties are simple, but they have an inherit weakness—if an algorithm assumes functions that have a particular property, then it provides no guarantee for functions that violate this property, even if the violation is slight.

Given the above situation, recent works began to consider *continuous* versions of function properties. Probably the most significant among these continuous versions so far are the submodularity ratio and the curvature. The submodularity ratio (originally defined by Das and Kempe [16]) is a parameter $\gamma \in [0, 1]$ replacing the binary submodularity property that a set function can either have or not have. A value of 1 corresponds to a fully submodular function, and lower values of γ represent some violation of submodularity (the worse the violation, the lower γ). Similarly, the curvature (defined by Conforti and Cornuéjol [13]) is a parameter $c \in [0, 1]$ replacing the binary linearity property that a set function can either have or not have. A value of 1 corresponds to a fully linear function, and lower values of c represent some violation of linearity.

A central conceptual contribution of Das and Kempe [16] was that they were able to demonstrate that continuous function properties further extend the usefulness of submodular maximization to new machine learning applications (such as subset selection for regression and dictionary selection). This has motivated a long list of works on such properties (see [3, 25, 26, 29, 34] for a few examples), including works that combine both the submodularity ratio and the curvature (see, e.g., [3]). However, to the best of our knowledge, no continuous version of the binary monotonicity property has been suggested so far.[1] See Appendix A for additional related work.

We note that the monotonicity property of set functions plays a central role in submodular maximization, and basically every problem in this field has been studied for both monotone and non-monotone objective functions. Naturally, monotone objective functions enjoy improved approximation guarantees compared to general functions, and it is natural to ask how much of this improvement applies also to functions that are almost monotone (in some sense). Since such functions often arise in machine learning applications when a diversity promoting component is added to a basic monotone objective, obtaining better guarantees for them should strongly enhance the usefulness of submodular maximization as a tool for many machine learning applications.

Formally, a non-negative set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ over a ground set $\mathcal{N}$ is (increasingly) *monotone* if $f(S) \subseteq f(T)$ for every $S \subseteq T \subseteq \mathcal{N}$. Similarly, we define the *monotonicity ratio* of such a function f as the maximum value $m \in [0, 1]$ such that $m \cdot f(S) \leq f(T)$ for every two sets $S \subseteq T \subseteq \mathcal{N}$. Equivalently, one can define the monotonicity ratio m by $m \triangleq \min_{S\subseteq T\subseteq \mathcal{N}}[f(T)/f(S)]$, where the ratio $f(T)/f(S)$ is assumed to be 1 whenever $f(S) = 0$. Intuitively, the monotonicity ratio measures how much of the value of a set S can be lost when additional elements are added to S. One can view m as the distance of f from monotonicity. In particular, $m = 1$ if and only if f is monotone.

Our main contribution in this paper is demonstrating the usefulness of the monotonicity ratio in machine learning applications, which we do in two steps.

- First, we show (in Sections 3, 4 and 5) that for many standard submodular maximization algorithms one can prove new approximation guarantees that depend on the monotonicity ratio. These approximation guarantees interpolate between the known approximation ratios of these algorithms for monotone and non-monotone submodular functions.
- Then, using the above new approximation guarantees, we derive new approximation ratios for the standard applications of movie recommendation, quadratic programming, image summarization and ride-share optimization. Our guarantees improve over the state-of-the-art for most values of the problems' parameters. See Section 6 for more detail.

Remark. Computing the monotonicity ratio m of a given function seems to be difficult. Thus, the algorithms we analyze avoid assuming access to m, and the value of m is only used in the analyses of these algorithms. Nevertheless, in the context of particular applications, we are able to bound m, and plugging this bound into our general results yields our improved guarantees for these applications.

1.1 Our Results

Given a ground set $\mathcal{N}$, a set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ is submodular if $f(S \cup \{u\}) - f(S) \geq f(T \cup \{u\}) - f(T)$ for every two sets $S \subseteq T \subseteq \mathcal{N}$ and element $u \in \mathcal{N} \setminus T$. Submodular maximization problems ask to maximize such functions subject to various constraints. To allow for multiplicative approximation guarantees for these problems, it is usually assumed that the objective function f is non-negative. Accordingly, we consider in this paper the following three basic problems.

- Given a non-negative submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$, find a set $S \subseteq \mathcal{N}$ that (approximately) maximizes f. This problem is termed "unconstrained submodular maximization", and is studied in Section 3.
- Given a non-negative submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ and an integer parameter $0 \leq k \leq |\mathcal{N}|$, find a set $S \subseteq \mathcal{N}$ of size at most k that (approximately) maximizes f among such sets. This problem is termed "maximizing a submodular function subject to a cardinality constraint", and is studied in Section 4.
- Given a non-negative submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ and a matroid $\mathcal{M}$ over the same ground set, find a set $S \subseteq \mathcal{N}$ that is independent in $\mathcal{M}$ and (approximately) maximizes f

[1]Following the appearance of the pre-print version of this paper, we learned that Iyer defined in his Ph.D. thesis [30] such a property, which is identical to the one we define. However, Iyer only used this property to prove the result appearing below as Theorem 4.1; thus, our work is the first to systematically study this property.

among such sets. This problem is termed “maximizing a submodular function subject to a matroid constraint”, and is studied in Section 5 (see Section 5 for the definition of matroids).

We present both algorithmic and inapproximability results for the above problems. Our algorithmic results reanalyze a few standard algorithms, and surprisingly show that almost all these algorithms guarantee an approximation ratio of $m \cdot \alpha_{\text{mon}} + (1 - m) \cdot \alpha_{\text{non-mon}}$, where m is the monotonicity ratio, α_{mon} is the approximation ratio known for the algorithm when f is monotone, and $\alpha_{\text{non-mon}}$ is the approximation ratio known for the algorithm when f is a general non-negative submodular function.

While the above mentioned algorithmic results lead to our improved guarantees for applications, our inapproximability results represent our main technical contribution. In general, these results are based on the symmetry gap framework of Vondrák [52]. The original version of this framework is able to deal both with the case of general (not necessarily monotone) submodular functions, and with the case of monotone submodular functions; which in our terms correspond to the cases of $m \geq 0$ and $m \geq 1$, respectively. However, to prove our inapproximability results, we had to show that the framework extends to arbitrary lower bounds on m, which was challenging because the original proof of the framework is highly based on derivatives of continuous functions. From this point of view, submodularity is defined as having non-positive second-order derivatives, and monotonicity is defined as having non-negative first-order derivatives. However, the definition of the monotonicity ratio cannot be easily restated in terms of derivatives;[2] and thus, handling it required us to come up with a different proof approach.

Interestingly, our results for unconstrained submodular maximization proves that the optimal approximation ratio for this problem does not exhibit a linear dependence on m. Thus, the nice linear dependence demonstrated by almost all our algorithmic results is probably an artifact of looking at standard algorithms rather than representing the true nature of the monotonicity ratio, and we expect future algorithms tailored to take advantage of the monotonicty ratio to improve over this linear dependence. The reason that we concentrate in this work on reanalyzing standard submodular maximization algorithms rather than inventing new ones is that we want to stress the power obtained by using the new notion of monotonicity ratio, as opposed to power gained via new algorithmic innovations. This is in line with the research history of the submodularity ratio and the curvature. For both of these parameters, the original works concentrated on reanalyzed the standard greedy algorithm in view of the new suggested parameter; and the invention of algorithms tailored to the parameter was deferred to later works (see [49] and [12] for examples of such algorithms for the curvature and submodularity ratio, respectively).

Over the years, the standard submodular maximization algorithms have been extended and improved in various ways. Some works presented accelerated and/or parallelized versions of these algorithms, while other works generalized the algorithms beyond the realm of set functions (for example, to (DR-)submodular functions over lattices or continuous domains). Since our motivation in this paper is related to the monotonicity ratio, which is essentially independent of the extensions and improvements mentioned above, we mostly analyze the vanilla versions of all the algorithms considered. This keeps our analyses relatively simple. However, our experiments are based on more state-of-the-art versions of the algorithms. Similarly, many continuous properties (including the submodularity ratio) have weak versions that only depend on the behavior of the function for nearly feasible sets, and immediately enjoy most of the results that apply to the original strong property. The definition of such weak versions is useful for capturing additional application, but often add little from a theoretical perspective. Therefore, in the theoretical parts of this paper we consider only the monotonicity ratio as it is defined above; but for the sake of one of our applications we later define also the natural corresponding weak property.

2 Preliminaries and Basic Observations

In this section we define the notation used in this paper, and then state some useful basic observations. Given an element $u \in \mathcal{N}$ and a set $S \subseteq \mathcal{N}$, we use $S + u$ and $S - u$ as shorthands for $S \cup \{u\}$ and $S \setminus \{u\}$. Additionally, given a set function $f \colon 2^{\mathcal{N}} \to \mathbb{R}$, we define $f(u \mid S) \triangleq f(S + u) - f(S)$—this value is known as the marginal contribution of u to S with respect to f. Similarly, given an additional

[2]To see why that is the case, notice that a function can have a monotonicity ratio close to 1, even in the presence of very negative derivatives, as long as these derivatives do not occur over too long sections.

set $T \subseteq \mathcal{N}$, we define $f(T \mid S) \triangleq f(S \cup T) - f(S)$. We also use $\mathbf{1}_S$ to denote the characteristic vector of the set S, i.e., a vector in $[0,1]^{\mathcal{N}}$ that has 1 in the coordinates corresponding to elements that appear in S and 0 to the other coordinates. Finally, if f is non-negative, we say that it is m-monotone if its monotonicity ratio is at least m; and given an event $\mathcal{E}$, we denote by $\mathbf{1}[\mathcal{E}]$ the indicator of this event, i.e., a random variable that takes the value 1 when the event happens, 0 otherwise .

Next, we present a well-known continuous extension of set functions. Given a set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$, its *multilinear extension* is a function $F\colon [0,1]^{\mathcal{N}} \to \mathbb{R}$ defined as follows. For every vector $\mathbf{x} \in [0,1]^{\mathcal{N}}$, let $\mathtt{R}(\mathbf{x})$ to be a random subset of $\mathcal{N}$ that includes every element $u \in \mathcal{N}$ with probability x_u, independently. Then, $F(\mathbf{x}) = \mathbb{E}[f(\mathtt{R}(\mathbf{x}))]$. Another extension of set functions is central to the proof of the next lemma, which generalizes Lemma 2.2 of [6]—see Appendix B for the proof.

Lemma 2.1. *Let $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ be a non-negative m-monotone submodular function. For every deterministic set $O \subseteq \mathcal{N}$ and random set $D \subseteq \mathcal{N}$, $\mathbb{E}[f(O \cup D)] \geq (1 - (1-m) \cdot \max_{u \in \mathcal{N}} \Pr[u \in D]) \cdot f(O)$.*

We conclude this section with the following observation, which we view as evidence that the class of non-negative m-monotone functions is a natural class for every $m \in [0,1]$.

Observation 2.2. *For every two non-negative m-monotone functions $f, g\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ and constant $c \geq 0$, the following functions are also m-monotone: (i) $h(S) = f(S) + g(S)$, (ii) $h(S) = f(S) + c$, and (iii) $h(S) = c \cdot f(S)$.*

3 Unconstrained Maximization

Recall that in the unconstrained submodular maximization problem, we are given a non-negative submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$, and the objective is to find a set $S \subseteq \mathcal{N}$ that (approximately) maximizes $f(S)$. Buchbinder et al. [7] gave the first $1/2$-approximation algorithm for this problem, known as the (randomized) double greedy algorithm. As its name suggests, double greedy maintains two solutions: one starting as the empty set, and one starting as the entire ground set. Then, it considers all elements, and greedily decides for each element either to add it to the originally empty set, or remove it from the other set. When the algorithm terminates, the two sets are identical, and their common value is the output of the algorithm. The $1/2$-approximation guarantee of double greedy is known to be optimal in general due to a matching inapproximability result due to Feige et al. [21]. Nevertheless, in this section we determine the extent to which one can improve over this guarantee as a function of the monotonicity ratio m of f.

Theorem 3.1. *The double greedy algorithm of Buchinder et al. [7] guarantees $[1/(2-m)]$-approximation for unconstrained submodular maximization, and no polynomial time algorithm obtains an approximation ratio of $1/(2-m) + \varepsilon$ for any constant $\varepsilon > 0$.*[3]

Interestingly, Theorem 3.1 shows that the optimal approximation ratio for unconstrained submodular maximization does not have a linear dependence on m. The first part of Theorem 3.1 is proved in Appendix C.1. Below, we concentrate on proving the second part of Theorem 3.1. We do this using a generalization of the symmetry gap framework of Vondrák [52] that is informally stated as Theorem 3.2 (see Appendix C.2 for the formal statement of the theorem). The fractional solution mentioned in this informal statement is evaluate using the multilinear extension of the submodular objective function.

Theorem 3.2. *Consider a non-negative m-monotone submodular function f and a collection $\mathcal{F} \subseteq 2^{\mathcal{N}}$ of feasible sets such that the problem $\max\{f(S) \mid S \in \mathcal{F}\}$ is symmetric with respect to some group $\mathcal{G}$ of permutations over $\mathcal{N}$. If the best fractional solution for this problem which is symmetric with respect to $\mathcal{G}$ is worse by a factor of γ compared to the optimal solution, then we say that the problem has a symmetry gap of γ. In this case, exponentially many value oracle queries are required to obtain $(1+\varepsilon)\gamma$-approximation for the class of problems $\max\{\tilde{f}(S) \mid S \in \tilde{F}\}$ in which $\tilde{f}$ is a non-negative m-monotone submodular function, and $\tilde{F}$ is some generalization of $\mathcal{F}$ (in particular, if $\mathcal{F}$ is a matroid/cardinality constraint, then so is $\tilde{F}$).*

[3]In the second part of Theorem 3.1, like in all the other inapproximability results in this paper, we make the standard assumption that the objective function f can be accessed only through a value oracle that given a set $S \subseteq \mathcal{N}$ returns $f(S)$.

To use Theorem 3.2, we need to define a submodular maximization problem with a significant symmetry gap. Let us choose $\mathcal{N} = \{u, v\}$, $f(S) = m \cdot \mathbf{1}[S \neq \varnothing] + (1 - m) \cdot (|S| \bmod 2)$ and $\mathcal{F} = 2^{\mathcal{N}}$, where m is an arbitrary constant $m \in [0, 1]$. One can verify that f is submodular and non-negative, that its monotonicity ratio is exactly m, and that the problem $\max\{f(S) \mid S \in \mathcal{F}\}$ is symmetric with respect to the group $\mathcal{G}$ of the two possible permutations of $\mathcal{N}$. The following lemma calculates the symmetry gap of this problem, and its proof can be found in Appendix C.3. The second part of Theorem 3.1 follows from this lemma and Theorem 3.2.

Lemma 3.3. *The problem* $\max\{f(S) \mid S \in \mathcal{F}\}$ *has a symmetry gap of* $\frac{1}{2-m}$.

4 Maximization with a Cardinality Constraint

In this section we consider the problem of maximizing a non-negative submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ subject to a cardinality constraint. In other words, we are given an integer value $1 \leq k \leq |\mathcal{N}|$, and the objective is to output a set $S \subseteq \mathcal{N}$ of size at most k (approximately) maximizing f among such sets. A standard greedy algorithm for this problem starts with the empty set, and then iteratively adds elements to this set, choosing in each iteration the element whose addition increases the value of the set by the most. When the objective function f is guaranteed to be monotone, it is long known that this greedy algorithm guarantees $(1 - 1/e)$-approximation for the above problem [44], and that this is essentially the best possible for any polynomial time algorithm [43]. However, the greedy algorithm has no constant approximation guarantee when the objective function is not guaranteed to be monotone (see [4] for an example demonstrating this). In Appendix D.1 we prove Theorem 4.1, which generalizes the result of [44], and proves an approximation guarantee for the greedy algorithm that deteriorates gracefully with the monotonicity ratio m.

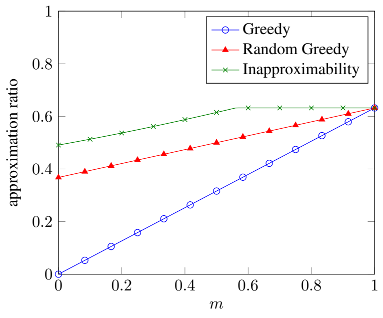
Theorem 4.1. *The Greedy algorithm (Algorithm 2) has an approximation ratio of at least* $m(1 - 1/e)$ *for the problem of maximizing a non-negative* m*-monotone submodular function subject to a cardinality constraint.*

Following a long line of works [35, 22, 46, 52, 6, 20], the state-of-the-art approximation guarantee for the case in which the objective function f is not guaranteed to be monotone is currently 0.385 [5]. However, the algorithm obtaining this approximation ratio is quite involved, which limits its practicality. Arguably, the state-of-the-art approximation ratio obtained by a "simple" algorithm is the $1/e \approx 0.367$-approximation obtained by an algorithm called Random Greedy, which adds to its solution, in each iteration, a uniformly random element out of the k elements that can (individually) add the most to the value of this solution. Random Greedy has the nice property that for monotone objective functions it recovers the optimal $1 - 1/e$ approximation guarantee. In Appendix D.2 we prove Theorem 4.2, which gives an approximation guarantee for Random Greedy that smoothly changes as a function of m and recovers both the above mentioned $1/e$ and $1 - 1/e$ guarantees.

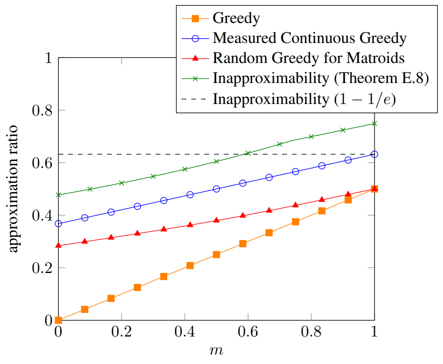
Theorem 4.2. *Random Greedy (Algorithm 3) has an approximation ratio of at least* $m(1 - 1/e) + (1 - m) \cdot (1/e)$ *for the problem of maximizing a non-negative* m*-monotone submodular function subject to a cardinality constraint.*

There is still a gap between the state-of-the-art 0.385-approximation for non-monotone objectives and the state-of-the-art inapproximability result due to Oveis Gharan and Vondrák [46], which only shows that no polynomial time algorithm can guarantee a better than roughly 0.491-approximation. In Appendix D.3 we give Theorem D.4, which uses Theorem 3.2 to prove an inapproximability result that smoothly depends on m and recovers the above mentioned inapproximability results for $m = 0$ and $m = 1$.

To get an intuitive understanding of Theorem D.4, we numerically evaluated it for various values of m. The plot obtained in this way appears in Figure 1a. For context, this figure also includes all the other results proved in this section. As is evident from Figure 1a, Theorem D.4 improves over the $1 - 1/e$ inapproximability result of Nemhauser and Wolsey [43] only for m that is smaller than roughly 0.56. This is surprising since, intuitively, one would expect the best possible approximation ratio to be strictly worse than $1 - 1/e$ for any $m < 1$. However, we were unable to prove an inapproximability that is even slightly lower than $1 - 1/e$ for any $m > 0.56$. Understanding whether this is an artifact of our proof or a real phenomenon is an interesting question that we leave open.



(a) Results for cardinality constraint (Section 4)



(b) Results for matroid constraint (Section 5)

Figure 1: Graphical representation of the results of Sections 4 and 5

5 Maximization with a Matroid Constraint

In this section we consider the problem of maximizing a non-negative submodular function subject to a matroid constraint. A matroid $\mathcal{M}$ over the ground set $\mathcal{N}$ is defined as a pair $\mathcal{M} = (\mathcal{N}, \mathcal{I})$, where $\mathcal{I}$ is a non-empty subset of $2^{\mathcal{N}}$ obeying two properties for every two sets $S, T \subseteq \mathcal{N}$: (i) if $S \subseteq T$ and $T \in \mathcal{I}$, then $S \in \mathcal{I}$; and (ii) if $S, T \in \mathcal{I}$ and $|S| < |T|$, then there exists an element $u \in T \setminus S$ such that $S + u \in \mathcal{I}$. A set $S \subseteq \mathcal{N}$ is called *independent* with respect to a matroid $\mathcal{M}$ if it belongs to $\mathcal{I}$ (otherwise, we say that S is dependent with respect to $\mathcal{M}$); and the matroid constraint corresponding to a given matroid $\mathcal{M}$ allows only independent sets with respect to this matroid as feasible solutions. Hence, we can restate the problem we consider in this section in the following more formal way. Given a non-negative submodular function $f : 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ and a matroid $\mathcal{M} = (\mathcal{N}, \mathcal{I})$, output an independent set $S \in \mathcal{I}$ (approximately) maximizing f among all such sets. It is also useful to note that an independent set $S \in \mathcal{I}$ is called a *base* of $\mathcal{M}$ if it is an inclusion-wise maximal independent set, i.e., S is not a subset of any other independent set.

A standard greedy algorithm for the above problem starts with the empty set, and then iteratively adds to it elements, choosing in each iteration the element that increases the value of the solution by the most among the elements whose addition to the solution does not violate the matroid constraint. When the objective function f is guaranteed to be monotone, this greedy algorithm guarantees $1/2$-approximation [24]. Our first result for this section (proved in Appendix E.1) shows how this approximation guarantee changes as a function of m (the greedy algorithm has no constant guarantee for non-monotone functions in this case as well).

Theorem 5.1. *The Greedy algorithm (Algorithm 4) has an approximation ratio of at least $m/2$ for maximizing a non-negative m-monotone submodular function subject to a matroid constraint.*

The approximation ratio of the greedy algorithm was improved over by the seminal work of Călinescu et al. [9], who described the Continuous Greedy algorithm whose approximation ratio is $1 - 1/e$ when f is monotone; matching the inapproximability result of Nemhauser and Wolsey [43]. In contrast, when f is not guaranteed to be monotone, the approximability of the problem is less well-understood. On the one hand, after a long line of works [35, 22, 46, 52, 20], the state-of-the-art approximation ratio for the problem is 0.385 [5], but on the other hand, it is only known that no polynomial time algorithm for the problem can guarantee 0.478-approximation [46].

Unfortunately, the above mentioned state-of-the-art 0.385-approximation algorithm is quite involved. Therefore, we chose to consider in this work two other algorithms. The first algorithm is Measure Continuous Greedy (due to [22]) which guarantees an approximation ratio of $1/e - o(1) \approx 0.367$. This algorithm performs only slightly worse than the above state-of-the-art, and is a central component of all the currently known algorithms achieving better than $1/e$-approximation. Measured Continuous Greedy is also known to guarantee $(1 - 1/e - o(1))$-approximation when the objective f is monotone, and the next theorem (proved in Appendix E.2) shows that its approximation guarantee changes smoothly with the monotonicity ratio of f.

Theorem 5.2. *Measured Continuous Greedy (Algorithm 5) has an approximation ratio of at least $m(1 - 1/e) + (1 - m) \cdot (1/e) - o(1)$ for maximizing a non-negative m-monotone submodular function subject to a matroid constraint, where the $o(1)$ term diminishes with the ground set's size.*[4]

The other algorithm we consider is an algorithm called Random Greedy for Matroids (due to [6]). Unlike Measured Continuous Greedy and almost all the other algorithms suggested for non-monotone objectives to date, this algorithm is combinatorial, which makes it appealing in practice. It starts with a base solution consisting of dummy elements representing empty slots, and iteratively performs swaps on this base solution in the following way. In each iteration, the algorithm picks a base M maximizing the (individual) marginal value of the elements within it with respect to the current base solution. For every element in M, the algorithm identifies a distinct element of the current base solution with which it can be swapped, and then for a uniformly random element of M such a swap is indeed done. Buchbinder et al. [6] proved an approximation ratio of roughly $(1 + e^{-2})/4$ for Random Greedy for Matroids. The next theorem shows how this approximation guarantee improves as a function of the monotonicity ratio. In this theorem we refer to the rank k of the matroid constraint $\mathcal{M}$, which is the size of the largest independent set with respect to this matroid. We also note that the algorithm we analyze is identical to the algorithm of [6] up to two modifications: our algorithm makes more iterations, and it updates the solution in an iteration only when this increases the solution's value.

Theorem 5.3. *For every $\varepsilon \in (0, 1)$, Random Greedy for Matroids (Algorithm 6) has an approximation ratio of at least $\frac{1+m+e^{-2/(1-m)}}{4} - \varepsilon - o_k(1)$ for the problem of maximizing a non-negative m-monotone submodular function subject to a matroid constraint (except in the case of $m = 1$ in which the approximation ratio is $1/2 - \varepsilon - o_k(1)$), where $o_k(1)$ represents a term that diminishes with k.*

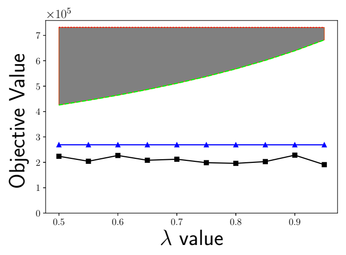
Theorem 5.3 is proved in Appendix E.3. Let us also mention Theorem E.8, which appears in Appendix E.4 and uses Theorem 3.2 to generalize the 0.478 inapproximability result of Oveis Gharan and Vondrák [46]. To get an intuitive understanding of Theorem E.8, we numerically evaluated it for various values of m, and depict the results in Figure 1b. For context, this figure also includes all the other results proved in this section. Somewhat surprisingly, Figure 1b shows that Theorem E.8 does not generalize the $1 - 1/e$ inapproximability result of Nemhauser and Wolsey [43] for monotone functions despite the fact that this inapproximability result holds for every monotonicity ratio $m \in [0, 1]$. This resembles the inability of Theorem D.4 to improve over the same inapproximability result for large values of m.

6 Applications and Experiment Results

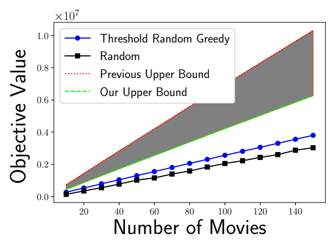
Many machine learning applications require optimization of non-monotone submodular functions subject to some constraint. Unfortunately, such functions enjoy relatively low approximation guarantees. Nevertheless, in many cases the non-monotone objective functions have a significant monotone component that can be captured by the monotonicity ratio. In this section, we discuss two concrete applications with non-monotone submodular objective functions. For each application we provide a lower bound on the monotonicity ratio m of the objective function, which translates via our results from the previous sections into an improved approximation guarantee for the application.

To demonstrate the value of our improved guarantees in experiments, we took the following approach. The output of an approximation algorithm provides an upper bound on the value of the optimal solution for the problem (formally, this upper bound is the value of the output over the approximation ratio of the algorithm). Thus, we plot in each experiment the upper bound on the value of the optimal solution obtained with and without taking into account the monotonicity ratio, which gives a feeling of how the magnitude of our improvements compare to other values of interest (such as the gaps between the performances of the algorithms considered). In Appendix F we give a third application (Image summarization) that we study in the same way; and in Appendix G we lower bound the monotonicity ratio of a fourth application (Ride-Share Optimization).

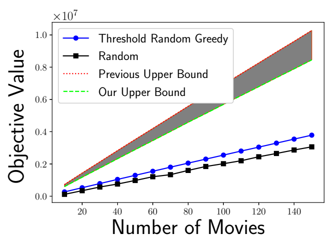
[4]Technically, Measured Continuous Greedy is an algorithm for maximizing the multilinear extesnion of a non-negative submodular function subject to a general solvable down-closed convex body P constraint, and we prove in Appendix E.2 that it guarantees the approximation ratio stated in Theorem 5.2 for this setting. However, this implies the result stated in Theorem 5.2 using a standard reduction (see Appendix E.2 for further detail).



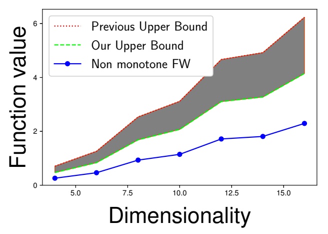
(a) Results when up to 10 movies can be selected for varying λ.



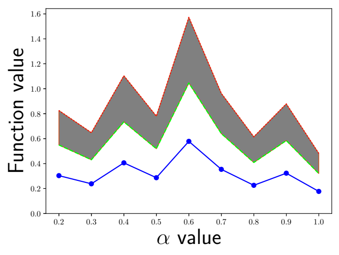
(b) Results for $\lambda = 0.55$ when the number movies in the solution varies.



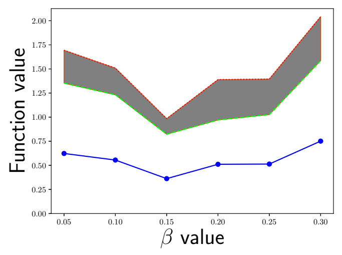
(c) Results for $\lambda = 0.85$ when the number movies in the solution varies.



(d) Varying the dimensionality n for fixed $\alpha = 0.3$ and $\beta = 0.2$.



(e) Varying α for fixed $\beta = 0.2$ and $n = 4$.



(f) Varying β for fixed $\alpha = 0.5$ and $n = 4$.

Figure 2: Experimental results for Personalized Movie Recommendation (a–c) and Quadratic Programming (d–f). Each plot includes the output of the algorithms we consider as well the previous and improved upper bounds on the optimal value (the area between these two bounds is shaded).

6.1 Personalized Movie Recommendation

The first application we consider is Personalized Movie Recommendation. Consider a movie recommendation system where each user specifies what genres she is interested in, and the system has to provide a representative subset of movies from these genres. Assume that each movie is represented by a vector consisting of users' ratings for the corresponding movie. One challenge here is that each user does not necessarily rate all the movies, hence, the vectors representing the movies do not necessarily have similar sizes. To overcome this challenge, a low-rank matrix completion techniques [10] can be performed on the matrix with missing values in order to obtain a complete rating matrix. Formally, given few ratings from k users to n movies we obtain in this way a rating matrix $\mathbf{M}$ of size $k \times n$. Following [40], to score the quality of a selected subset of movies, we use the function $f(S) = \sum_{u \in \mathcal{N}} \sum_{v \in S} s_{u,v} - \lambda \sum_{u \in S} \sum_{v \in S} s_{u,v}$. Here, $\mathcal{N}$ is the set of n movies, $\lambda \in [0, 1]$ is a parameter and $s_{u,v}$ denotes the similarity between movies u and v (the similarity $s_{u,v}$ can be calculated based on the matrix $\mathbf{M}$ in multiple ways: cosine similarity, inner product, etc). Note that the first term in f's definition captures coverage, while the second term captures diversity. Thus, the parameter λ denotes the importance of diversity in the returned subset.

One can verify that the above defined function f is non-negative and submodular. The next theorem, proved in Appendix H.1, analyzes the monotonicity ratio of this function. In this theorem we assume that the similarity scores $s_{u,v}$ are non-negative and obey $s_{u,v} = s_{v,u}$ for every $u, v \in \mathcal{N}$. Note that the above mentioned ways to define these scores have these properties. Interestingly, it turns out that the function f is monotone when λ is small enough despite the fact that this function is traditionally treated as non-monotone (e.g., in [40, 23]). This is a nice unexpected result of the use of the monotonicity ratio, which required us to really understand the degree of non-monotonicity represented by the objective function.

Theorem 6.1. *The objective function f is monotone for $0 \leq \lambda \leq 1/2$ and $2(1 - \lambda)$-monotone for $1/2 \leq \lambda \leq 1$.*

To demonstrate the value of our lower bound on the monotonicity ratio, we followed the experimental setup of [40] and used a subset of movies from the MovieLens data set [28] which includes 10,437 movies. Each movie in this data set is represented by a 25 dimensional feature vector calculated using user ratings, and we used inner products to obtain the similarity values $s_{i,j}$ based on these vectors.

In our experiment we employed accelerated versions of the algorithms analyzed in Section 4 for a cardinality constraint. Specifically, instead of the Greedy algorithm we used Threshold Greedy [1] and Sample Greedy [39]; and instead of Random Greedy we used a threshold based version of this algorithm due to [8] that we refer to as Threshold Random Greedy (Algorithm 6 in [8]). All three algorithms had almost identical performance in our experiments (see Appendix I), thus, to avoid confusion, in Figure 2 we draw only the output of Threshold Random Greedy.

Each plot of Figure 2 depicts the outputs of Threshold Random Greedy and a scarecrow algorithm called Random that simply outputs a random subset of movies of the required size. Each point in the plots represents the average value of the outputs of 10 executions of these algorithms. We also depict in each plot the upper bound on the value of the optimal solution based on the general approximation ratio of Random Greedy and the improved approximation ratio implied by Theorems 4.2 and 6.1—the area between the two upper bounds is shaded. In Figure 2a we plot these values for the case in which we asked the algorithms to pick at most 10 movies, and we vary the parameter λ. In Figures 2b amd 2c we plotted the same values for a fixed parameter λ, while varying the maximum cardinality (number of movies) allowed for the output set. Since the height of the shaded area is on the same order of magnitude as the values of the solutions produced by Threashold Random Greedy (especially when λ is close to $1/2$), our results demonstrate that the improved upper bound we are able to prove is much tighter than the state-of-the-art. Furthermore, our improved upper bound shows that the gap between the empirical outputs of Threshold Random Greedy and Random is much more significant as a percentage of the value of the optimal solution than one could believe based on the weaker bound.

6.2 Quadratic Programming

Consider the function

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x} + c \ . \tag{1}$$

By choosing appropriate matrix $\mathbf{H}$, vector $\mathbf{h}$ and scalar c, this function can be made to have various properties. Specifically, we would like to make it non-negative and DR-submodular (DR-submodularity is an extension of submodularity to continuous functions—see Appendix J for more detail). Our goal in this section is to maximize F under a polytope constraint given by $P = \{\mathbf{x} \in \mathbb{R}^n_{\geq 0} \mid A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \leq \mathbf{u}, A \in \mathbb{R}^{m\times n}_{\geq 0}, \mathbf{b} \in \mathbb{R}^m_{\geq 0}\}$ for some dimensions n and m.

Following Bian et al. [2], we set $m = n$, choose the matrix $\mathbf{H} \in \mathbb{R}^{n\times n}$ to be a randomly generated symmetric matrix whose entries are drawn uniformly at random (and independently) from $[-1, 0]$, and choose $\mathbf{A} \in \mathbb{R}^{m\times n}$ to be a randomly generated matrix whose entries are drawn uniformly at random from $[v, v + 1]$ for $v = 0.01$ (this choice of v guarantees that the entries of A are strictly positive). We also set $\mathbf{b} = \bar{1}$ (i.e., $\mathbf{b}$ is the all ones vector), and $\mathbf{u}$ to be the upper bound on P given by $u_j = \min_{j\in[m]} b_i/A_{i,j}$ for every $j \in [n]$. Finally, we set $\mathbf{h} = -\beta \cdot \mathbf{H}^T\mathbf{u}$ for a parameter $\beta > 0$.

The non-positivity of $\mathbf{H}$ guarantees that f is DR-submodular. To make sure that f is also non-negative, the value of c should be at least $-\min_{\bar{0}\leq\mathbf{x}\leq\mathbf{u}} \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x}$ (where $\bar{0}$ is the all zeros vector). This value can be approximately obtained by using QUADPROGIP[5] [53]. Let the value of this minimum be M; then we set $c = -M + \alpha|M|$ for some parameter $\alpha > 0$.

The definition of the monotonicity ratio can be extend to the continuous setting we consider in this section using the formula $m = \inf_{\bar{0}\leq\mathbf{x}\leq\mathbf{y}\leq\mathbf{u}} \frac{F(\mathbf{y})}{F(\mathbf{x})}$, where the ratio $F(\mathbf{y})/F(\mathbf{x})$ is understood as 1 whenever $F(\mathbf{x}) = 0$. The following theorem analyzes the monotonicity ratio of the function F given in Equation (1) based on this definition. The proof of this theorem can be found in Appendix H.2.

Theorem 6.2. *For $\beta \in (0, 1/2)$, the objective function F given by Equation* (1) *is $\frac{(1-2\beta)\cdot\alpha}{1+\alpha}$-monotone. Furthermore, when $\min_{\bar{0}\leq x\leq\mathbf{u}}(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}\mathbf{x}) \geq 0$, F is even $(1 - 2\beta)$-monotone.*

We applied the Non-monotone Frank-Wolfe algorithm of Bian et al. [2] to the above defined optimization problem (we refer the reader to Appendix J for further detail about this algorithm and

[5]We have used IBM CPLEX optimization studio `https://www.ibm.com/products/ilog-cplex-optimization-studio`.

its analysis). Figures 2d, 2e and 2f depict the results we obtained. Specifically, Figure 2d shows the value of the solution obtained by Non-monotone Frank-Wolfe for $\alpha = 0.3$ and $\beta = 0.2$ as the dimensionality n varies. The shaded area is the area between the previous upper bound on the optimal value, and our upper bound that takes advantage on the monotonicity ratio bound given by Theorem 6.2. Figures 2e and 2f are similar, but they fix the dimensionality n to be 4, and vary α or β instead. Let us discuss now some properties of Figures 2d, 2e and 2f. (i) Each data point in these figures corresponds to a single instance drawn from the distribution described above. This implies that the plots in these figures vary for different runs of our experiment, but the plots that we give represent a (single) typical run. (ii) The size of the the shaded area depends on α and β, but also on the sign of $\min_{\bar{0} \leq \mathbf{x} \leq \mathbf{u}}(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}\mathbf{x})$. This is the reason that this size behaves somewhat non-continuously in Figure 2f. Interestingly, the sign of this minimum is mostly a function of β. In other words, there are values of β for which the minimum is non-negative with high probability, and other values for which the minimum is negative with high probability. (iii) One can see that the use of the monotonicity ratio significantly improves the upper bound on the optimal value, especially when $\min_{\bar{0} \leq \mathbf{x} \leq \mathbf{u}}(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}\mathbf{x})$ happens to be non-negative.

7 Conclusion

In this paper we have defined the monotonicity ratio, analyzed how the approximation ratios of standard submodular maximization algorithms depend on this ratio, and then demonstrated that this leads to improved approximation guarantees for the applications of movie recommendation, image summarization and quadratic programming. We believe that the monotonicity ratio is a natural parameter of submodular maximization problems, refining the binary distinction between monotone and non-monotone objective functions and improving the power of submodular maximization tools in machine learning applications. Thus, we hope to see future work towards understanding the optimal dependence on m of the approximation ratios of various submodular maximization problems.

An important take-home message from our work is that, at least in the unconstrained submodular maximization case, the optimal algorithm has an approximation ratio whose dependence on m is non-linear. Such algorithms are rarely obtained using current techniques, which might be one of the reasons why these techniques have so far failed to obtain tight approximation guarantees for constrained non-monotone submodular maximization.

Funding Transparency Statement

Funding in direct support of this work: This research was supported in part by Israel Science Foundation (ISF) grant no. 459/20.

Additional revenues related to this work: None

References

[1] Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *SODA*, pages 1497–1514, 2014. doi: 10.1137/1.9781611973402.110. URL `https://doi.org/10.1137/1.9781611973402.110`.

[2] An Bian, Kfir Yehuda Levy, Andreas Krause, and Joachim M. Buhmann. Non-monotone continuous DR-submodular maximization: Structure and algorithms. In *NeurIPS*, pages 486–496, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/58238e9ae2dd305d79c2ebc8c1883422-Abstract.html`.

[3] Andrew An Bian, Joachim M. Buhmann, Andreas Krause, and Sebastian Tschiatschek. Guarantees for greedy maximization of non-submodular functions with applications. In *ICML*, pages 498–507, 2017. URL `http://proceedings.mlr.press/v70/bian17a.html`.

[4] Niv Buchbinder and Moran Feldman. Submodular functions maximization problems. In Teofilo F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics, Second Edition, Volume 1: Methologies and Traditional Applications*, pages 753–788. Chapman and

Hall/CRC, 2018. doi: 10.1201/9781351236423-42. URL https://doi.org/10.1201/9781351236423-42.

[5] Niv Buchbinder and Moran Feldman. Constrained submodular maximization via a nonsymmetric technique. *Math. Oper. Res.*, 44(3):988–1005, 2019. doi: 10.1287/moor.2018.0955. URL https://doi.org/10.1287/moor.2018.0955.

[6] Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In Chandra Chekuri, editor, *SODA*, pages 1433–1452. SIAM, 2014. doi: 10.1137/1.9781611973402.106. URL https://doi.org/10.1137/1.9781611973402.106.

[7] Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM J. Comput.*, 44(5):1384–1402, 2015. doi: 10.1137/130929205. URL https://doi.org/10.1137/130929205.

[8] Niv Buchbinder, Moran Feldman, and Roy Schwartz. Comparing apples and oranges: Query trade-off in submodular maximization. *Math. Oper. Res.*, 42(2):308–329, 2017. doi: 10.1287/moor.2016.0809. URL https://doi.org/10.1287/moor.2016.0809.

[9] Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011. doi: 10.1137/080733991. URL https://doi.org/10.1137/080733991.

[10] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

[11] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *FOCS*, pages 575–584, 2010. doi: 10.1109/FOCS.2010.60. URL https://doi.org/10.1109/FOCS.2010.60.

[12] Lin Chen, Moran Feldman, and Amin Karbasi. Weakly submodular maximization beyond cardinality constraints: Does randomization help greedy? In *ICML*, pages 803–812, 2018. URL http://proceedings.mlr.press/v80/chen18b.html.

[13] Michele Conforti and Gérard Cornuéjols. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discret. Appl. Math.*, 7(3):251–274, 1984. doi: 10.1016/0166-218X(84)90003-9. URL https://doi.org/10.1016/0166-218X(84)90003-9.

[14] Min Cui, Donglei Du, Dachuan Xu, and Ruiqi Yang. Approximation algorithm for maximizing nonnegative weakly monotonic set functions. In *CSoNet*, pages 50–58, 2021. doi: 10.1007/978-3-030-91434-9_5. URL https://doi.org/10.1007/978-3-030-91434-9_5.

[15] Abhimanyu Das and David Kempe. Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection. In *ICML*, pages 1057–1064, 2011.

[16] Abhimanyu Das and David Kempe. Approximate submodularity and its applications: Subset selection, sparse approximation and dictionary selection. *J. Mach. Learn. Res.*, 19:3:1–3:34, 2018. URL http://jmlr.org/papers/v19/16-534.html.

[17] Anirban Dasgupta, Ravi Kumar, and Sujith Ravi. Summarization through submodularity and dispersion. In *ACL*, pages 1014–1022. The Association for Computer Linguistics, 2013. URL https://aclanthology.org/P13-1100/.

[18] Ethan R. Elenberg, Alexandros G. Dimakis, Moran Feldman, and Amin Karbasi. Streaming weak submodularity: interpreting neural networks on the fly. In *NeurIPS*, pages 4047–4057, 2017.

[19] Ehsan Elhamifar and M Clara De Paolis Kaluza. Online summarization via submodular and convex optimization. In *CVPR*, pages 1783–1791, 2017.

[20] Alina Ene and Huy L. Nguyen. Constrained submodular maximization: Beyond $1/e$. In Irit Dinur, editor, *FOCS*, pages 248–257. IEEE Computer Society, 2016. doi: 10.1109/FOCS.2016.34. URL https://doi.org/10.1109/FOCS.2016.34.

[21] Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing non-monotone submodular functions. *SIAM J. Comput.*, 40(4):1133–1153, 2011. doi: 10.1137/090779346. URL https://doi.org/10.1137/090779346.

[22] Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In Rafail Ostrovsky, editor, *FOCS*, pages 570–579. IEEE Computer Society, 2011. doi: 10.1109/FOCS.2011.46. URL https://doi.org/10.1109/FOCS.2011.46.

[23] Moran Feldman, Christopher Harshaw, and Amin Karbasi. Greed is good: Near-optimal submodular maximization via greedy optimization. In *COLT*, pages 758–784, 2017. URL http://proceedings.mlr.press/v65/feldman17b.html.

[24] M. Fisher, G. Nemhauser, and L. Wolsey. An analysis of approximations for maximizing submodular set functions–II. *Mathematical Programming*, 8:73–87, 1978.

[25] Mehrdad Ghadiri, Richard Santiago, and F. Bruce Shepherd. A parameterized family of meta-submodular functions. *CoRR*, abs/2006.13754, 2020. URL https://arxiv.org/abs/2006.13754.

[26] Mehrdad Ghadiri, Richard Santiago, and F. Bruce Shepherd. Beyond submodular maximization via one-sided smoothness. In Dániel Marx, editor, *SODA*, pages 1006–1025. SIAM, 2021. doi: 10.1137/1.9781611976465.63. URL https://doi.org/10.1137/1.9781611976465.63.

[27] Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. Constrained non-monotone submodular maximization: Offline and secretary algorithms. In *WINE*, pages 246–257, 2010. doi: 10.1007/978-3-642-17572-5_20. URL https://doi.org/10.1007/978-3-642-17572-5_20.

[28] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (TiiS)*, 5(4):1–19, 2015.

[29] Rishabh K. Iyer, Stefanie Jegelka, and Jeff A. Bilmes. Curvature and optimal algorithms for learning and minimizing submodular functions. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2742–2750, 2013.

[30] Rishabh Krishnan Iyer. *Submodular optimization and machine learning: Theoretical results, unifying and scalable algorithms, and applications*. PhD thesis, University of Washington, 2015.

[31] Ehsan Kazemi, Shervin Minaee, Moran Feldman, and Amin Karbasi. Regularized submodular maximization at scale. In *ICML*, pages 5356–5366. PMLR, 2021.

[32] Katrin Kirchhoff and Jeff Bilmes. Submodularity for data selection in machine translation. In *EMNLP*, pages 131–141, 2014.

[33] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[34] Alan Kuhnle, J. David Smith, Victoria G. Crawford, and My T. Thai. Fast maximization of non-submodular, monotonic functions on the integer lattice. In Jennifer G. Dy and Andreas Krause, editors, *ICML*, pages 2791–2800. PMLR, 2018. URL http://proceedings.mlr.press/v80/kuhnle18a.html.

[35] Jon Lee, Vahab S. Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In Michael Mitzenmacher, editor, *STOC*, pages 323–332. ACM, 2009. doi: 10.1145/1536414.1536459. URL https://doi.org/10.1145/1536414.1536459.

[36] Qi Lei, Lingfei Wu, Pin-Yu Chen, Alex Dimakis, Inderjit S. Dhillon, and Michael J. Witbrock. Discrete adversarial attacks and submodular optimization with applications to text classification. In Ameet Talwalkar, Virginia Smith, and Matei Zaharia, editors, *MLSys*. mlsys.org, 2019. URL https://proceedings.mlsys.org/book/284.pdf.

[37] Hui Lin and Jeff Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies (HLT)*, pages 912–920, 2010.

[38] László Lovász. Submodular functions and convexity. In A. Bachem, M. Grötschel, and B. Korte, editors, *Mathematical Programming: the State of the Art*, pages 235–257. Springer, 1983.

[39] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier than lazy greedy. In *AAAI*, pages 1812–1818, 2015. URL `http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9956`.

[40] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, and Amin Karbasi. Fast constrained submodular maximization: Personalized data summarization. In *ICML*, pages 1358–1367. PMLR, 2016.

[41] Marko Mitrovic, Ehsan Kazemi, Morteza Zadimoghaddam, and Amin Karbasi. Data summarization at scale: A two-stage submodular approach. In *ICML*, pages 3596–3605. PMLR, 2018.

[42] Marko Mitrovic, Ehsan Kazemi, Moran Feldman, Andreas Krause, and Amin Karbasi. Adaptive sequence submodularity. *NeurIPS*, 32:5352–5363, 2019.

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

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

[45] Ashkan Norouzi-Fard, Jakub Tarnawski, Slobodan Mitrovic, Amir Zandieh, Aidasadat Mousavifar, and Ola Svensson. Beyond 1/2-approximation for submodular maximization on massive data streams. In *ICML*, pages 3829–3838. PMLR, 2018.

[46] Shayan Oveis Gharan and Jan Vondrák. Submodular maximization by simulated annealing. In Dana Randall, editor, *SODA*, pages 1098–1116. SIAM, 2011. doi: 10.1137/1.9781611973082.83. URL `https://doi.org/10.1137/1.9781611973082.83`.

[47] Benjamin Qi. On maximizing sums of non-monotone submodular and linear functions. *CoRR*, abs/2205.15874, 2022. doi: 10.48550/arXiv.2205.15874. URL `https://doi.org/10.48550/arXiv.2205.15874`. To appear in ISAAC 2022.

[48] Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, 2003.

[49] Maxim Sviridenko, Jan Vondrák, and Justin Ward. Optimal approximation for submodular and supermodular optimization with bounded curvature. *Math. Oper. Res.*, 42(4):1197–1218, 2017. doi: 10.1287/moor.2016.0842. URL `https://doi.org/10.1287/moor.2016.0842`.

[50] Sebastian Tschiatschek, Rishabh K Iyer, Haochen Wei, and Jeff A Bilmes. Learning mixtures of submodular functions for image collection summarization. In *NeurIPS*, pages 1413–1421, 2014.

[51] Sebastian Tschiatschek, Adish Singla, and Andreas Krause. Selecting sequences of items via submodular maximization. In Satinder P. Singh and Shaul Markovitch, editors, *AAAI*, pages 2667–2673. AAAI Press, 2017. URL `http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14898`.

[52] Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM J. Comput.*, 42(1):265–304, 2013. doi: 10.1137/110832318. URL `https://doi.org/10.1137/110832318`.

[53] Wei Xia, Juan-Carlos Vera, and Luis F. Zuluaga. Globally solving nonconvex quadratic programs via linear integer programming techniques. *INFORMS J. Comput.*, 32(1):40–56, 2020. doi: 10.1287/ijoc.2018.0883. URL `https://doi.org/10.1287/ijoc.2018.0883`.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] Most of them in the supplementary material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] in the supplementary material.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Additional Related Work

Lin and Bilmes [37] described an algorithm that takes advantages of a continuous partial monotonicity property, but unlike the monotonicity ratio, their property was defined in terms of the particular submodular objective they were interested in. More recently, Cui et al. [14] considered a weaker, but still binary, version of monotonicity called weak-monotonicity.

B Proof of Lemma 2.1

In this section we prove Lemma 2.1.

Lemma 2.1. *Let $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ be a non-negative m-monotone submodular function. For every deterministic set $O \subseteq \mathcal{N}$ and random set $D \subseteq \mathcal{N}$, $\mathbb{E}[f(O \cup D)] \geq (1 - (1-m) \cdot \max_{u \in \mathcal{N}} \Pr[u \in D]) \cdot f(O)$.*

To prove this lemma we first have to define the Lovász extension of set functions. The *Lovász extension* of a set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ is a function $\hat{f}\colon [0,1]^{\mathcal{N}} \to \mathbb{R}$ defined as follows. For every vector $\mathbf{x} \in [0,1]^{\mathcal{N}}$,

$$\hat{f}(\mathbf{x}) = \int_0^1 f(T_\lambda(\mathbf{x}))d\lambda \enspace ,$$

where $T_\lambda(\mathbf{x}) \triangleq \{u \in \mathcal{N} \mid x_u \geq \lambda\}$. The Lovász extension of a submodular function is known to be convex. More important for us is the following known lemma regarding this extension. This lemma stems from an equality, proved by Lovász [38], between the Lovász extension of a submodular function and another extension known as the convex closure.

Lemma B.1. *Let $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ be a submodular function, and let $\hat{f}$ be its Lovász extension. For every $\mathbf{x} \in [0,1]^{\mathcal{N}}$ and random set $D_{\mathbf{x}} \subseteq \mathcal{N}$ obeying $\Pr[u \in D_{\mathbf{x}}] = x_u$ for every $u \in \mathcal{N}$ (i.e., the marginals of $D_{\mathbf{x}}$ agree with $\mathbf{x}$), $\hat{f}(\mathbf{x}) \leq \mathbb{E}[f(D_{\mathbf{x}})]$.*

Using the last lemma, we can now prove Lemma 2.1.

Proof of Lemma 2.1. Let $\mathbf{x}$ be the vector of marginals of $O \cup D$, i.e., $x_u = \Pr[u \in O \cup D]$ for every $u \in \mathcal{N}$. Then, by Lemma B.1,

$$\begin{aligned}
\mathbb{E}[f(O \cup D)] \geq \hat{f}(\mathbf{x}) &= \int_0^1 f(T_\lambda(\mathbf{x}))d\lambda \\
&= \int_0^{\max_{u \in \mathcal{N}} \Pr[u \in D]} f(T_\lambda(\mathbf{x}))d\lambda + \int_{\max_{u \in \mathcal{N}} \Pr[u \in D]}^1 f(T_\lambda(\mathbf{x}))d\lambda \\
&= \int_0^{\max_{u \in \mathcal{N}} \Pr[u \in D]} f(O \cup T_\lambda(\mathbf{x}))d\lambda + (1 - \max_{u \in \mathcal{N}} \Pr[u \in D]) \cdot f(O) \enspace ,
\end{aligned}$$

where the last equality holds since the elements of O appear in $T_\lambda(\mathbf{x})$ for every $\lambda \in [0,1]$, and no other element appears in $T_\lambda(\mathbf{x})$ when $\lambda > \Pr[u \in D]$. Using the definition of the monotonicity ratio, the expression $f(O \cup T_\lambda(\mathbf{x}))$ on the rightmost side of the previous equation can be lower bounded by $m \cdot f(O)$, which yields

$$\begin{aligned}
\mathbb{E}[f(O \cup D)] &\geq \int_0^{\max_{u \in \mathcal{N}} \Pr[u \in D]} m \cdot f(O)d\lambda + (1 - \max_{u \in \mathcal{N}} \Pr[u \in D]) \cdot f(O) \\
&= m \cdot \max_{u \in \mathcal{N}} \Pr[u \in D] \cdot f(O) + (1 - \max_{u \in \mathcal{N}} \Pr[u \in D]) \cdot f(O) \\
&= (1 - (1-m) \cdot \max_{u \in \mathcal{N}} \Pr[u \in D]) \cdot f(O) \enspace . \qquad \square
\end{aligned}$$

C Proofs of Section 3

In this section we give the proofs of Section 3.

C.1 Proof of the first part of Theorem 3.1

In this section we prove the first part of Theorem 3.1, which is restated by the following theorem.

Theorem C.1. *The double greedy algorithm of Buchinder et al. [7] guarantees* $[1/(2-m)]$*-approximation for unconstrained submodular maximization.*

Let $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ be an arbitrary non-negative m-monotone submodular function over the ground set $\mathcal{N}$. To prove Theorem C.1, we need to show that given f, the double greedy algorithm of Buchinder et al. [7] outputs a (random) set S obeying $\mathbb{E}[f(S)] \geq f(OPT)/(2-m)$, where OPT is some subset of $\mathcal{N}$ maximizing f. Therefore, we start by looking at the approximation guarantees that are known for double greedy when ignoring the monotonicity ratio.

Buchinder et al. [7] proved that the output S of double greedy always obeys

$$\mathbb{E}[f(S)] \geq \frac{2f(OPT) + f(\varnothing) + f(\mathcal{N})}{4} \enspace .$$

However, it turns out that this guarantee is only a special case of a more general guarantee that can be proved. Specifically, we prove below the following guarantee.

Proposition C.2. *The (random) output set S of double greedy obeys*

$$\mathbb{E}[f(S)] \geq \frac{2r}{(r+1)^2} \cdot f(OPT) + \frac{1}{(r+1)^2} \cdot f(\varnothing) + \frac{r^2}{(r+1)^2} \cdot f(\mathcal{N})$$

for every value $r > 0$, simultaneously (i.e., the algorithm need not know r).

Proposition C.2 is based on ideas first used by Buchbinder et al. [6] in the context of an algorithm that is related to double greedy. Recently, Qi [47] observed that these ideas are useful also in the context of the double greedy algorithm, and used them to derive an improved result for a related problem termed "Regularized Unconstrained Submodular Maximization". Proposition C.2 is another consequence of the application of these ideas to double greedy.

Before getting to the proof of Proposition C.2, let us show that it implies Theorem C.1.

Proof of Theorem C.1. Since f is m monotone and $OPT \subseteq \mathcal{N}$, $f(N) \geq m \cdot f(OPT)$. Additionally, the non-negativity of f guarantees $f(\varnothing) \geq 0$. Plugging both these observations into the guarantee of Proposition C.2, we get

$$\begin{aligned}\mathbb{E}[f(S)] &\geq \frac{2r}{(r+1)^2} \cdot f(OPT) + \frac{1}{(r+1)^2} \cdot f(\varnothing) + \frac{r^2}{(r+1)^2} \cdot f(\mathcal{N}) \\ &\geq \frac{2r}{(r+1)^2} \cdot f(OPT) + \frac{r^2}{(r+1)^2} \cdot [m \cdot f(OPT)] = \frac{2r + r^2 m}{(r+1)^2} \cdot f(OPT) \enspace .\end{aligned}$$

Since the above inequality holds for every $r > 0$, we can choose $r = 1/(1-m)$, and get

$$\begin{aligned}\mathbb{E}[f(S)] &\geq \frac{2r + r^2 m}{(r+1)^2} \cdot f(OPT) = \frac{2/r + m}{(1+1/r)^2} \cdot f(OPT) \\ &= \frac{2(1-m) + m}{(1+(1-m))^2} \cdot f(OPT) = \frac{2-m}{(2-m)^2} \cdot f(OPT) = \frac{1}{2-m} \cdot f(OPT) \enspace . \qquad \square\end{aligned}$$

The rest of this section is devoted to proving Proposition C.2. To prove this proposition, we first need to describe the double greedy algorithm that it analyzes, which appears as Algorithm 1. In a nutshell, this algorithm maintains two solutions X and Y that are originally the empty set and entire ground set, respectively. In every iteration, the algorithm considers a different element of the ground set, and either adds it to X, or removes it from Y. Once all the elements have been considered, the sets X and Y become identical, and they are the output of the algorithm. Note also that Algorithm 1 uses n to denote the size of the ground set $\mathcal{N}$.

The heart of the analysis of Algorithm 1 is Lemma C.3, which was proved by Qi [47], and we prove here in more detail for completeness. To state Lemma C.3, we need to define for every integer $0 \leq i \leq n$ the set $OPT_i = (OPT \cup X_i) \cap Y_i$. Notice that OPT_i agrees with X_i and Y_i on all the elements on which these two sets agree (i.e., elements that Algorithm 1 considered in its first i iterations). On the remaining elements, OPT_i agrees with OPT. Thus, as i increases, OPT_i evolves from being equal to OPT to being equal to the output set $X_n = Y_n$ of Algorithm 1.

Algorithm 1: `Double-Greedy`

```
1  Denote the elements of $\mathcal{N}$ by $u_1, u_2, \ldots, u_n$ in an arbitrary order.
2  Let $X_0 \leftarrow \varnothing$ and $Y_0 \leftarrow \varnothing$.
3  for $i = 1$ to $n$ do
4      Let $a_i \leftarrow f(u_i \mid X_{i-1})$ and $b_i \leftarrow -f(u_i \mid Y_{i-1} - u_i)$.
5      if $b_i \leq 0$ then Let $X_i \leftarrow X_{i-1} + u_i$ and $Y_i \leftarrow Y_{i-1}$.
6      else if $a_i \leq 0$ then Let $X_i \leftarrow X_{i-1}$ and $Y_i \leftarrow Y_{i-1} - u_i$.
7      else
8          with probability $\frac{a_i}{a_i+b_i}$ do Let $X_i \leftarrow X_{i-1} + u_i$ and $Y_i \leftarrow Y_{i-1}$.
9          otherwise Let $X_i \leftarrow X_{i-1}$ and $Y_i \leftarrow Y_{i-1} - u_i$. // Occurs with prob. $\frac{b_i}{a_i+b_i}$.
10     return $X_n (= Y_n)$.
```

Lemma C.3. *For every integer* $1 \leq i \leq n$ *and* $r > 0$*,*

$$\mathbb{E}[f(OPT_{i-1}) - f(OPT_i)] \leq \frac{1}{2}\mathbb{E}[r^{-1}(f(X_i) - f(X_{i-1})) + r(f(Y_i) - f(Y_{i-1}))] \enspace .$$

Proof. By the law of total expectation, it suffices to prove the lemma conditioned on any particular choice for the random bits tossed in the first $i-1$ iterations of Algorithm 1. Notice that once these random bits are fixed, OPT_{i-1}, X_{i-1} and Y_{i-1} become deterministic sets, and so do the numbers a_i and b_i calculated by Algorithm 1. We now need to consider three cases based on the values of these numbers.

The first case is the case of $b_i \leq 0$. In this case Algorithm 1 deterministically set $X_i \leftarrow X_{i-1} + u_i$ and $Y_i \leftarrow Y_{i-1}$, which reduces the inequality that we need to prove to

$$f(OPT_{i-1}) - f(OPT_{i-1} + u_i) \leq \tfrac{1}{2r}[f(X_{i-1} + u_i) - f(X_{i-1})] \enspace . \tag{2}$$

Observe that the description of Algorithm 1 implies $X_{i-1} \subseteq Y_{i-1}$, $u_i \notin X_{i-1}$ and $u_i \in Y_{i-1}$ (see Buchinder et al. [7] for a formal proof of these properties). Given these properties, the submodularity of f shows that the right hand side of Inequality (2) is non-negative because

$$\begin{aligned} f(X_{i-1} + u_i) - f(X_{i-1}) \geq{}& [f((X_{i-1} + u_i) \cup (Y_{i-1} - u_i)) \\ &+ f((X_{i-1} + u_i) \cap (Y_{i-1} - u_i)) - f(Y_{i-1} - u_i)] - f(X_{i-1}) \\ ={}& f(Y_{i-1}) + f(X_{i-1}) - f(Y_{i-1} - u_i) - f(X_{i-1}) = -b_i \geq 0 \enspace . \end{aligned}$$

To complete the proof of the first case, it remains to show that the left hand side of Inequality (2) is non-positive. If $u_i \in OPT_{i-1}$, then this left hand side is trivially 0. Otherwise, since $OPT_{i-1} \subseteq Y_{i-1}$ by definition, the submodularity of f shows that this left hand side is non-positive because

$$f(OPT_{i-1}) - f(OPT_{i-1} + u_i) \leq f(Y_{i-1} - u_i) - f(Y_{i-1}) = b_i \leq 0 \enspace .$$

The second case that we need to consider is the case of $b_i > 0$ and $a_i \leq 0$. However, since the analysis of this case is analogous to the analysis of the previous case, we omit it. Thus, we are left with the case in which both a_i and b_i are positive. The analysis of this case consists of two sub-cases depending on whether $u_i \in OPT$ or not. Since the proofs of these sub-cases are analogous to each other, we assume from this point on that $u_i \notin OPT$.

In the case we consider, Algorithm 1 sets $X_i \leftarrow X_{i-1} + u_i$ and $Y_i \leftarrow Y_{i-1}$ with probability $a_i/(a_i + b_i)$, and otherwise it sets $X_i \leftarrow X_{i-1}$ and $Y_i \leftarrow Y_{i-1} - u_i$. Thus, the law of total expectation implies

$$\begin{aligned} \mathbb{E}[f(OPT_{i-1}) - f(OPT_i)] ={}& \frac{a_i}{a_i + b_i} \cdot [f(OPT_{i-1}) - f(OPT_{i-1} + u_i)] \\ &+ \frac{b_i}{a_i + b_i} \cdot [f(OPT_{i-1}) - f(OPT_{i-1})] \\ ={}& \frac{a_i}{a_i + b_i} \cdot [f(OPT_{i-1}) - f(OPT_{i-1} + u_i)] \enspace , \end{aligned}$$

and

$$\begin{aligned}
\mathbb{E}[r^{-1}(f(X_i) - f(X_{i-1})) + r(f(Y_i) - f(Y_{i-1}))]& \\
&= \frac{a_i}{a_i + b_i} \cdot [r^{-1}(f(X_{i-1} + u_i) - f(X_{i-1})) + r(f(Y_{i-1}) - f(Y_{i-1}))] \\
&\quad + \frac{b_i}{a_i + b_i} \cdot [r^{-1}(f(X_{i-1}) - f(X_{i-1})) + r(f(Y_{i-1} - u_i) - f(Y_{i-1}))] \\
&= \frac{r^{-1}a_i^2}{a_i + b_i} + \frac{rb_i^2}{a_i + b_i} \enspace .
\end{aligned}$$

Plugging both these equalities into the inequality that we need to prove, we get that this inequality is equivalent to

$$\frac{a_i}{a_i + b_i} \cdot [f(OPT_{i-1}) - f(OPT_{i-1} + u_i)] \leq \frac{1}{2}\left[\frac{r^{-1}a_i^2}{a_i + b_i} + \frac{rb_i^2}{a_i + b_i}\right] \enspace .$$

Furthermore, like in the first case, we have $f(OPT_{i-1}) - f(OPT_{i-1} + u_i) \leq b_i$, and therefore, it suffices to prove the inequality

$$\frac{a_i b_i}{a_i + b_i} \leq \frac{1}{2}\left[\frac{r^{-1}a_i^2}{a_i + b_i} + \frac{rb_i^2}{a_i + b_i}\right] \enspace ,$$

which holds since

$$r^{-1}a_i^2 + rb_i^2 = (a_i/\sqrt{r} - b_i\sqrt{r})^2 + 2(a_i/\sqrt{r})(b_i\sqrt{r}) \geq 2a_i b_i \enspace . \qquad \square$$

Using Lemma C.3, we can now prove Proposition C.2.

Proof of Proposition C.2. Throughout this proof, r is an arbitrary positive number. Summing up the guarantees of Lemma C.3 for all integers $1 \leq i \leq n$ yields

$$\sum_{i=1}^{n} \mathbb{E}[f(OPT_{i-1}) - f(OPT_i)] \leq \frac{1}{2}\mathbb{E}\left[r^{-1} \cdot \sum_{i=1}^{n}(f(X_i) - f(X_{i-1})) + r \cdot \sum_{i=1}^{n}(f(Y_i) - f(Y_{i-1}))\right].$$

Using the linearity of expectation, we can collapse the telescopic sums in the last inequality, and get

$$\mathbb{E}[f(OPT_0) - f(OPT_n)] \leq \frac{1}{2}\mathbb{E}[r^{-1}(f(X_n) - f(X_0)) + r(f(Y_n) - f(Y_0))] \enspace .$$

As explained above, $OPT_0 = OPT$ and $OPT_n = X_n = Y_n$. Additionally, X_0 and Y_0 are set by Algorithm 1 to $\varnothing$ and $\mathcal{N}$, respectively. Plugging all these equalities into the previous inequality reduces it to

$$\mathbb{E}[f(OPT) - f(X_n)] \leq \frac{1}{2}\mathbb{E}[r^{-1}(f(X_n) - f(\varnothing)) + r(f(X_n) - f(\mathcal{N}))] \enspace .$$

The proposition now follows by rearranging the above inequality, and observing that X_n is the output set S mentioned in the statement of the proposition. $\square$

C.2 Proof of Theorem 3.2

In this section, we show how the proof of the symmetry gap technique due to Vondrák [52] can be adapted to prove Theorem 3.2. Let us begin the section by stating some definitions that are required in order to formally state Theorem 3.2.

Definition C.4. [Strong symmetry] Consider a non-negative submodular function f and a collection $\mathcal{F} \subseteq 2^{\mathcal{N}}$ of feasible sets. The problem $\max\{f(S) \mid S \in \mathcal{F}\}$ is strongly symmetric with respect to a group of permutations $\mathcal{G}$ on $\mathcal{N}$, if (1) $f(S) = f(\sigma(S))$ for all $S \subseteq \mathcal{N}$ and $\sigma \in \mathcal{G}$, and (2) $S \in \mathcal{F} \iff S' \in \mathcal{F}$ whenever $\mathbb{E}_{\sigma \in \mathcal{G}}[\mathbf{1}_{\sigma(S)}] = \mathbb{E}_{\sigma \in \mathcal{G}}[\mathbf{1}_{\sigma(S')}]$, where $\mathbb{E}_{\sigma \in \mathcal{G}}$ represents the expectation over picking σ uniformly at random out of $\mathcal{G}$.

Definition C.5. [Symmetry gap] Consider a non-negative submodular function f and a collection $\mathcal{F} \subseteq 2^{\mathcal{N}}$ of feasible sets. Let $F(x)$ be the multilinear extension of f and $P(\mathcal{F}) \subseteq [0,1]^{\mathcal{N}}$ be the convex hull of $\mathcal{F}$. Then, if the problem $\max\{f(S) \mid S \in \mathcal{F}\}$ is strongly symmetric with respect to a group $\mathcal{G}$ of permutation, then its symmetry is defined as

$$\frac{\max\{F(\bar{\mathbf{x}}) \mid \mathbf{x} \in P(\mathcal{F})\}}{\max\{F(\mathbf{x}) \mid \mathbf{x} \in P(\mathcal{F})\}} ,$$

where $\bar{\mathbf{x}} \triangleq \mathbb{E}_{\sigma \in \mathcal{G}}[\sigma(\mathbf{x})]$.

Definition C.6. [Refinement] Consider a set $\mathcal{F} \subseteq 2^{\mathcal{N}}$, and let X be some set. We say that $\tilde{\mathcal{F}} \subseteq 2^{\mathcal{N} \times X}$ is a refinement of $\mathcal{F}$ if

$$\tilde{F} = \left\{ S \subseteq \mathcal{N} \times X \;\middle|\; \mathbf{x} \in P(\mathcal{F}), \text{ where } x_u = \tfrac{|S \cap (\{u\} \times X)|}{|X|} \text{ for every } u \in \mathcal{N} \right\} .$$

Using the above definitions, we can now formally state the theorem that we want to prove.

Theorem 3.2. *Consider a non-negative m-monotone submodular function f and a collection $\mathcal{F} \subseteq 2^{\mathcal{N}}$ of feasible sets such that the problem $\max\{f(S) \mid S \in \mathcal{F}\}$ is strongly symmetric with respect to some group $\mathcal{G}$ of permutations over $\mathcal{N}$ and has a symmetry gap γ. Let $\mathcal{C}$ be the class of problems $\max\{\tilde{f}(S) \mid S \in \tilde{F}\}$ in which $\tilde{f}$ is a non-negative m-monotone submodular function, and $\tilde{F}$ is a refinement of F. Then, for every $\varepsilon > 0$, any (even randomized) $(1+\varepsilon)\gamma$-approximation algorithm for the class $\mathcal{C}$ would require exponentially many value queries to $\tilde{f}$.*

The crux of the symmetry gap technique is two lemmata due to [52] that we restate below. Lemma C.7 shows that given a non-negative set function f, one can obtain from it two continuous versions: a continuous version $\hat{F}$ that resembles f itself, and a continuous version $\hat{G}$ that resembles a symmetrized version of f. Distinguishing between $\hat{F}$ and $\hat{G}$ is difficult, however, this does not translate into an hardness for discrete problems since $\hat{F}$ and $\hat{G}$ are continuous. Therefore, Vondrák [52] proved also Lemma C.8, which shows how these continuous functions can be translated back into set functions with appropriate properties.

Lemma C.7 (Lemma 3.2 of [52]). *Consider a function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ invariant under a group of permutations $\mathcal{G}$ on the ground set $\mathcal{N}$. Let $F(\mathbf{x})$ be the multilinear extension of F, define $\bar{x} = \mathbb{E}_{\sigma \in \mathcal{G}}[\mathbf{1}_{\sigma(\mathbf{x})}]$ and fix any $\varepsilon > 0$. Then, there is $\delta > 0$ and functions $\hat{F}, \hat{G}\colon [0,1]^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ (which are also symmetric with respect to $\mathcal{G}$), satisfying the following:*

1. *For all $\mathbf{x} \in [0,1]^{\mathcal{N}}$, $\hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$.*
2. *For all $\mathbf{x} \in [0,1]^{\mathcal{N}}$, $|\hat{F}(\mathbf{x}) - F(\mathbf{x})| \leq \varepsilon$.*
3. *Whenever $\|\mathbf{x} - \bar{\mathbf{x}}\|_2 \leq \delta$, $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x})$ and the value depends only on $\bar{\mathbf{x}}$.*
4. *The first partial derivatives of $\hat{F}$ and $\hat{G}$ are absolutely continuous.*
5. *If f is monotone, then, for every element $u \in \mathcal{N}$, $\frac{\partial \hat{F}}{\partial x_u} \geq 0$ and $\frac{\partial \hat{G}}{\partial x_u} \geq 0$ everywhere.*
6. *If f is submodular then, for every two elements $u, v \in \mathcal{N}$, $\frac{\partial^2 \hat{F}}{\partial x_u \partial x_v} \leq 0$ and $\frac{\partial^2 \hat{G}}{\partial x_u \partial x_v} \leq 0$ almost everywhere.*

Lemma C.8 (Lemma 3.1 of [52]). *Let n be a positive integer, and let $F\colon [0,1]^{\mathcal{N}} \to \mathbb{R}$ and $X = [n]$. If we define $f\colon 2^{\mathcal{N} \times X} \to \mathbb{R}_{\geq 0}$ so that $f(S) = F(\mathbf{x})$, where $x_u = \frac{1}{n}|S \cap (\{u\} \times X)|$. Then,*

1. *if $\frac{\partial F}{\partial x_u} \geq 0$ everywhere for each element $u \in \mathcal{N}$, then f is monotone,*
2. *and if the first partial derivatives of F are absolutely continuous and $\frac{\partial^2 F}{\partial x_u \partial x_v} \leq 0$ almost everywhere for all elements $u, v \in \mathcal{N}$, then f is submodular.*

One can note that the above lemmata have the property that if the function f plugged into Lemma C.7 is monotone, then the discrete functions obtained by applying Lemma C.8 to the functions $\hat{F}$ and $\hat{G}$ are also monotone. This is the reason that the framework of [52] applies to monotone functions (as well as general, not necessarily monotone, functions). Therefore, to get the proof of [52] to yield Theorem 3.2, it suffices to prove the following two modified versions of Lemmata C.7 and C.8. These modified versions preserve m-monotonicity for any $m \in [0, 1]$, rather than just standard monotonicity.

Lemma C.9 (modified version of Lemma C.7)**.** *Consider a function $f: 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ that is m-monotone and invariant under a group of permutations $\mathcal{G}$ on the ground set $\mathcal{N}$. Let $F(\mathbf{x})$ be the multilinear extension of F, define $\bar{x} = \mathbb{E}_{\sigma \in \mathcal{G}}[\mathbf{1}_{\sigma(\mathbf{x})}]$ and fix any $\varepsilon > 0$. Then, there is $\delta > 0$ and functions $\hat{F}, \hat{G}: [0,1]^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ (which are also symmetric with respect to $\mathcal{G}$), satisfying the following:*

1. *For all $\mathbf{x} \in [0,1]^{\mathcal{N}}$, $\hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$.*
2. *For all $\mathbf{x} \in [0,1]^{\mathcal{N}}$, $|\hat{F}(\mathbf{x}) - F(\mathbf{x})| \leq \varepsilon$.*
3. *Whenever $\|\mathbf{x} - \bar{\mathbf{x}}\|_2 \leq \delta$, $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x})$ and the value depends only on $\bar{\mathbf{x}}$.*
4. *For every two vectors $\mathbf{x}, \mathbf{y} \in [0,1]^{\mathcal{N}}$ obeying $\mathbf{x} \leq \mathbf{y}$, $m \cdot F(\mathbf{x}) \leq F(\mathbf{y})$.*
5. *If f is submodular then, for every two elements $u, v \in \mathcal{N}$, $\frac{\partial^2 \hat{F}}{\partial x_u \partial x_v} \leq 0$ and $\frac{\partial^2 \hat{G}}{\partial x_u \partial x_v} \leq 0$ almost everywhere.*

Lemma C.10 (modified version of Lemma C.8)**.** *Let n be a positive integer, and let $F: [0,1]^{\mathcal{N}} \to \mathbb{R}$ and $X = [n]$. If we define $f: 2^{\mathcal{N} \times X} \to \mathbb{R}_{\geq 0}$ so that $f(S) = F(\mathbf{x})$, where $x_u = \frac{1}{n}|S \cap (\{u\} \times X)|$. Then,*

1. *if for some value $m \in [0,1]$ the inequality $m \cdot F(\mathbf{x}) \leq F(\mathbf{y})$ holds for any two vectors $\mathbf{x}, \mathbf{y} \in [0,1]^{\mathcal{N}}$ that obey $\mathbf{x} \leq \mathbf{y}$, then f is m-monotone,*
2. *and if the first partial derivatives of F are absolutely continuous and $\frac{\partial^2 F}{\partial x_u \partial x_v} \leq 0$ almost everywhere for all elements $u, v \in \mathcal{N}$, then f is submodular.*

The proof of Lemma C.9 is quite long and appears below. However, before getting to this proof, let first give the much simpler proof of Lemma C.10.

Proof of Lemma C.10. The second point in Lemma C.10 follows immediately from Lemma C.8, so we concentrate on proving the first point. In other words, we assume that $m \cdot F(\mathbf{x}) \leq F(\mathbf{y})$ for every two vectors $\mathbf{x}, \mathbf{y} \in [0,1]^{\mathcal{N}}$ obeying $\mathbf{x} \leq \mathbf{y}$, and we need to show that $m \cdot f(S) \leq f(T)$ for every two sets $S \subseteq T \subseteq \mathcal{N}$.

Let us define two vectors $\mathbf{x}^{(S)}, \mathbf{x}^{(T)} \subseteq [0,1]^{\mathcal{N}}$ as follows. For every $u \in \mathcal{N}$,

$$x_u^{(S)} = \frac{1}{n}|S \cap (\{u\} \times X)| \quad \text{and} \quad x_u^{(T)} = \frac{1}{n}|T \cap (\{u\} \times X)| \enspace .$$

Since $S \subseteq T$, we get $\mathbf{x}^{(S)} \leq \mathbf{x}^{(T)}$, which implies $m \cdot F(\mathbf{x}^{(S)}) \leq F(\mathbf{x}^{(T)})$; and the last inequality proves the lemma since $f(S) = F(\mathbf{x}^{(S)})$ and $f(T) = F(\mathbf{x}^{(T)})$ by the definition of f. $\square$

We now get to the proof of Lemma C.9. We use in this proof functions $\hat{F}$ and $\hat{G}$ that are similar to the ones constructed by Vondrák [52] in the proof of Lemma C.7. Specifically, like in the proof of [52], we define

$$\hat{G}(\mathbf{x}) = G(\mathbf{x}) + 256M|\mathcal{N}|\alpha J(\mathbf{x}) \enspace ,$$

where M is the maximum value that the function f takes on any set, G is a symmetrized version of the multilinear extension F of f defined as $G(\mathbf{x}) = F(\bar{\mathbf{x}})$, $J(\mathbf{x}) \triangleq |\mathcal{N}|^2 + 3|\mathcal{N}| \cdot \sum_{u \in \mathcal{N}} x_u - \left(\sum_{u \in \mathcal{N}} x_u\right)^2$, and α is a positive value that is independent of $\mathbf{x}$. Similarly, the function $\hat{F}$ was defined by Vondrák [52] as

$$\hat{F}(\mathbf{x}) = \tilde{F}(\mathbf{x}) + 256M|\mathcal{N}|\alpha J(\mathbf{x}) \enspace ,$$

where the function $\tilde{F}$ interpolates between the multilinear extension F of f and its symmetrized version G, and is given by

$$\tilde{F}(\mathbf{x}) = (1 - \phi(D(\mathbf{x}))) \cdot F(\mathbf{x}) + \phi(D(\mathbf{x})) \cdot G(\mathbf{x}) \enspace .$$

Here, $D(\mathbf{x}) \triangleq \|\mathbf{x} - \bar{\mathbf{x}}\|_2^2$, and $\phi\colon \mathbb{R}_{\geq 0} \to [0,1]$ is a function which is defined using the following lemma.

Lemma C.11 (Lemma 3.7 of [52])**.** *For any $\alpha, \beta > 0$, there is $\delta > (0, \beta)$ and a function $\phi\colon \mathbb{R}_{\geq 0} \to [0,1]$ with an absolutely continuous first derivative such that*

- *For $t \in [0, \delta]$, $\phi(t) = 1$.*
- *For $t \geq \beta$, $\phi(t) < e^{-1/\alpha}$.*
- *For all $t \geq 0$, $|t\phi'(t)| \leq 4\alpha$.*
- *For almost all $t \geq 0$, $|t^2\phi''(t)| \leq 10\alpha$.*

Vondrák [52] proved that the above functions $\hat{F}$ and $\hat{G}$ have all the properties guaranteed by Lemma C.7 for the δ whose existence is guaranteed by Lemma C.11 when the values of α and β are set to be $\alpha = \frac{\varepsilon}{2000M|\mathcal{N}|^3}$ and $\beta = \frac{\varepsilon}{16M|\mathcal{N}|}$. Moreover, the proof of [52] continues to work as long as $\alpha \leq \frac{\varepsilon}{2000M|\mathcal{N}|^3}$ and $\beta \leq \frac{\varepsilon}{16M|\mathcal{N}|}$. Therefore, we assume below that $\alpha = \min\{1, \frac{\varepsilon}{2000M|\mathcal{N}|^3}\}$ and $\beta = \min\{\alpha^2, \frac{\varepsilon}{16M|\mathcal{N}|}\}$, and we prove only the part of Lemma C.9 that is not stated in the guarantees of Lemma C.7, which is Property 4 of the lemma. We begin by showing that the function $\hat{G}$ indeed has this property.

Lemma C.12. *For every two vectors $\mathbf{x}, \mathbf{y} \in [0,1]^{\mathcal{N}}$ obeying $\mathbf{x} \leq \mathbf{y}$, $m \cdot \hat{G}(\mathbf{x}) \leq \hat{G}(\mathbf{y})$.*

Proof. Consider the random sets $\mathtt{R}(\bar{\mathbf{x}})$ and $\mathtt{R}(\bar{\mathbf{y}})$. Since $\bar{\mathbf{x}} \leq \bar{\mathbf{y}}$, the set $\mathtt{R}(\bar{\mathbf{y}})$ stochastically dominates $\mathtt{R}(\bar{\mathbf{x}})$. In other words, one can correlate the randomness of these sets in a way that does not alter their distributions, but guarantees that the inclusion $\mathtt{R}(\bar{\mathbf{x}}) \subseteq \mathtt{R}(\bar{\mathbf{y}})$ holds deterministically. Assuming this done, we get

$$m \cdot G(\mathbf{x}) = m \cdot F(\bar{\mathbf{x}}) = m \cdot \mathbb{E}[f(\mathtt{R}(\bar{\mathbf{x}}))] \leq \mathbb{E}[f(\mathtt{R}(\bar{\mathbf{y}}))] = F(\bar{\mathbf{y}}) = G(\mathbf{y}) \enspace, \tag{3}$$

where the inequality follows from the linearity of the expectation and the m-monotonicity of f.

Observe now that for every element $u \in \mathcal{N}$, the partial derivative of J with respect to z_u at any point $\mathbf{z} \in [0,1]^{\mathcal{N}}$ is

$$\frac{\partial J(\mathbf{z})}{\partial z_u} = 3|\mathcal{N}| - 2\sum_{v \in \mathcal{N}} z_v \geq |\mathcal{N}| \geq 0 \enspace.$$

Hence, the inequality $\mathbf{x} \leq \mathbf{y}$ implies $m \cdot J(\mathbf{x}) \leq J(\mathbf{x}) \leq J(\mathbf{y})$. Together with Inequality (3), this implies the lemma. □

One can observe that the arguments used to prove Inequality(3) in the proof of the last lemma also show that $m \cdot F(\mathbf{x}) \leq F(\mathbf{y})$, which is a fact that we use below. However, proving that $\hat{F}$ also has this property (and therefore, obeys Property 4 of Lemma C.9) is more involved. As a first step towards this goal, we bound the gradient of

$$\tilde{F}(\mathbf{x}) - F(\mathbf{x}) = \phi(D(\mathbf{x})) \cdot [G(\mathbf{x}) - F(\mathbf{x})] \enspace.$$

The following lemma does that in the regime in which $D(\mathbf{x})$ is small, and the next lemma handles the other regime.

Lemma C.13. *For every element $u \in \mathcal{N}$ and vector $\mathbf{x} \in [0,1]^{\mathcal{N}}$ obeying $D(\mathbf{x}) \leq \beta$, the absolute value of the partial derivative $\frac{\partial\{\phi(D(\mathbf{x}))\cdot[G(\mathbf{x})-F(\mathbf{x})]\}}{\partial x_u}$ is at most $72\sqrt{\beta}M|\mathcal{N}| \leq 72\alpha M|\mathcal{N}|$.*

Proof. Observe that

$$\begin{aligned}\frac{\partial\{\phi(D(\mathbf{x})) \cdot [G(\mathbf{x}) - F(\mathbf{x})]\}}{\partial x_u} ={}& \phi'(D(\mathbf{x})) \cdot \frac{\partial D(\mathbf{x})}{\partial x_u} \cdot [G(\mathbf{x}) - F(\mathbf{x})] \\ &+ \phi(D(\mathbf{x})) \cdot \left[\frac{\partial G(\mathbf{x})}{\partial x_u} - \frac{\partial F(\mathbf{x})}{\partial x_u}\right] \enspace.\end{aligned}$$

To use this equation to bound the absolute value of the left hand side, we need to make some observations. First, Lemma 3.6 of [52] shows that $\|\nabla D(\mathbf{x})\|_2 = 2\sqrt{D(\mathbf{x})}$, which implies

$$\frac{\partial D(\mathbf{x})}{\partial x_u} \leq \|\nabla D(\mathbf{x})\|_2 = 2\sqrt{D(\mathbf{x})} \enspace .$$

Additionally, Lemma 3.5 of [52] shows that $|G(\mathbf{x}) - F(\mathbf{x})| \leq 8M|\mathcal{N}| \cdot D(\mathbf{x})$, and therefore,

$$\begin{aligned}\left|\phi'(D(\mathbf{x})) \cdot \frac{\partial D(\mathbf{x})}{\partial x_u} \cdot [G(\mathbf{x}) - F(\mathbf{x})]\right| &\leq |\phi'(D(\mathbf{x}))| \cdot \left|\frac{\partial D(\mathbf{x})}{\partial x_u}\right| \cdot |G(\mathbf{x}) - F(\mathbf{x})| \\ &\leq |\phi'(D(\mathbf{x}))| \cdot 2\sqrt{D(\mathbf{x})} \cdot 8M|\mathcal{N}| \cdot D(\mathbf{x}) \\ &= |D(\mathbf{x}) \cdot \phi'(D(\mathbf{x}))| \cdot 16M|\mathcal{N}| \cdot \sqrt{D(\mathbf{x})} \\ &\leq 64\alpha\sqrt{\beta}M|\mathcal{N}| \enspace ,\end{aligned}$$

where the second inequality follows from Lemma C.11 and our assumption that $D(\mathbf{x}) \leq \beta$.

We now observe that

$$\left|\phi(D(\mathbf{x})) \cdot \left[\frac{\partial G(\mathbf{x})}{\partial x_u} - \frac{\partial F(\mathbf{x})}{\partial x_u}\right]\right| = \phi(D(\mathbf{x})) \cdot \left|\frac{\partial G(\mathbf{x})}{\partial x_u} - \frac{\partial F(\mathbf{x})}{\partial x_u}\right|$$
$$\leq \phi(D(\mathbf{x})) \cdot \|\nabla G(\mathbf{x}) - \nabla F(\mathbf{x})\|_2 \leq \phi(D(\mathbf{x})) \cdot 8M|\mathcal{N}| \cdot \sqrt{D(\mathbf{x})} \leq 8\sqrt{\beta}M|\mathcal{N}| \enspace ,$$

where the second inequality holds since Lemma 3.5 of [52] shows that $\|\nabla G(\mathbf{x}) - F(\mathbf{x})\|_2 \leq 8M|\mathcal{N}| \cdot \sqrt{D(\mathbf{x})}$; and the last inequality holds by our assumption that $D(\mathbf{x}) \leq \beta$ and by recalling that the range of ϕ is $[0, 1]$.

Combining all the above yields

$$\left|\frac{\partial\{\phi(D(\mathbf{x})) \cdot [G(\mathbf{x}) - F(\mathbf{x})]\}}{\partial x_u}\right| \leq 64\alpha\sqrt{\beta}M|\mathcal{N}| + 8\sqrt{\beta}M|\mathcal{N}| \leq 72\sqrt{\beta}M|\mathcal{N}| \enspace ,$$

where the second inequality holds since $\alpha \leq 1$. □

Lemma C.14. *For every element $u \in \mathcal{N}$ and vector $\mathbf{x} \in [0,1]^{\mathcal{N}}$ obeying $D(\mathbf{x}) \geq \beta$, the absolute value of the partial derivative $\frac{\partial\{\phi(D(\mathbf{x}))\cdot[G(\mathbf{x})-F(\mathbf{x})]\}}{\partial x_u}$ is at most $72\alpha M|\mathcal{N}|^{3/2}$.*

Proof. Repeating the proof of Lemma C.13, except for the use of the inequality $D(\mathbf{x}) \leq \beta$ (which does not hold in the current lemma) and the inequality $\phi(\mathbf{x}) \leq 1$ (which too weak for our current purpose), we get

$$\left|\phi(D(\mathbf{x})) \cdot \left[\frac{\partial G(\mathbf{x})}{\partial x_u} - \frac{\partial F(\mathbf{x})}{\partial x_u}\right]\right| \leq 64\alpha M|\mathcal{N}| \cdot \sqrt{D(\mathbf{x})} + |\phi(D(\mathbf{x}))| \cdot 8M|\mathcal{N}| \cdot \sqrt{D(\mathbf{x})} \enspace .$$

The expression $\phi(D(\mathbf{x}))$ can be upper bounded by $e^{-1/\alpha} \leq \alpha$ by Lemma C.11. Also, $D(\mathbf{x}) = \|\mathbf{x} - \bar{\mathbf{x}}\|_2^2 \leq |\mathcal{N}|$. The lemma now follows by plugging these two upper bounds into the previous inequality. □

Corollary C.15. *For every element $u \in \mathcal{N}$ and vector $\mathbf{x} \in [0,1]^{\mathcal{N}}$, the absolute value of the partial derivative $\frac{\partial\{\phi(D(\mathbf{x}))\cdot[G(\mathbf{x})-F(\mathbf{x})]\}}{\partial x_u}$ is at most $72\alpha M|\mathcal{N}|^{3/2}$.*

The last corollary implies that $\tilde{F}$ can be presented as the sum of F and a component that changes slowly. Therefore, if we add to $\tilde{F}$ a function that increases quickly enough (as is done to define $\hat{F}$), then we should get a function that can be represented as F plus a monotone component. This is the intuition formalized in the proof of the next lemma.

Lemma C.16. *The function $\hat{F}(\mathbf{x}) - F(\mathbf{x})$ has non-negative partial derivatives for every $\mathbf{x} \in [0,1]^{\mathcal{N}}$.*

Proof. By the definition of $\hat{F}(\mathbf{x})$,

$$\hat{F}(\mathbf{x}) - F(\mathbf{x}) = \tilde{F}(\mathbf{x}) - F(\mathbf{x}) + 256M|\mathcal{N}|\alpha J(\mathbf{x}) \enspace .$$

By Corollary C.15 and the observation that all the partial derivatives of $J(\mathbf{x})$ are at least $|\mathcal{N}|$ (see the proof of Lemma C.12), the last equality implies, for every element $u \in \mathcal{N}$,

$$\frac{\partial[\hat{F}(\mathbf{x}) - F(\mathbf{x})]}{\partial x_u} \geq -72\alpha M|\mathcal{N}|^{3/2} + 256\alpha M|\mathcal{N}|^2 \geq 0 \enspace .$$ □

We are now ready to show that $\hat{F}$ obeys Property 4 of Lemma C.9.

Lemma C.17. *For every two vectors* $\mathbf{x}, \mathbf{y} \in [0,1]^{\mathcal{N}}$ *obeying* $\mathbf{x} \leq \mathbf{y}$*,* $m \cdot \hat{F}(\mathbf{x}) \leq \hat{F}(\mathbf{y})$.

Proof. Note that $\overline{\mathbf{1}_{\varnothing}} = \mathbf{1}_{\varnothing}$, which implies that $G(\mathbf{1}_{\varnothing}) = F(\mathbf{1}_{\varnothing})$, and therefore,

$$\tilde{F}(\mathbf{1}_{\varnothing}) - F(\mathbf{1}_{\varnothing}) = \phi(D(\mathbf{1}_{\varnothing})) \cdot [G(\mathbf{1}_{\varnothing}) - F(\mathbf{1}_{\varnothing})] = 0 \enspace .$$

Plugging this observation into the definition of $\hat{F}$ now gives

$$\hat{F}(\mathbf{1}_{\varnothing}) - F(\mathbf{1}_{\varnothing}) = \tilde{F}(\mathbf{1}_{\varnothing}) - F(\mathbf{1}_{\varnothing}) + 256M|\mathcal{N}|\alpha J(\mathbf{1}_{\varnothing}) = 256M|\mathcal{N}|\alpha J(\mathbf{1}_{\varnothing}) \enspace .$$

Since all the first partial derivatives of $\hat{F}(\mathbf{z}) - F(\mathbf{z})$ are non-negative by Lemma C.16, the last inequality implies

$$\hat{F}(\mathbf{y}) - F(\mathbf{y}) \geq \hat{F}(\mathbf{x}) - F(\mathbf{x}) \geq 256M|\mathcal{N}|\alpha J(\mathbf{x}) \geq 0 \enspace .$$

Hence,

$$m\cdot\hat{F}(\mathbf{x}) \leq m\cdot[F(\mathbf{x})+\hat{F}(\mathbf{y})-F(\mathbf{y})] \leq F(\mathbf{y})+m\cdot[\hat{F}(\mathbf{y})-F(\mathbf{y})] \leq F(\mathbf{y})+[\hat{F}(\mathbf{y})-F(\mathbf{y})] = \hat{F}(\mathbf{y}) \enspace ,$$

where the second inequality holds by the discussion immediately after the proof of Lemma C.12, and the last inequality holds since $m \leq 1$ and $\hat{F}(\mathbf{y}) - F(\mathbf{y}) \geq 0$. □

C.3 Proof of Lemma 3.3

Lemma 3.3. *The problem* $\max\{f(S) \mid S \in \mathcal{F}\}$ *has a symmetry gap of* $\frac{1}{2-m}$.

Proof. Observe that our definition of $\mathcal{F}$ implies that $P(\mathcal{F}) = [0,1]^{\mathcal{N}}$. Therefore,

$$\max\{F(\mathbf{x}) \mid \mathbf{x} \in P(\mathcal{F})\} = \max\{F(\mathbf{x}) \mid \mathbf{x} \in [0,1]^{\mathcal{N}}\} = \max\{f(S) \mid S \subseteq \mathcal{N}\} = 1 \enspace , \tag{4}$$

where the second equality holds since, for every vector $\mathbf{x}$, $F(\mathbf{x})$ is a convex combination of values of f for subsets of $\mathcal{N}$; and on the other hand, for every set $S \subseteq \mathcal{N}$, $f(S) = F(\mathbf{1}_S)$.

Observe now that the definition of f implies that

$$\begin{aligned} F(\mathbf{x}) &= m[1 - (1-x_u)(1-x_v)] + (1-m)\cdot[x_u(1-x_v) + x_v(1-x_u)] \\ &= x_u + x_v - x_u x_v(2-m) \enspace . \end{aligned}$$

Since $\bar{\mathbf{x}}$ is a vector that has the value $(x_u + x_v)/2$ in both its coordinates, if we we use the shorthand $y = (x_u + x_v)/2$, then we get

$$F(\bar{\mathbf{x}}) = 2y - (2-m)y^2 \enspace .$$

This expression is maximized for $y = 1/(2-m)$, and the maximum attained for this y is

$$\frac{2}{2-m} - \frac{(2-m)}{(2-m)^2} = \frac{1}{2-m} \enspace .$$

Since the value $y = 1/(2-m)$ is obtained, for example, when $\mathbf{x} = (y, y) \in [0,1]^{\mathcal{N}}$, the above implies

$$\max\{F(\bar{\mathbf{x}}) \mid \mathbf{x} \in P(\mathcal{F})\} = \frac{1}{2-m} \enspace .$$

Together with Equation (4), this implies the lemma. □

D Inapproximability and Proofs of Section 4

In this section we state and analyze the algorithms used to prove the results given in Section 4. We also state and prove in Section D.3 the inapproximability result mentioned in Section 4.

D.1 Analysis of the Greedy Algorithm

In this section we prove Theorem 4.1, which we repeat here for convenience.

Theorem 4.1. *The Greedy algorithm (Algorithm 2) has an approximation ratio of at least $m(1 - 1/e)$ for the problem of maximizing a non-negative m-monotone submodular function subject to a cardinality constraint.*

The greedy algorithm starts with an empty solution, and then augments this solution in k iterations (recall that k is the maximum cardinality allowed for a feasible solution). Specifically, in iteration i, the algorithm adds to the current solution the element u_i with the best (largest) marginal contribution with respect to the current solution—but only if this addition does not decrease the value of the solution. A formal description of the greedy algorithm appears as Algorithm 2. Note that in this description the solution of the algorithm after i iterations, for every integer $0 \leq i \leq n$, is denoted by A_i.

Algorithm 2: The Greedy Algorithm (f, k)

```
1 Let $A_0 \leftarrow \varnothing$.
2 for $i = 1$ to $k$ do
3     Let $u_i$ be the element of $\mathcal{N} \setminus A_{i-1}$ maximizing $f(u_i \mid A_{i-1})$.
4     if $f(u_i \mid A_{i-1}) \geq 0$ then Let $A_i \leftarrow A_{i-1} + u_i$.
5     else Let $A_i \leftarrow A_{i-1}$.
6 return $A_k$.
```

Our first step towards proving Theorem 4.1 is the following lemma, which lower bounds the increase in the value of $f(A_i)$ as a function of i. Specifically, the lemma shows that this increase is significant as long as there is a significant gap between between $f(A_{i-1})$ and $m \cdot f(OPT)$, where OPT is an arbitrary optimal solution.

Lemma D.1. *For every integer $1 \leq i \leq k$, $f(A_i) - f(A_{i-1}) \geq k^{-1}[m \cdot f(OPT) - f(A_{i-1})]$.*

Proof. We need to distinguish between two cases. Consider first the case in which $f(u_i \mid A_{i-1}) \geq 0$. In this case,

$$\begin{aligned} f(A_i) - f(A_{i-1}) = f(u_i \mid A_{i-1}) &\geq \frac{|OPT \setminus A_{i-1}|}{k} \cdot f(u_i \mid A_{i-1}) \\ &\geq \frac{|OPT \setminus A_{i-1}|}{k} \cdot \max_{u \in OPT \setminus A_{i-1}} f(u \mid A_{i-1}) \geq \frac{\sum_{u \in OPT \setminus A_{i-1}} f(u \mid A_{i-1})}{k} \\ &\geq \frac{f(OPT \cup A_{i-1}) - f(A_{i-1})}{k} \geq \frac{m \cdot f(OPT) - f(A_{i-1})}{k} \enspace , \end{aligned}$$

where the first inequality holds since $|OPT \setminus A_{i-1}| \leq |OPT| \leq k$ because OPT is a feasible solution, the second inequality is due to the way used by the greedy algorithm to choose the element u_i, the penulatimate inequality follows from the submodularity of f, and the last inequality holds since f is m-monotone.

Consider now the case in which $f(u_i \mid A_{i-1}) < 0$. In this case, $f(A_i) - f(A_{i-1}) = 0$ because $A_i = A_{i-1}$. Furthermore, repeating the arguments used to prove the above inequality yields

$$\begin{aligned} m \cdot f(OPT) - f(A_{i-1}) &\leq |OPT \setminus A_{i-1}| \cdot \max_{u \in OPT \setminus A_{i-1}} f(u \mid A_{i-1}) \\ &\leq |OPT \setminus A_{i-1}| \cdot \max_{u \in \mathcal{N} \setminus A_{i-1}} f(u \mid A_{i-1}) \leq 0 \enspace . \end{aligned}$$ □

Rearranging the last lemma, we get the following inequality.

$$m \cdot f(OPT) - f(A_i) \leq (1 - 1/k) \cdot [m \cdot f(OPT) - f(A_{i-1})] \enspace . \tag{5}$$

This inequality bounds the rate in which the gap between $m \cdot f(OPT)$ reduces as a function of i. This allows us to prove Theorem 4.1.

Proof of Theorem 4.1. Combining Inequality (5) for every integer $1 \leq i \leq k$ yields

$$m \cdot f(OPT) - f(A_k) \leq (1 - 1/k)^k \cdot [m \cdot f(OPT) - f(A_0)] \enspace .$$

Rearranging this inequality, we get

$$f(A_k) \geq m \cdot f(OPT) - m \cdot (1 - 1/k)^k \cdot [f(OPT) - f(A_0)] \geq m \cdot \left(1 - \frac{1}{e}\right) \cdot f(OPT) \enspace ,$$

where the last inequality follows from the non-negativity of f and the inequality $(1 - 1/k)^k \leq \frac{1}{e}$. □

D.2 Analysis of Random Greedy

In this section we prove Theorem 4.2, which we repeat here for convenience.

Theorem 4.2. *Random Greedy (Algorithm 3) has an approximation ratio of at least* $m(1 - 1/e) + (1 - m) \cdot (1/e)$ *for the problem of maximizing a non-negative* m*-monotone submodular function subject to a cardinality constraint.*

Like the standard greedy algorithm from Section D.1, the Random Greedy algorithm starts with an empty solution, and then augments it in k iterations. Specifically, in iteration i the algorithm finds a set M_i of at most k elements whose total marginal contribution with respect to the current solution is maximal. Then, at most one element of M_i is added to the algorithm's current solution in a random way guaranteeing that every element of M_i is added to the solution with probability exactly $1/k$. A formal presentation of the Random Greedy algorithm appears as Algorithm 3. Note that in this presentation the solution of the algorithm after i iterations is denoted by A_i.

Algorithm 3: Random Greedy (f, k)

```
1 Let $A_0 \leftarrow \varnothing$.
2 for $i = 1$ to $k$ do
3     Let $M_i \leftarrow \arg\max_{B \subseteq \mathcal{N} \setminus A_{i-1}, |B| \leq k} \{\sum_{u \in B} f(u \mid A_{i-1})\}$.
4     with probability $(1 - |M_i|/k)$ do
5         $A_i \leftarrow A_{i-1}$.
6     otherwise
7         Let $u_i$ be a uniformly random element of $M_i$.
8         Set $A_i \leftarrow A_{i-1} + u_i$.
9 return $A_k$.
```

We start the analysis of the Random Greedy algorithm with the following lemma.

Lemma D.2. *For every integer* $0 \leq i \leq k$ *and element* $u \in \mathcal{N}$, $\Pr[u \in A_i] \leq 1 - (1 - 1/k)^i$.

Proof. Note that in each iteration i of Algorithm 3, any element $u \in \mathcal{N} \setminus A_{i-1}$ is added to the current solution with probability of at most $1/k$. Hence,

$$\Pr[u \in A_i] = 1 - \Pr[u \notin A_i] = 1 - \prod_{j=1}^{i} \Pr[u \notin A_j \mid u \notin A_{j-1}] \leq 1 - (1 - 1/k)^i \enspace . \qquad □$$

Plugging the guarantee of the last lemma into Lemma 2.1 yields the following lower bound on the expected value of $A_i \cup OPT$.

Corollary D.3. *For every integer* $0 \leq i \leq k$, $\mathbb{E}[f(A_i \cup OPT)] \geq [1 - (1 - m) \cdot (1 - (1 - \frac{1}{k})^i)] \cdot f(OPT) = m \cdot f(OPT) + (1 - m)(1 - \frac{1}{k})^i \cdot f(OPT)$.

Using the last corollary we are now ready to prove Theorem 4.2.

Proof of Theorem 4.2. Let $\mathcal{E}_{i-1}$ be an arbitrary possible choice for the random decisions of Random Greedy during its first $i-1$ iterations. Observe that, conditioned on $\mathcal{E}_{i-1}$ happening,

$$\begin{aligned}\mathbb{E}[f(A_i) - f(A_{i-1})] &= \frac{\sum_{u \in M_i} f(u \mid A_{i-1})}{k} \\ &\geq \frac{\sum_{u \in OPT \setminus A_{i-1}} f(u \mid A_{i-1})}{k} \geq \frac{f(A_{i-1} \cup OPT) - f(A_{i-1})}{k} \enspace ,\end{aligned}$$

where the first inequality follows from the choice of M_i by the algorithm, and the second inequality follows from submodularity. Taking now expectation over the choice $\mathcal{E}_{i-1}$ that realized, the last inequality yields

$$\begin{aligned}\mathbb{E}[f(A_i) - f(A_{i-1})] &\geq \frac{\mathbb{E}[f(A_{i-1} \cup OPT)] - \mathbb{E}[f(A_{i-1})]}{k} \qquad (6) \\ &\geq \frac{m \cdot f(OPT) + (1-m)(1-\frac{1}{k})^{i-1} \cdot f(OPT) - \mathbb{E}[f(A_{i-1})]}{k} \enspace ,\end{aligned}$$

where the second inequality is due to Corollary D.3.

The last inequality lower bounds the expected increase in the value of the solution of Random Greedy in every iteration. This implies also a lower bound on the expected value of $f(A_i)$. To complete the proof of the theorem, we need to prove a closed form for this implied lower bound, which we do by induction. Specifically, let us prove by induction on i that

$$\mathbb{E}[f(A_i)] \geq \left[m \cdot \left(1 - \left(1 - \frac{1}{k}\right)^i\right) + (1-m) \cdot \frac{i}{k} \cdot \left(1 - \frac{1}{k}\right)^{i-1}\right] \cdot f(OPT) \qquad (7)$$

for every integer $0 \leq i \leq k$, which implies the theorem by plugging $i = k$ because $(1 - 1/k)^k \leq 1/e \leq (1 - 1/k)^{k-1}$.

For $i = 0$, Inequality (7) holds since the non-negativity of f guarantees that $f(A_0) \geq 0 = [(1-m) \cdot (\frac{0}{k}) \cdot (1 - \frac{1}{k})^{-1} + m \cdot (1 - (1 - \frac{1}{k})^0)] \cdot f(OPT)$. Consider now some integer $0 < i \leq k$, and let us prove Inequality (7) for this value of i assuming that its holds for $i - 1$. By Inequality (6),

$$\begin{aligned}\mathbb{E}[f(A_i)] &= \mathbb{E}[f(A_{i-1})] + \mathbb{E}[f(A_i) - f(A_{i-1})] \\ &\geq \mathbb{E}[f(A_{i-1})] + \frac{m \cdot f(OPT) + (1-m)(1-\frac{1}{k})^{i-1} \cdot f(OPT) - \mathbb{E}[f(A_{i-1})]}{k} \\ &= \left(1 - \frac{1}{k}\right) \cdot \mathbb{E}[f(A_{i-1})] + \frac{m + (1-m)(1-\frac{1}{k})^{i-1}}{k} \cdot f(OPT) \enspace .\end{aligned}$$

Plugging the induction hypothesis into the last inequality, we get

$$\begin{aligned}\mathbb{E}[f(A_i)] &\geq \left(1 - \frac{1}{k}\right) \cdot \left[m \cdot \left(1 - \left(1 - \frac{1}{k}\right)^{i-1}\right) + (1-m) \cdot \frac{i-1}{k} \cdot \left(1 - \frac{1}{k}\right)^{i-2}\right] \cdot f(OPT) \\ &\qquad + \frac{m + (1-m)(1-\frac{1}{k})^{i-1}}{k} \cdot f(OPT) \\ &= \left[m\left(1 - \left(1 - \frac{1}{k}\right)^i\right) + (1-m) \cdot \frac{i}{k} \cdot \left(1 - \frac{1}{k}\right)^{i-1}\right] \cdot f(OPT) \enspace . \qquad \square\end{aligned}$$

D.3 Inapproximability for a Cardinality Constraint

In this section we state and prove the inapproximability result stated in Section 4.

Theorem D.4. *For any constant $\varepsilon > 0$, no polynomial time algorithm can obtain an approximation ratio of*

$$\min_{\alpha \in [0,1]} \frac{\max_{x \in [0,1]}\{\alpha(mx^2 + 2x - 2x^2) + 2(1-\alpha)(1 - e^{x-1})(1 - (1-m)x)\}}{\max\{1, 2(1-\alpha)\}} + \varepsilon$$

for the problem of maximizing a non-negative m-monotone submodular function subject to a cardinality constraint.

We prove Theorem D.4 using the symmetry gap technique, and specifically, via our extension of this technique proved in Theorem 3.2. To use this theorem, we need to construct an instance of our problem in which there is a large gap between the values of the best (general) solution and the best symmetric solution. Our instance is based on an instance constructed by Oveis Gharan and Vondrák [46]. However, the objective function in the original instance of [46] is not m-monotone for any $m > 0$, and therefore, we need to modify it so that it becomes m-monotone for a value $m \in [0, 1]$ of our choosing.

Fix some positive integer value r to be determined later and some value $\alpha \in [0, 1]$. The ground set of the instance we construct is $\mathcal{N} = \{a, b\} \cup \{a_i, b_i \mid i \in [r]\}$, and the constraint of the instance is a cardinality constraint allowing a feasible solution to include up to 2 elements. The objective function of our instance is the function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ defined by $f(S) = \alpha \cdot f_1(S) + (1-\alpha)[f_2(S) + f_3(S)]$, where

$$f_1(S) = m \cdot \mathbf{1}[S \cap \{a, b\} \neq \varnothing] + (1 - m) \cdot (|S \cap \{a, b\}| \bmod 2) \enspace ,$$
$$f_2(S) = \mathbf{1}[S \cap \{a_i \mid i \in [r]\} \neq \varnothing] \cdot (1 - (1 - m) \cdot \mathbf{1}[a \in S])$$

and

$$f_3(S) = \mathbf{1}[S \cap \{b_i \mid i \in [r]\} \neq \varnothing] \cdot (1 - (1 - m) \cdot \mathbf{1}[b \in S]) \enspace .$$

Let us denote the above described instance of submodular maximization subject to a cardinality constraint by $\mathcal{I}$. We begin the analysis of $\mathcal{I}$ by proving some properties of its objective function.

Lemma D.5. *The objective function f of $\mathcal{I}$ is non-negative, m-monotone and submodular.*

Proof. We prove below that the functions f_1, f_2 and f_3 have the properties stated in the lemma. This implies that f also has these properties by Observation 2.2 and the well-known closure of the class of submodular functions to multiplication by a non-negative constant and addition (see, e.g., Lemma 1.2 of [4]). The function f_1 is identical to the function proved in Section 3 to have the properties stated in the lemma, and the functions f_2 and f_3 are identical to each other up to switching the roles of a with b and a_i with b_i. Therefore, to prove that both f_2 and f_3 have the properties stated by the lemma it suffices to show that f_2 has these properties, which we do in the rest of this proof.

Clearly, f_2 is non-negative. To see that f_2 is a submodular function, note that

- For every set $S \subseteq \mathcal{N} - a$, $f_2(a \mid S) = -\mathbf{1}[S \cap \{a_i \mid i \in [r]\} \neq \varnothing] \cdot (1 - m)$.
- For every integer $1 \leq i \leq r$ and set $S \subseteq \mathcal{N} - a_i$, $f_2(a_i \mid S) = \mathbf{1}[S \cap \{a_i \mid i \in [r]\} = \varnothing] \cdot (1 - (1 - m) \cdot \mathbf{1}[a \in S])$.
- For every element $u \in (\mathcal{N} - a) \setminus \{a_i \mid i \in [r]\}$ and set $S \subseteq \mathcal{N} - u$, $f_2(u \mid S) = 0$.

Since all the above marginal contributions are down-monotone functions of S (i.e., functions whose value can only decrease when elements are added to S), the function f_2 is submodular.

It remains to argue why f_2 is m-monotone. Consider any two sets $S \subseteq T \subseteq \mathcal{N}$. If $f_2(S) = 0$, then the inequality $m \cdot f(S) \leq f(T)$ follows from the non-negativity of f_2. Therefore, consider the case in which $f_2(S) > 0$, which implies that $S \cap \{a_i \mid i \in [r]\} \neq \varnothing$; and therefore, $f_2(S) = (1 - (1 - m) \cdot \mathbf{1}[a \in S]) \leq 1$. Since S is a subset of T, we also get $f_2(T) = (1 - (1 - m) \cdot \mathbf{1}[a \in T]) \geq m$, and hence, $m \cdot f_2(S) \leq m \cdot 1 = m \leq f_2(T)$. □

A cardinality constraint is symmetric in the sense that the feasibility of a set depends only on the number of elements in it, and is completely independent of the identity of these elements. Let us now denote by $\mathcal{G}$ the group of permutations of $\mathcal{N}$ that are equivalent to applying any number of the following two steps: (1) switching a with b and a_i with b_i for every $i \in [r]$, or (2) switching a_i with a_j for two integers $i, j \in [r]$. The first step preserves the value of f because it simply switches the values of f_2 and f_3, while leaving the value of f_1 unaffected; and the second step preserves the value of f since it deals with elements that both f_1 and f_3 ignore, and f_2 treats in the same way. Hence, for every set $S \subseteq \mathcal{N}$ and permutation $\sigma \in \mathcal{G}$, we have $f(S) = f(\sigma(S))$, which implies the following observation.

Observation D.6. *The instance $\mathcal{I}$ is strongly symmetric with respect to $\mathcal{G}$.*

To use Theorem 3.2, we still need to bound the symmetry gap of $\mathcal{I}$, which we do next.

Lemma D.7. *The symmetry gap of $\mathcal{I}$ is at most*

$$\begin{aligned}&\frac{\max_{x\in[0,1]}\{\alpha(mx^2+2x-2x^2)+2(1-\alpha)[1-(1-(1-x)/r)^r](1-(1-m)x)\}}{\max\{1,2(1-\alpha)\}}\\ \leq{}& \frac{\max_{x\in[0,1]}\{\alpha(mx^2+2x-2x^2)+2(1-\alpha)(1-e^{x-1})(1-(1-m)x)\}}{\max\{1,2(1-\alpha)\}}+2/r \enspace .\end{aligned}$$

Proof. Two possible feasible solutions for $\mathcal{I}$ are the sets $\{a, b_1\}$ and $\{a_1, b_1\}$ whose values according to f are 1 and $2(1-\alpha)$, respectively. Therefore, the value of the optimal solution for $\mathcal{I}$ is at least $\max\{1, 2(1-\alpha)\}$. Since the symmetry gap is the ratio between the value of the best symmetric solution and the value of the best solution, to prove the lemma it remains to argue that the best symmetric solution for $\mathcal{I}$ has a value of $\max_{x\in[0,1]}\{\alpha(mx^2+2x-2x^2)+2(1-\alpha)[1-(1-(1-x)/r)^r](1-(1-m)x)\}$.

We remind the reader that a symmetric solution for $\mathcal{I}$ is $\bar{\mathbf{y}} = \mathbb{E}_{\sigma\in\mathcal{G}}[\mathbf{y}]$ for some vector $\mathbf{y}\in[0,1]^{\mathcal{N}}$ obeying $\|\mathbf{y}\|_1 \leq 2$. Since a and b can be exchanged with each other by the permutations of $\mathcal{G}$, the values of the coordinates of a and b in $\bar{\mathbf{y}}$ must be equal to each other. Similarly, every two elements of $\{a_i, b_i \in i \in [r]\}$ can be exchanged by the permutations of $\mathcal{G}$, and therefore, the values of the coordinates of these elements in $\bar{\mathbf{y}}$ must all be identical. Thus, any symmetric solution $\bar{\mathbf{y}}$ can be represented as

$$\bar{y}_u = \begin{cases} x & \text{if } u = a \text{ or } u = b \enspace , \\ z & \text{if } u \in \{a_i, b_i \mid i \in [r]\}\end{cases}$$

for some values $x, z \in [0,1]$ obeying $2x + 2rz \leq 2$ (or equivalently, $z \leq (1-x)/r$). The value of this solution (according to the multilinear extension F of f) is

$$\begin{aligned}&\alpha[m(1-(1-x)^2)+2(1-m)x(1-x)]+2(1-\alpha)(1-(1-z)^r)(1-(1-m)x)\\ ={}&\alpha(mx^2+2x-2x^2)+2(1-\alpha)(1-(1-z)^r)(1-(1-m)x) \enspace .\end{aligned}$$

Since this expression is a non-decreasing function of z, the maximum value of any symmetry solution for $\mathcal{I}$ is

$$\begin{aligned}&\max_{\substack{x,z\in[0,1]\\ z\leq(1-x)/r}}\{\alpha(mx^2+2x-2x^2)+2(1-\alpha)(1-(1-z)^r)(1-(1-m)x)\}\\ ={}&\max_{x\in[0,1]}\{\alpha(mx^2+2x-2x^2)+2(1-\alpha)[1-(1-(1-x)/r)^r](1-(1-m)x)\} \enspace . \qquad \square\end{aligned}$$

Since any refinement of a cardinality constraint is a cardinality constraint over a larger ground set, plugging Lemma D.5, Observation D.6 and Lemma D.7 into Theorem 3.2 yields the following corollary.

Corollary D.8. *For every constant $\varepsilon' > 0$, no polynomial time algorithm for maximizing a non-negative m-monotone submodular function subject to a cardinality contraint obtains an approximation ratio of*

$$\frac{\max_{x\in[0,1]}\{\alpha(mx^2+2x-2x^2)+2(1-\alpha)(1-e^{x-1})(1-(1-m)x)\}}{\max\{1,2(1-\alpha)\}}+2/r+\varepsilon' \enspace .$$

Theorem D.4 now follows from the last corollary by choosing $\varepsilon' = \varepsilon/2$, $r = \lceil 4/\varepsilon\rceil$ and

$$\alpha = \arg\min_{\alpha'\in[0,1]} \frac{\max_{x\in[0,1]}\{\alpha'(mx^2+2x-2x^2)+2(1-\alpha')(1-e^{x-1})(1-(1-m)x)\}}{\max\{1,2(1-\alpha')\}} \enspace .$$

E Inapproximability and Proofs of Section 5

In this section we state and analyze the algorithms used to prove the results given in Section 5. We also state and prove in Section E.4 the inapproximability result mentioned in Section 5.

E.1 Analysis of the Greedy algorithm

A version of the greedy algorithm designed for matroid constraints appears as Algorithm 4. This algorithm starts with an empty solution, and then iteratively adds elements to this solution, where the element added in each iteration is the element with the largest marginal contrition with respect to the current solution among all the elements whose addition to the solution does not violate feasibility. The algorithm terminates when no additional elements can be added to the solution without decreasing its value.

Algorithm 4: The Greedy Algorithm (for a Matroid Constraint) $(f, \mathcal{M} = (\mathcal{N}, \mathcal{I}))$

```
1 Let $A_0 \leftarrow \varnothing$, and $i \leftarrow 0$.
2 while true do
3     Let $u_{i+1}$ be the element of $\{v \in \mathcal{N} \setminus A_i \mid A_i + v \in \mathcal{I}\}$ maximizing $f(u_{i+1} \mid A_i)$.
4     if $f(u_{i+1} \mid A_i) \geq 0$ then Let $A_{i+1} \leftarrow A_i + u_{i+1}$, and then, increase $i$ by 1.
5     else return $A_i$.
```

Theorem 5.1. *The Greedy algorithm (Algorithm 4) has an approximation ratio of at least $m/2$ for maximizing a non-negative m-monotone submodular function subject to a matroid constraint.*

Proof. Lemma 3.2 of [27] shows that the greedy algorithm outputs a solution S of value at least $f(S \cup OPT)/2$ for the problem of maximizing a non-negative submodular function f subject to a matroid constraint, where OPT is an optimal solution for the problem.[6] The theorem now follows since for an m-monotone function f we are guaranteed to have $f(S \cup OPT) \geq m \cdot f(OPT)$. □

E.2 Analysis of Measured Continuous Greedy

In this section, we reanalyze the Measured Continuous Greedy algorithm of [22] in view of the monotonicity ratio. Given a non-negative submodular function $f: 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ and a down-closed solvable[7] convex body $P \subseteq [0,1]^{\mathcal{N}}$, Measured Continuous Greedy is an algorithm designed to find a vector $\mathbf{x} \in P$ that approximately maximize $F(\mathbf{x})$, where F is the multilinear extension of f. Specifically, we prove the following theorem.

Theorem E.1. *Given a non-negative m-monotone submodular function $f: 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$, a solvable down-close convex body $P \subseteq [0,1]^{\mathcal{N}}$ and a parameter $T \geq 0$, Measured Continuous Greedy outputs a vector $\mathbf{x} \in [0,1]^{\mathcal{N}}$ obeying $F(\mathbf{x}) \geq [m(1-e^{-T}) + (1-m)Te^{-T}] \cdot f(OPT)$, where F is the multilinear extension of f and OPT is the set maximizing f among all sets whose characteristic vectors belong to P. Furthermore, $\mathbf{x} \in P$ whenever $T \in [0,1]$.*

We note that Feldman et al. [22] discussed conditions that guarantee that $\mathbf{x}$ belongs to P also for some values of T that are larger than 1. However, the above stated form of Theorem E.1 already suffices to prove Theorem 5.2. Let us explain why this is the case. When P is the matroid polytope $P_{\mathcal{M}}$ of a matroid $\mathcal{M}$, there are algorithms called Pipage Rounding [9] and Swap Rounding [11] that, given a vector $\mathbf{x} \in P$ produce a set S that is independent in $\mathcal{M}$ and also obeys $\mathbb{E}[f(S)] \geq F(\mathbf{x}) - o(1) \cdot f(OPT)$. Therefore, one can obtain an algorithm for maximizing f subject to the matroid $\mathcal{M}$ by executing Measured Continuous Greedy with $P = P_{\mathcal{M}}$ and $T = 1$, and then applying either Pipage Rounding or Swap Rounding to the resulting vector; which yields an algorithm with the properties specified by Theorem 5.2.

We now describe the version of Measured Continuous Greedy that we analyze (given as Algorithm 5). For simplicity, we chose to analyze a continuous version of this algorithm that assumes direct access to the multilinear extension F of the objective function rather than just to the objective function itself. We refer the reader to [22] for details about discretizing the algorithm and avoiding the assumption of direct access to F. We also note that the $o(1)$ error term in the approximation guarantee stated in

[6]In fact, Lemma 3.2 of [27] proves a more general result for p-set systems, but it implies the stated result since matroids are 1-set systems.

[7]A body $P \subseteq [0,1]^{\mathcal{N}}$ is *solvable* if one can efficiently optimize linear functions subject to it, and is *down-closed* if $\mathbf{y} \in P$ implies $\mathbf{x} \in P$ for every vector $\mathbf{x} \in [0,1]^{\mathcal{N}}$ obeying $\mathbf{x} \leq \mathbf{y}$ (this inequality should be understood to hold coordinate-wise).

Theorem E.1 is due to these issues. Our description of Measured Continuous Greedy requires some additional notation, namely, given two vectors $\mathbf{x}$ and $\mathbf{y}$, we denote by $\mathbf{x} \vee \mathbf{y}$ their coordinate-wise maximum and by $\mathbf{x} \odot \mathbf{y}$ their coordinate-wise multiplication.

Measured Continuous Greedy starts at "time" 0 with the empty solution, and improves this solution during the time interval $[0, T]$. We denote the solution of the algorithm at time t by $\mathbf{y}(t)$. At every time $t \in [0, T]$, the algorithm calculates a vector $\mathbf{w}$ whose u-coordinate is the gain that can be obtained by increasing this coordinates in the solution $\mathbf{y}(t)$ to be 1 (i.e., $w_u(t) = F(\mathbf{y}(t) \vee 1_{\{u\}}) - F(\mathbf{y}(t))$). Then, the algorithm finds a vector $\mathbf{x}(t) \in P$ that maximizes the objective function $\mathbf{w}(t) \cdot \mathbf{x}(t)$, and adds to the solution $\mathbf{y}(t)$ an infinitesimal part of $(\mathbf{1}_{\mathcal{N}} - \mathbf{y}(t)) \odot \mathbf{x}(t)$ (to understand where the last expression comes from, we note that when $\mathbf{x}$ is integral, fully adding $(\mathbf{1}_{\mathcal{N}} - \mathbf{y}(t)) \odot \mathbf{x}(t)$ to $\mathbf{y}(t)$ sets to 1 all the coordinates that are 1 in $\mathbf{x}(t)$, which matches the "spirit" of the definition of $\mathbf{w}$).

Algorithm 5: Measured Continuous Greedy(f, P, T)

```
1 Let $\mathbf{y}(0) \leftarrow 1_{\varnothing}$.
2 foreach $t \in [0, T)$ do
3     For each $u \in \mathcal{N}$, let $w_u(t) \leftarrow F(\mathbf{y}(t) \vee 1_{\{u\}}) - F(\mathbf{y}(t))$.
4     Let $\mathbf{x}(t) \leftarrow \arg\max_{\mathbf{x} \in P}\{\mathbf{w}(t) \cdot \mathbf{x}\}$.
5     Increase $\mathbf{y}(t)$ at a rate of $\frac{d\mathbf{y}(t)}{dt} = (\mathbf{1}_{\mathcal{N}} - \mathbf{y}(t)) \odot \mathbf{x}(t)$.
6 return $y(T)$.
```

The first step in the analysis of Measured Continuous Greedy is bounding the maximum value of the coordinates of the solution $\mathbf{y}(t)$.

Lemma E.2. *For every $t \in [0, T]$, $\|\mathbf{y}(t)\|_\infty \leq 1 - e^{-t}$.*

Proof. Fix an arbitrary element $u \in \mathcal{N}$, and let us explain why $y_u(t) \leq 1 - e^{-t}$. By Line 5 of Algorithm 5, $y_u(t)$ obeys the differential inequality

$$\frac{dy_u(t)}{dt} = (1 - y_u(t)) \cdot x_u(t) \leq 1 - y_u(t) \enspace ,$$

and the solution of this differential inequality for the initial condition $y_u = 0$ is

$$y_u(t) \leq 1 - e^{-t} \enspace . \qquad \square$$

We are now ready to prove Theorem E.1

Proof of E.1. Recall that $\mathbf{x}(t)$ is a vector inside P for every time $t \in [0, T]$, and since P is down-closed, $(\mathbf{1}_{\mathcal{N}} - \mathbf{y}(t)) \odot \mathbf{x}(t)$ and $\mathbf{1}_{\varnothing}$ both belong to P as well. This means that for $T \leq 1$ the vector $\mathbf{y}(T) = (1 - T) \cdot \mathbf{1}_{\varnothing} + \int_0^T (\mathbf{1}_{\mathcal{N}} - y(t)) \odot \mathbf{x}(t) dt$ is a convex combination of vectors in P, and therefore belongs to P by the convexity of P.

It remains to lower bound the value of $F(\mathbf{y}(T))$. By the chain rule,

$$\frac{dF(\mathbf{y}(t))}{dt} = \sum_{u \in \mathcal{N}} \left(\frac{dy_u(t)}{dt} \cdot \frac{\partial F(\mathbf{y})}{\partial y_u}\Big|_{\mathbf{y}=\mathbf{y}(t)} \right) = \sum_{u \in \mathcal{N}} \left((1 - y_u(t)) \cdot x_u(t) \cdot \frac{\partial F(\mathbf{y})}{\partial y_u}\Big|_{\mathbf{y}=\mathbf{y}(t)} \right) \enspace .$$

Since F is multilinear, its partial derivative with respect to a single coordinate is equal to the difference between the value of the function for two different values of this coordinate over the difference between these values. Plugging this observation into the previous inequality yields

$$\frac{dF(\mathbf{y}(t))}{dt} = \sum_{u \in \mathcal{N}} \left((1 - y_u(t)) \cdot x_u(t) \cdot \frac{F(\mathbf{y}(t) \vee 1_{\{u\}}) - F(\mathbf{y}(t))}{1 - y_u(t)} \right) = \mathbf{x}(t) \cdot \mathbf{w}(t) \enspace .$$

One possible candidate to be $\mathbf{x}(t)$ is $\mathbf{1}_{OPT}$. Hence, by the definition of $\mathbf{x}(t)$, $\mathbf{x}(t) \cdot \mathbf{w}(t) \geq \mathbf{1}_{OPT} \cdot \mathbf{w}(t)$. Combining this inequality with the previous one, we get

$$\begin{aligned}
\frac{dF(\mathbf{y}(t))}{dt} &\geq \mathbf{1}_{OPT} \cdot \mathbf{w}(t) = \sum_{u \in OPT} \left[F(\mathbf{y}(t) \vee 1_{\{u\}}) - F(\mathbf{y}(t))\right] \\
&\geq F(\mathbf{y}(t) \vee \mathbf{1}_{OPT}) - F(\mathbf{y}(t)) \geq [1 - (1-m) \cdot \|\mathbf{y}(t)\|_\infty] \cdot f(OPT) - F(\mathbf{y}(t)) \\
&\geq [1 - (1-m)(1-e^{-t})] \cdot f(OPT) - F(\mathbf{y}(t)) \\
&= [m + (1-m)e^{-t}] \cdot f(OPT) - F(\mathbf{y}(t)) \enspace ,
\end{aligned}$$

where the second inequality holds by the submodularity of f, the penultimate inequality holds by Lemma 2.1, and the last inequality follows from Lemma E.2.

Solving the differential inequality that we got for the initial condition $F(\mathbf{y}(0)) \geq 0$ (which holds by the non-negativity of f) yields

$$F(y(t)) \geq \left[m(1-e^{-t}) + (1-m)te^{-t}\right] \cdot f(OPT) \enspace ,$$

and the theorem now follows by plugging $t = T$. $\square$

E.3 Analysis of Random Greedy for Matroids

In this section we prove Theorem 5.3, which we repeat here for convenience.

Theorem 5.3. *For every $\varepsilon \in (0,1)$, Random Greedy for Matroids (Algorithm 6) has an approximation ratio of at least $\frac{1+m+e^{-2/(1-m)}}{4} - \varepsilon - o_k(1)$ for the problem of maximizing a non-negative m-monotone submodular function subject to a matroid constraint (except in the case of $m = 1$ in which the approximation ratio is $1/2 - \varepsilon - o_k(1)$), where $o_k(1)$ represents a term that diminishes with k.*

To prove the theorem, we first need to state the algorithm it refers to. Towards this goal, let us assume that the ground set $\mathcal{N}$ contains a set D of $2k$ "dummy" elements that are known to the algorithm and have the following two properties.

- $f(S) = (S \setminus D)$ for every set $S \subseteq \mathcal{N}$.
- $S \in \mathcal{I}$ if and only if $S \setminus D \in \mathcal{I}$ and $|S| \leq k$.

This assumption is useful since it allows us to assume that the optimal solution OPT is a base of $\mathcal{M}$, and thus, simplifies the description of our algorithm (Random Greedy for Matroids). We can justify our assumption using the following procedure: (i) add $2k$ dummy elements to the ground set, (ii) extend f and $\mathcal{I}$ according to the above properties, (iii) execute Random Greedy for Matroids on the resulting instance, and (iv) remove from the output of the algorithm any dummy elements that end up in it. This procedure guarantees that any approximation guarantee obtained by Random Greedy for Matroids using our assumption can be obtained also without the assumption.

Our version of the Random Greedy for Matroids algorithm is given as Algorithm 6. Like the original version of the algorithm (due to [6]), our version starts with a base of $\mathcal{M}$ consisting only of dummy elements, and then modifies it in a series of iterations. In each iteration i, the algorithm starts with a solution S_{i-1}, and then identifies a base M_i of $\mathcal{M}$ whose elements have the largest total marginal contribution with respect to S_{i-1} (M_i is also required to be disjoint from S_{i-1}). The algorithm then picks a uniformly random element $u_i \in S_{i-1}$, and adds it to the solution S_{i-1} at the expense of an element $g_i(u_i)$ of S_{i-1} given by a function g_i that is chosen carefully (the existence of such a function follows, for example, from Corollary 39.12a of [48]).

As mentioned above, our version of Random Greedy for Matroids differs compared to the version of [6] in two respects. The first modification is in the number of iterations that the algorithm makes. To get the result of Buchbinder et al. [6], it suffices to use k iterations. However, the optimal number of iterations increases with m, and therefore, our version of the algorithm uses k/ε iterations for some parameter $\varepsilon \in (0,1)$ (we assume without loss of generality that k/ε is integral; otherwise, we can replace ε with a value which is smaller than ε by at most a factor of 2 and has this property). Furthermore, since we do not want to assume knowledge of m in the algorithm, we use a number of iterations that is appropriate for $m = 1$, which requires us to make the second modification to the algorithm; namely, we check whether replacing $g(u_i)$ with u_i is beneficial, and make the swap only

Algorithm 6: Random Greedy for Matroids($f, \mathcal{M} = (\mathcal{N}, \mathcal{I}), \varepsilon$)

```
1 Initialize $S_0$ to be an arbitrary base containing only elements of $D$.
2 for $i = 1$ to $k/\varepsilon$ do
3     Let $M_i \subseteq N$ be a base of $\mathcal{M}$ that contains only elements of $\mathcal{N} \setminus S_{i-1}$ and maximizes
      $\sum_{u \in M_i} f(u \mid S_{i-1})$ among all such bases.
4     Let $g_i$ be a function mapping each element of $M_i$ to an element of $S_{i-1}$ obeying
      $S_{i-1} - g_i(u) + u \in \mathcal{I}$ for every $u \in S_{i-1}$.
5     Let $u_i$ be a uniformly random element from $M_i$. if $f(S_{i-1} - g_i(u_i) + u_i) > f(S_{i-1})$
      then Let $S_i \leftarrow S_{i-1} - g_i(u_i) + u_i$.
6     else Let $S_i \leftarrow S_{i-1}$.
7 return $S_{k/\varepsilon}$.
```

if this is indeed the case. This guarantees that doing more iterations can never decrease the value of the algorithm's solution.

Since Theorem 5.3 is trivial for a constant k, we can assume in the analysis of Algorithm 6 that k is larger than any given constant. The first step in this analysis is proving the following lower bound on the expected value of $OPT \cup S_i$.

Observation E.3. *For every integer* $0 \leq i \leq k/\varepsilon$, $\mathbb{E}[f(OPT \cup S_i)] \geq \frac{1}{2}(1 + m + (1 - m)(1 - 2/k)^i) \cdot f(OPT)$.

Proof. For every integer $0 \leq i \leq k/\varepsilon$ and element $u \in \mathcal{N} \setminus D$, let $p_{u,i}$ denote the probability u belongs to S_i. We would like to argue that when $i > 0$, we have $p_{u,i} \leq p_{u,i-1}(1 - 2/k) + 1/k$. To see why this is the case, note that u belongs to S_i only if one of the following happens: (i) u belongs to S_{i-1} and is not removed from the solution (happens with probability $p_{u,i-1}(1 - 1/k)$ since $g_i(u_i)$ is a uniformly random element of S_{i-1}), or (ii) u belongs to M_{i-1} and is chosen as u_i (happens with probability at most $(1 - p_{u,i})/k$). Therefore,

$$p_{u,i} \leq p_{u,i-1} \cdot (1 - 1/k) + (1 - p_{u,i-1})/k = p_{u,i-1} \cdot (1 - 2/k) + 1/k \quad . \tag{8}$$

Next, we aim to prove by induction that $p_{u,i} \leq \frac{1}{2}(1 - (1 - 2/k)^i)$ for every integer $0 \leq i \leq k/\varepsilon$. For $i = 0$, this is true since $u \in \mathcal{N} \setminus D$ implies that $p_{u,0} = 0 = \frac{1}{2}(1 - (1 - 2/k)^0)$. Assume now that the claim holds for $i - 1$, and let us prove it for $i \geq 1$. By the induction hypothesis and Inequality (8),

$$p_{u,i} \leq p_{u,i-1}(1 - 2/k) + 1/k \leq \tfrac{1}{2}(1 - (1 - 2/k)^{i-1})(1 - 2/k) + 1/k = \tfrac{1}{2}(1 - (1 - 2/k)^i) \quad .$$

The observation now follows since Lemma 2.1 guarantees that $\mathbb{E}[f(OPT \cup S_i)] = \mathbb{E}[f(OPT \cup (S_i \setminus D))] \geq (1 - (1 - m) \cdot \max_{u \in \mathcal{N} \setminus D} p_{i,u}) \cdot f(OPT)$. $\square$

Below we prove a lower bound on the value of the solution of Algorithm 6 after any number of iterations. However, to prove this lower bound we first need to prove the following technical observation.

Observation E.4. *For every positive integer* i,

$$\left(1 - \frac{2}{k}\right)^{i-1} \geq e^{-\frac{2i}{k}} - \frac{k}{i} \cdot o_k(1) \quad .$$

Proof. Note that

$$\begin{aligned} e^{-\frac{2i}{k}} = \left(e^{-\frac{2}{k}}\right)^i \leq \left(1 - \frac{2}{k} + \frac{4}{k^2}\right)^i \leq \left(1 - \frac{2}{k}\right)^i + \frac{4}{k^2} \cdot i \left(1 - \frac{2}{k} + \frac{4}{k^2}\right)^{i-1} \\ \leq \left(1 - \frac{2}{k}\right)^i + \frac{4i}{k^2}\left(1 - \frac{1}{k}\right)^{i-1} \leq \left(1 - \frac{2}{k}\right)^i + \frac{4i}{k^2} \cdot e^{-\frac{i-1}{k}} \quad , \end{aligned}$$

where the third inequality holds for $k \geq 4$, and the second inequality holds since the derivative of the function $(1 - 2/k + x)^i$ is $i(1 - 2/k + x)^{i-1}$, which implies

$$\left(1 - \frac{2}{k} + \frac{4}{k^2}\right)^i = \left(1 - \frac{2}{k}\right)^i + \int_0^{4/k^2} i(1 - 2/k + x)^{i-1} dx$$
$$\leq \left(1 - \frac{2}{k}\right)^i + \frac{4i}{k^2}(1 - 2/k + 4/k^2)^{i-1} dx \enspace .$$

To complete the proof of the observation, it remains to note that, since the maximum of the function $x^2 e^{-x}$ for $x \geq 0$ is $4e^{-2}$,

$$\frac{4i}{k^2} \cdot e^{-\frac{i-1}{k}} \leq \frac{16e^{-2}}{i} \cdot e^{\frac{1}{k}} = \frac{k}{i} \cdot o_k(1) \enspace . \qquad \square$$

We are now ready to prove the promised lower bound on the value of the solution S_i of Algorithm 6 after any number of iterations.

Lemma E.5. *For every integer* $0 \leq i \leq k/\varepsilon$*,*

$$\mathbb{E}[f(S_i)] \geq \left[\frac{1+m}{4} \cdot \left(1 - e^{-\frac{2i}{k}}\right) + \frac{(1-m)i}{2k} \cdot e^{-\frac{2i}{k}} - o_k(1)\right] \cdot f(OPT) \enspace .$$

Proof. For $i = 0$ the lemma follows from the non-negativity of f since the right hand side of the inequality that we need to prove is non-positive for $i = 0$. Together with Observation E.4, this implies that it suffices to prove the following inequality

$$\mathbb{E}[f(S_i)] \geq \left[\frac{1+m}{4} \cdot \left(1 - \left(1 - \frac{2}{k}\right)^i\right) + \frac{(1-m)i}{2k} \cdot \left(1 - \frac{2}{k}\right)^{i-1}\right] \cdot f(OPT) \enspace , \qquad (9)$$

and the rest of the proof is devoted to this goal.

Fix an arbitrary integer $1 \leq i \leq k/\varepsilon$. We would like to derive a lower bound on the expected marginal contribution of the element u_i to the set S_{i-1}, and an upper bound on the expected marginal contribution of the element $g(u_i)$ to the set $S_{i-1} \setminus g(u_i)$. Let A_{i-1} be an event fixing all random choices of Algorithm 6 up to iteration $i - 1$ (including), and let $\mathcal{A}_{i-1}$ be the set of all possible A_{i-1} events. Conditioned on any event $A_{i-1} \in \mathcal{A}_{i-1}$, the sets S_{i-1} and M_i becomes deterministic, and we can define M'_i as a set containing the elements of $OPT \setminus S_{i-1}$ plus enough dummy elements of $D \setminus S_{i-1}$ to make the size of M'_i exactly k. Then,

$$\mathbb{E}[f(u \mid S_{i-1}) \mid A_{i-1}] = \frac{\sum_{u \in M_i} f(u \mid S_{i-1})}{k} \geq \frac{\sum_{u \in M'_i} f(u \mid S_{i-1})}{k}$$
$$= \frac{\sum_{u \in OPT \setminus S_{i-1}} f(u \mid S_{i-1})}{k} \geq \frac{f(OPT \cup S_{i-1}) - f(S_{i-1})}{k} \enspace ,$$

where S_i, M_i and M'_i represent here their values conditioned on A_i, the first inequality follows from the definition of M_i and the second inequality holds by the submodularity of f. Similarly,

$$\mathbb{E}[f(g(u_i) \mid S_{i-1} - g(u_i)) \mid A_{i-1}] = \frac{\sum_{u \in M_i} f(g(u_i) \mid S_{i-1} - g(u))}{k}$$
$$\leq \frac{f(S_{i-1}) - f(\varnothing)}{k} \leq \frac{f(S_{i-1})}{k} \enspace ,$$

where the first inequality follows from the submodularity of f. Taking expectation over the event A_{i-1}, we get

$$\mathbb{E}[f(u_i \mid S_{i-1})] \geq \frac{\mathbb{E}[f(OPT \cup S_{i-1})] - \mathbb{E}[f(S_{i-1})]}{k}$$
$$\geq \frac{\frac{1}{2}(1 + m + (1-m)(1 - 2/k)^{i-1}) \cdot f(OPT) - \mathbb{E}[f(S_{i-1})]}{k} \enspace ,$$

where the last inequality is due to Observation E.3, and

$$\mathbb{E}[f(g(u_i) \mid S_{i-1} - g(u_i))] \leq \frac{\mathbb{E}[f(S_{i-1})]}{k} \enspace .$$

Combing the last two inequalities now yields

$$\begin{aligned}\mathbb{E}[f(S_i)] &\geq \mathbb{E}[f(S_{i-1} - g(u_i) + u_i)] \qquad (10)\\ &= \mathbb{E}[f(S_{i-1})] + \mathbb{E}[f(u_i \mid S_{i-1} - g(u_i))] - \mathbb{E}[f(g(u_i) \mid S_{i-1} - g(u_i)]\\ &\geq \mathbb{E}[f(S_{i-1})] + \mathbb{E}[f(u_i \mid S_{i-1})] - \mathbb{E}[f(g(u_i) \mid S_{i-1} - g(u_i)]\\ &\geq \left(1 - \frac{2}{k}\right) \cdot \mathbb{E}[f(S_{i-1})] + \frac{\frac{1}{2}(1 + m + (1-m)(1-2/k)^{i-1}) \cdot f(OPT)}{k} \enspace ,\end{aligned}$$

where the first inequality follows from the submodularity of f since $g(u_i) \neq u_i$ because $g(u_i) \in S_{i-1}$ and $u_i \in M_i$.

Since Inequality (10) holds for every integer $1 \leq i \leq k/\varepsilon$, we can use it repeatedly to get, for every integer $0 \leq i \leq k/\varepsilon$,

$$\begin{aligned}\mathbb{E}[f(S_i)] \geq \frac{1}{2k}\left[(1+m)\sum_{j=1}^{i}\left(1-\frac{2}{k}\right)^{i-j} + (1-m)\sum_{j=1}^{i}\left(1-\frac{2}{k}\right)^{i-1}\right] \cdot f(OPT)\\ + \left(1-\frac{2}{k}\right)^{i} \cdot f(S_0) \enspace .\end{aligned}$$

Since the non-negativity of f guarantees that $f(S_0) \geq 0$, the last inequality implies Inequality (9), and therefore, completes the proof of the lemma. □

One can show that the lower bound for $f(S_i)$ proved by Lemma E.5 is maximized when $i = k/(1-m)$. Unfortunately, we cannot simply plug this i value into the lower bound due to two issues: this value of i might not be integral, and this value of i might be larger than the number k/ε of iterations. The following two lemmata prove the approximation guarantee of Theorem 5.3 despite these issues, and together they complete the proof of the theorem.

Lemma E.6. *When $m \leq 1 - \varepsilon$, the approximation ratio of Algorithm 6 is at least*

$$\frac{1 + m + e^{-2/(1-m)}}{4} - o_k(1) \enspace .$$

Proof. Let $i' = \lfloor k/(1-m) \rfloor$. Due to the condition of the lemma, Algorithm 6 makes at least i' iterations. Furthermore, since Algorithm 6 makes a swap in its solution only when this swap is beneficial, the expected value of the output of the algorithm is at least

$$\begin{aligned}\mathbb{E}[f(S_{i'})] &\geq \left[\frac{1+m}{4} \cdot \left(1 - e^{-\frac{2i'}{k}}\right) + \frac{(1-m)i'}{2k} \cdot e^{-\frac{2i'}{k}} - o_k(1)\right] \cdot f(OPT)\\ &\geq \left[\frac{1+m}{4} \cdot \left(1 - e^{\frac{2}{k} - \frac{2}{1-m}}\right) + \frac{k-1}{2k} \cdot e^{-\frac{2}{1-m}} - o_k(1)\right] \cdot f(OPT)\\ &\geq \left[\frac{1+m}{4} \cdot \left(1 - e^{-\frac{2}{1-m}}\right) - \frac{e^{\frac{2}{k}} - 1}{2} + \frac{1}{2} \cdot e^{-\frac{2}{1-m}} - \frac{1}{2k} - o_k(1)\right] \cdot f(OPT) \enspace ,\end{aligned}$$

where the first inequality follows from Lemma E.5, and the second inequality holds since $k/(1-m) - 1 \leq i' \leq k/(1-m)$. Since the terms $\frac{e^{2/k}-1}{2}$ and $\frac{1}{2k}$ are both diminishing with k (and therefore, can be replaced with $o_k(1)$), the last inequality implies the lemma. □

Lemma E.7. *When $1 - \varepsilon \leq m < 1$, the approximation ratio of Algorithm 6 is at least*

$$\frac{1 + m + e^{-2/(1-m)}}{4} - \varepsilon - o_k(1) \enspace ,$$

and when $m = 1$ the approximation ratio of this algorithm is at least $1/2 - \varepsilon - o_k(1)$.

Proof. The output set of Algorithm 6 is $f(S_{k/\varepsilon})$. By Lemma E.5, the expected value of this set is at least

$$\begin{aligned}\mathbb{E}[f(S_{k/\varepsilon})] &\geq \left[\frac{1+m}{4} \cdot \left(1 - e^{-\frac{2}{\varepsilon}}\right) + \frac{1-m}{2\varepsilon} \cdot e^{-\frac{2}{\varepsilon}} - o_1(k)\right] \cdot f(OPT)\\ &\geq \left[\frac{1+m}{4} \cdot \left(1 - e^{-\frac{2}{\varepsilon}}\right) - o_k(1)\right] \cdot f(OPT)\\ &\geq \left[\frac{1+m}{4} \cdot \left(1 - \frac{1}{1+2/\varepsilon}\right) - o_k(1)\right] \cdot f(OPT)\\ &= \left[\frac{1+m}{2(\varepsilon+2)} - o_k(1)\right] \cdot f(OPT) \geq \left[\frac{1+m}{4} - \frac{\varepsilon}{4} - o_k(1)\right] \cdot f(OPT) \enspace ,\end{aligned}$$

where the third inequality holds since for every $x \geq 0$, $\ln(1/(1+x)) = \ln(1 - x/(1+x)) \geq -\frac{x/(1+x)}{1-x/(1+x)} = -x$.

The above inequality completes the proof for the case of $m = 1$. To complete the proof also for the case of $1 - \varepsilon \leq m < 1$, it suffice to observe that in this case

$$e^{-2/(1-m)} \leq e^{-2/\varepsilon} \leq \frac{1}{1+2/\varepsilon} \leq \frac{\varepsilon}{2} \enspace . \qquad \square$$

E.4 Inapproximability for a Matroid Constraints

In this section we state and prove the inapproximability result mentioned in Section 5.

Theorem E.8. *For any constant $\varepsilon > 0$, no polynomial time algorithm can obtain an approximation ratio of*

$$\min_{\alpha \in [0,1]} \max_{x \in [0,1/2]} \{\alpha(mx^2 + 2x - 2x^2) + 2(1-\alpha)(1 - e^{-1/2})(1 - (1-m)x)\} + \varepsilon$$

for the problem of maximizing a non-negative m-monotone submodular function subject to a matroid constraint.

The proof of Theorem E.8 is very similar to the proof of Theorem D.4. Recall that in Section D.3, we proved Theorem D.4 by constructing an instance $\mathcal{I}$ of submodular maximization subject to a cardinality constraint, and then applying Theorem 3.2 to this instance. The proof of Theorem E.8 is based on an instance $\mathcal{I}'$ of submoduar maximization subject to a matroid constrained that is identical to $\mathcal{I}$ except for the following difference. In $\mathcal{I}$, the constraint is a cardinality constraint allowing the selection of up to 2 elements from the ground set $\mathcal{N} = \{a, b\} \cup \{a_i, b_i \mid i \in [r]\}$. In $\mathcal{I}'$, we have instead a (simplified) partition matroid constraint allowing the selection of up to 1 element from $\{a, b\}$ and up to 1 element from $\{a_i, b_i \mid i \in [r]\}$.

Since the instances $\mathcal{I}$ and $\mathcal{I}'$ have the same objective function, the properties of this function stated in Lemma D.5 apply to both of them. Furthermore, one can verify that $\mathcal{I}'$ is strongly symmetric with respect to the group $\mathcal{G}$ of permutation defined in Section D.3. Therefore, we concentrate on analyzing the symmetry gap of $\mathcal{I}'$.

Lemma E.9. *The symmetry gap of $\mathcal{I}'$ is at most*

$$\begin{aligned}&\max_{x \in [0,1/2]} \{\alpha(mx^2 + 2x - 2x^2) + 2(1-\alpha)[1 - (1 - 1/(2r))^r](1 - (1-m)x)\}\\ \leq &\max_{x \in [0,1/2]} \{\alpha(mx^2 + 2x - 2x^2) + 2(1-\alpha)(1 - e^{-1/2})(1 - (1-m)x)\} + 1/(2r) \enspace .\end{aligned}$$

Proof. One possible feasible solution for $\mathcal{I}'$ is the set $\{a, b_1\}$ whose value according to f is 1. Therefore, the value of the optimal solution for $\mathcal{I}'$ is at least 1. Since the symmetry gap is the ratio between the value of the best symmetric solution and the value of the best solution, to prove the lemma it remains to argue that the best symmetric solution for $\mathcal{I}$ has a value of at most $\max_{x \in [0,1/2]}\{\alpha(mx^2 + 2x - 2x^2) + 2(1-\alpha)[1 - (1 - 1/(2r))^r](1 - (1-m)x)\}$.

We remind the reader that a symmetric solution for $\mathcal{I}'$ is $\bar{\mathbf{y}} = \mathbb{E}_{\sigma \in \mathcal{G}}[\mathbf{y}]$ for some vector $\mathbf{y} \in [0,1]^{\mathcal{N}}$ obeying $y_a + y_b \leq 1$ and $\sum_{i=1}^{r} y_{a_i} + y_{b_i} \leq 1$. Since a and b can be exchanged with each other by the permutations of $\mathcal{G}$, the values of the coordinates of a and b in $\bar{\mathbf{y}}$ must be equal to each other. Similarly, every two elements of $\{a_i, b_i \in i \in [r]\}$ can be exchanged by the permutations of $\mathcal{G}$, and therefore, the values of the coordinates of all these elements in $\bar{\mathbf{y}}$ must be all identical. Thus, any symmetric solution $\bar{\mathbf{y}}$ can be represented as

$$\bar{y}_u = \begin{cases} x & \text{if } u = a \text{ or } u = b \\ z & \text{if } u \in \{a_i, b_i \mid i \in [r]\} \end{cases} ,$$

for some values $x \in [0, 1/2]$ and $z \in [0, 1/(2r)]$. The value of this solution (according to the multilinear extension F of f) is

$$\begin{aligned} &\alpha[m(1-(1-x)^2) + 2(1-m)x(1-x)] + 2(1-\alpha)(1-(1-z)^r)(1-(1-m)x) \\ &= \alpha(mx^2 + 2x - 2x^2) + 2(1-\alpha)(1-(1-z)^r)(1-(1-m)x) \\ &\leq \alpha(mx^2 + 2x - 2x^2) + 2(1-\alpha)(1-(1-1/(2r))^r)(1-(1-m)x) \ . \end{aligned}$$

Therefore, one can obtain an upper bound on the value of the best symmetric solution for $\mathcal{I}'$ by taking the maximum of the last expression over all the values that x can take, which completes the proof of the lemma. □

Since any refinement of a (simplified) partition matroid constraint is a (generalized) partition matroid constraint on its own right, plugging Lemmata D.5 and Lemma E.9 into Theorem 3.2 yields the following corollary.

Corollary E.10. *For every constant $\varepsilon' > 0$, no polynomial time algorithm for maximizing a non-negative m-monotone submodular function subject to a matroid contraint obtains an approximation ratio of*

$$\max_{x \in [0,1/2]} \{\alpha(mx^2 + 2x - 2x^2) + 2(1-\alpha)(1-e^{-1/2})(1-(1-m)x)\} + 1/(2r) + \varepsilon' \ .$$

Theorem E.8 now follows from the last corollary by choosing $\varepsilon' = \varepsilon/2$, $r = \lceil \varepsilon^{-1} \rceil$ and

$$\alpha = \arg\min_{\alpha' \in [0,1]} \max_{x \in [0,1/2]} \{\alpha'(mx^2 + 2x - 2x^2) + 2(1-\alpha')(1-e^{-1/2})(1-(1-m)x)\} \ .$$

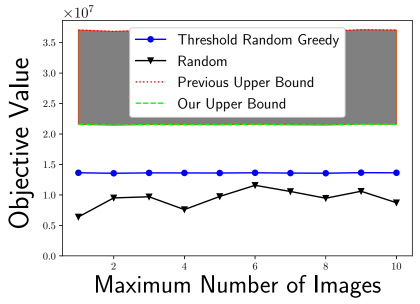
F Personalized Image Summarization

Consider a setting in which we get as input a collection $\mathcal{N}$ of images from ℓ disjoint categories (e.g., birds, dogs, cats) and the user specifies $r \in [\ell]$ categories, and then demands a subset of the images in these categories that summarize all the images of the categories. Following [40] again, to evaluate a given subset of images we use the function $f(S) = \sum_{u \in \mathcal{N}} \max_{v \in S} s_{u,v} - \frac{1}{|\mathcal{N}|} \sum_{u \in S} \sum_{v \in S} s_{u,v}$, where $s_{u,v}$ is a non-negative similarity between u and v.

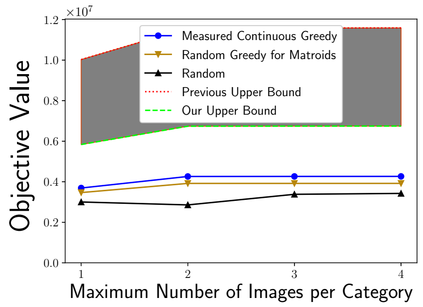
One can verify that the above function f is non-negative and submodular. Unfortunately, this function can have a very low monotonicity ratio. To compensate for this, we observe that most the analyses we described in the previous sections use the monotonicity ratio only to show that $f(S \cup T) \geq m \cdot f(S)$ for sets S and T that are feasible. This motivates the following weak version of the monotonicity ratio. We note that many continuous properties of set functions have such weak versions. For example, the original paper presenting the submodularity-ratio [16] presented in fact the weak version of this property, and the non-weak version was only formulated at a later point.

Definition F.1. Consider maximization of a non-negative function f subject to some constraint. In the context of this constraint, we say that f is *m-weakly monotone* if $f(S \cup T) \geq m \cdot f(S)$ holds for every two feasible sets S and T.

Theorem F.2. *The objective function f of personalized image summarization is $1 - \frac{2k}{|\mathcal{N}|}$-weakly monotone when the size of feasible solutions is at most k for some $1 \leq k \leq |\mathcal{N}|$.*



(a) Results for Personalized Image Summarization with a cardinality constraint for varying number of images in the summary produced.



(b) Results for Personalized Image Summarization with a matroid constraint. The x-axis is the number k of images allowed from each category.

Figure 3: Personalized Image Summerization Results

Proof. When $k \geq |\mathcal{N}|/2$, the theorem is trivial. Thus, we can assume below $k < |\mathcal{N}|/2$. Consider two feasible sets $S, T \in \mathcal{N}$, and let us lower bound $f(S \cup T)$.

$$\begin{aligned}
f(S \cup T) &= \sum_{u \in \mathcal{N}} \max_{v \in S \cup T} s_{u,v} - \frac{1}{|\mathcal{N}|} \sum_{u \in S \cup T} \sum_{v \in S \cup T} s_{u,v} \\
&\geq \sum_{u \in \mathcal{N}} \max_{v \in S \cup T} s_{u,v} - \frac{|S \cup T|}{|\mathcal{N}|} \sum_{u \in S \cup T} \max_{v \in S \cup T} s_{u,v} \\
&\geq \sum_{u \in \mathcal{N}} \max_{v \in S \cup T} s_{u,v} - \frac{2k}{|\mathcal{N}|} \sum_{u \in S \cup T} \max_{v \in S \cup T} s_{u,v} \\
&= \left(1 - \frac{2k}{|\mathcal{N}|}\right) \sum_{u \in \mathcal{N}} \max_{v \in S \cup T} s_{u,v} \geq \left(1 - \frac{2k}{|\mathcal{N}|}\right) \sum_{u \in \mathcal{N}} \max_{v \in S} s_{u,v} \enspace .
\end{aligned}$$

Using this lower bound, we now get

$$f(S) = \sum_{u \in E} \max_{v \in S} s_{u,v} - \frac{1}{|\mathcal{N}|} \sum_{u \in S} \sum_{v \in S} s_{u,v} \leq \sum_{u \in E} \max_{v \in S} s_{u,v} \leq \frac{f(S \cup T)}{1 - 2k/|\mathcal{N}|} \enspace ,$$

which completes the proof of the theorem since S and T have been chosen as arbitrary feasible sets. □

Our experiments for this setting are based on a subset of the CIFAR-10 dataset [33] including 10,000 Tiny Images. These images belong to 10 classes, with 1000 images per class. Each image consists of 32×32 RGB pixels represented by a 3072 dimensional vector. To compute the similarity $s_{u,v}$ between images, we used the dot product.

In our first experiment, we simply looked for a summary consisting of a limited number of images. Since this is a cardinality constraint, we again used the scarecrow algorithm Random and the accelerated versions mentioned in Section 6.1 of the algorithms from Section 4. In Figure 3a we depict the outputs of Threshold Random Greedy and Random for various limits on the number of images in the summary (like in Section 6.1 we omit the other non-scarecrow algorithms since their performance is essentially identical to the one of Threshold Random Greedy, and we refer the reader to Appendix I for more detail). Figure 3a also includes the upper bounds on the optimal solution obtained via the previous approximation ratio for Random Greedy and our improved approximation ratio (the area between the two upper bounds is shaded). We can see that the upper bound obtained via our improved approximation ratio is much tighter, and this upper bound also demonstrates that the gap between the non-scarecrow and the scarecrow algorithms is significant compared to the optimal solution.

In our second experiment, we looked for a summary containing up to k images from each category selected by the user for some parameter k (we assumed in the experiment that the user chose the categories: "airplane", "automobile" and "bird"). Since this is a (generalized partition) matroid constraint, in this experiment we used versions of the algorithms from Section 5. Specifically, we used Random Greedy for Matroids and an accelerated version of Measured Continuous Greedy based on the acceleration technique underlying the Accelerated Continuous Greedy of [1]. Additionally, we used in this experiment a scarecrow algorithm called Random that outputs a set containing a random selection of k images from each one of the chosen categories. The values of the outputs of all these algorithms are depicted in Figure 3b (values shown are averaged over 10 executions).

Figure 3b also includes upper bounds on the value of the optimal solution. The previous upper bound is computed based on the previously known approximation ratios of the algorithms, and our upper bound is computed based on the approximation ratios proved in Theorems 5.2 and 5.3 and the weak monotonicity ratio proved in Theorem F.2.[8] As is evident from the similarity between Figures 3a and 3b, our observations from the first experiment extend also the more general constraint considered in the current experiment.

G Ride-Share Optimization

In this application, given a set R of possible customer locations specified as (latitude, longitude) coordinate pairs, we aim to find a subset of these locations that will serve as waiting locations for drivers and minimizes the distance from each costumer to her closest driver. This problem can be modeled using the classical facility location problem, whose objective is known to be monotone and submodular. More formally, Mitrovic et al. [41] defined for every set T of locations the objective value $f(T)$ as

$$f(T) = \sum_{a \in R} \max_{b \in T} c(a, b) \enspace ,$$

where $c(a,b)$ is a convenience score defined by $c(a,b) \triangleq 2 - \frac{2}{1+e^{-200d(a,b)}}$, and $d(a,b) = |x_a - x_b| + |y_a - y_b|$ is the Manhattan distance between the points a and b.

One drawback of the above objective function is that it does not promote diversity in the set of chosen locations. For example, imagine a scenario where, due to congestion or road maintenance in a specific area, traffic in and out of this area is slow or completely blocked. If all the selected waiting locations happen to be inside the affected area (i.e., there is no diversity in the selected locations), it will be difficult for the drivers to move between the waiting locations and the customers. To avoid such unfavorable scenarios, a diversity component should be added to the objective function. However, when a diversity component is added, the function becomes non-monotone (but still submodular), making the approximation guarantees of state-of-the-art algorithms much lower, as is discussed above.

Using the monotonicity ratio, the effect of the diversity component on the approximation guarantee can be significantly reduced. For example, a natural way to add a diversity component is demonstrated by the next objective function.

$$f(T) = \sum_{a \in R} \max_{b \in T} c(a, b) - \frac{1}{|R|} \sum_{x \in T} \sum_{y \in T} c(x, y) \enspace .$$

One can note that the last function has the same form as the function discussed in Appendix F. Hence, by Theorem F.2, the previous function is $(1 - \frac{2k}{|R|})$-weakly monotone, where $1 \leq k \leq |R|$ is the maximum size of a feasible solution.

H Proofs of Section 6

In this section we prove the theorems from Section 6.

[8]From a purely formal point of view this upper bound is not fully justified since Measured Continuous Greedy is a rare example of an algorithm whose analysis cannot use in a black box fashion the weak monotonicity ratio instead of the monotonicity ratio. However, due to probabilistic concentration, we expect the upper bound to still hold up to at most a small error.

H.1 Proof of Theorem 6.1

Theorem 6.1. *The objective function f is monotone for $0 \leq \lambda \leq 1/2$ and $2(1-\lambda)$-monotone for $1/2 \leq \lambda \leq 1$.*

Proof. We first prove the first part of the theorem. Thus, we assume $\lambda \leq 1/2$, and we need to show that for arbitrary set $S \subseteq \mathcal{N}$ and element $u \in \mathcal{N} \setminus S$ the marginal contribution $f(u \mid S)$ is non-negative. This holds because

$$\begin{aligned} f(u \mid S) &= \sum_{v\in\mathcal{N}} s_{v,u} - \lambda\left[\sum_{v\in S} s_{u,v} + \sum_{v\in S} s_{v,u} + s_{u,u}\right] \\ &= \sum_{v\in\mathcal{N}} s_{v,u} - \lambda\left[2\sum_{v\in S} s_{v,u} + s_{u,u}\right] \geq \sum_{v\in\mathcal{N}} s_{v,u} - \sum_{v\in S} s_{v,u} - s_{u,u} \geq 0 \enspace, \end{aligned}$$

where the second equality holds because $s_{u,v} = s_{v,u}$, and the first inequality holds since $\lambda \leq 1/2$ in the case we consider and the $s_{u,v}$ values are non-negative.

It remains to prove the second part of the theorem. Thus, we assume from now on $\lambda \in [1/2, 1]$, and we consider two sets $S \subseteq T \subseteq \mathcal{N}$. To prove the theorem we need to show that $f(T) \geq 2(1-\lambda) \cdot f(S)$. The first step towards showing this is to prove the following lower bound on $f(S)$.

$$\begin{aligned} f(S) &= 2(1-\lambda)\cdot f(S) + (2\lambda-1)\cdot\left[\sum_{u\in\mathcal{N}}\sum_{v\in S} s_{u,v} - \lambda\sum_{u\in S}\sum_{v\in S} s_{u,v}\right] && (11) \\ &\geq 2(1-\lambda)\cdot f(S) + (2\lambda-1)\cdot\sum_{u\in T\setminus S}\sum_{v\in S} s_{u,v} \\ &= 2(1-\lambda)\cdot f(S) + (2\lambda-1)\cdot\sum_{u\in S}\sum_{v\in T\setminus S} s_{u,v} \enspace, && (12) \end{aligned}$$

where the inequality holds since $\lambda \leq 1$, and the second equality holds since $s_{u,v} = s_{v,u}$. Using this lower bound, we now get

$$\begin{aligned} f(T) &= f(S) + \sum_{u\in\mathcal{N}}\sum_{v\in T\setminus S} s_{u,v} - \lambda\left[\sum_{u\in S}\sum_{v\in T\setminus S} s_{u,v} + \sum_{u\in T\setminus S}\sum_{v\in S} s_{u,v} + \sum_{u\in T\setminus S}\sum_{v\in T\setminus S} s_{u,v}\right] \\ &= f(S) + \sum_{u\in\mathcal{N}}\sum_{v\in T\setminus S} s_{u,v} - \lambda\left[2\sum_{u\in S}\sum_{v\in T\setminus S} s_{u,v} + \sum_{u\in T\setminus S}\sum_{v\in T\setminus S} s_{u,v}\right] \\ &\geq f(S) + (1-2\lambda)\cdot\sum_{u\in S}\sum_{v\in T\setminus S} s_{u,v} \geq 2(1-\lambda)\cdot f(S) \enspace, \end{aligned}$$

where the first inequality holds since $\lambda \leq 1$, and the second inequality holds by Inequality (11). □

H.2 Proof of Theorem 6.2

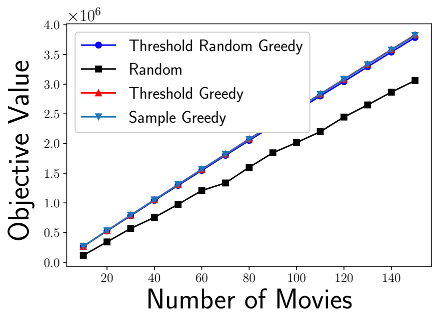
Theorem 6.2. *For $\beta \in (0, 1/2)$, the objective function F given by Equation (1) is $\frac{(1-2\beta)\cdot\alpha}{1+\alpha}$-monotone. Furthermore, when $\min_{\bar{0}\leq x\leq\mathbf{u}}(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}\mathbf{x}) \geq 0$, F is even $(1-2\beta)$-monotone.*

Proof. Fix two vectors $\bar{0} \leq \mathbf{x} \leq \mathbf{y} \leq \mathbf{u}$. We begin this proof by providing a lower bound on $F(\mathbf{y})$ and an upper bound on $F(\mathbf{x})$. The lower bound on $F(\mathbf{y})$ is as following.

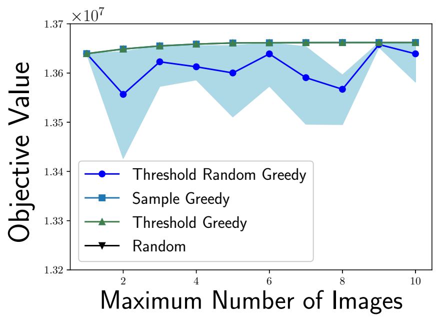
$$F(\mathbf{y}) = \frac{1}{2}\mathbf{y}^T\mathbf{H}\mathbf{y} + \mathbf{h}^T\mathbf{y} + c \geq \min_{\bar{0}\leq\mathbf{x}\leq\mathbf{u}}\left(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}\mathbf{x}\right) + c \enspace.$$

To get the upper bound on $F(\mathbf{x})$, we first need to prove an upper bound on c.

$$c \geq -\min_{0\leq\mathbf{x}\leq\mathbf{u}}\left(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x}\right) = -\min_{0\leq\mathbf{x}\leq\mathbf{u}}\left(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} - \beta\mathbf{u}^T\mathbf{H}\mathbf{x}\right) \geq -\left(\frac{1}{2} - \beta\right)\mathbf{u}^T\mathbf{H}\mathbf{u} \enspace.$$



(a) Performance of the various algorithms in the movie recommendation setting (for $\lambda = 0.75$).



(b) Performance of the non-scarecrow algorithms in the image summarization setting with a cardinality constraint. The shaded area represents the standard deviation of Threshold Random Greedy.

Figure 4: Comparing the performance of algorithms for a cardinality constraint in our experiments.

The promised upper bound on $F(\mathbf{x})$ now follows.

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x} + c \leq \mathbf{h}^T\mathbf{x} + c \leq \mathbf{h}^T\mathbf{u} + c = -\beta\mathbf{u}^T\mathbf{H}\mathbf{u} + c \leq \frac{\beta c}{1/2 - \beta} + c = \frac{c}{1 - 2\beta} \enspace,$$

where the first inequality holds since $\mathbf{H}$ is non-positive, and the second inequality holds since $\mathbf{h}$ is non-negative.

Recall now that $c = -M + \alpha|M|$, which implies

$$\min_{0 \leq \mathbf{x} \leq \mathbf{u}} \left(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x}\right) = M \geq -\frac{c}{1 + \alpha} \enspace,$$

and therefore,

$$F(\mathbf{y}) \geq -\frac{c}{1 + \alpha} + c = \frac{c\alpha}{1 + \alpha} \geq \frac{(1 - 2\beta)\alpha}{1 + \alpha} \cdot F(\mathbf{x}) \enspace.$$

It remains to consider the case in which $\min_{\bar{0} \leq \mathbf{x} \leq \mathbf{u}} \left(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}\mathbf{x}\right) \geq 0$. In this case

$$F(\mathbf{y}) \geq c \geq (1 - 2\beta) \cdot F(\mathbf{x}) \enspace. \qquad \square$$

I Additional Plots for Section 6

As discussed in Section 6, the various algorithms we use in the context of a cardinality constraint have very similar empirical performance. Figure 4a presents the performance of all these algorithms in the movie recommendation setting with the number of movies in the summery varying. One can observe that the lines of the three non-scarecrow algorithms almost overlap. Figure 4b presents the performance of the non-scarecrow algorithms in the image summarization setting. In this figure we had to ignore the scarecrow algorithm Random because otherwise the lines of the three non-scarecrows algorithms are indistinguishable. Furthermore, we had to zoom in on a very small range of y-axis values. Despite these steps, the lines of Sample Greedy and Threshold Greedy still completely overlap, but the large zoom allows us to see that Threshold Random Greedy is marginally worse.

J Maximizating DR-submodular Functions subject to a Polytope Constraint

There are (at least) two natural ways in which the notion of submodularity can be extended from set functions to continuous functions. The more restrictive of these is known as DR-submodularity (first defined by [3]). Given a domain $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_i$, where $\mathcal{X}_i$ is a closed range in $\mathbb{R}$ for every $i \in [n]$, a function $F \colon \mathcal{X} \to \mathbb{R}$ is *DR-submodular* if for every two vectors $\mathbf{a}, \mathbf{b} \in \mathcal{X}$, positive value k and coordinate $i \in [n]$ the inequality

$$F(\mathbf{a} + k\mathbf{e}_i) - F(\mathbf{a}) \geq F(\mathbf{b} + k\mathbf{e}_i) - F(\mathbf{b})$$

holds whenever $\mathbf{a} \leq \mathbf{b}$ and $\mathbf{b} + k\mathbf{e}_i \in \mathcal{X}$ (here and throughout the section $\mathbf{e}_i$ denotes the standard i-th basis vector, and comparison between two vectors should be understood to hold coordinate-wise). If F is continuously differentiable, then the above definition of DR-submodulrity is equivalent to ∇F being an antitone mapping from $\mathcal{X}$ to $\mathbb{R}^n$ (i.e., $\nabla F(\mathbf{a}) \geq \nabla F(\mathbf{b})$ for every two vectors $\mathbf{a}, \mathbf{b} \in \mathcal{X}$ that obey $\mathbf{a} \leq \mathbf{b}$). Moreover, when F is twice differentiable, it is DR-submodular if and only if its Hessian is non-positive at every vector $\mathbf{x} \in \mathcal{X}$.

In this section we consider the problem of maximizing a non-negative DR-submodular function $F \colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ subject to a solvable down-closed[9] convex body $P \subseteq \mathcal{X}$ (usually polytope) constraint. As is standard when dealing with problems of this kind, we assume that F is L-smooth, i.e., for every two vectors $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ it obeys

$$\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2$$

for some non-negative parameter L. Additionally, for simplicity, we assume that $\mathcal{X} = [0, 1]^n$. This assumption is without loss of generality because the natural mapping from $\mathcal{X}$ to $[0, 1]^n$ preserves all our results.

We analyze a variant of the Frank-Wolfe algorithm for the above problem due to [2] called Non-monotone Frank-Wolfe. This variant was motivated by the Measured Continuous Greedy algorithm studied in Section E.2, and its assumes access to the first order derivatives of F. The details of the algorithm we consider appear as Algorithm 7. This algorithm gets a quality control parameter $\varepsilon \in (0, 1)$, and it is assumed that ε^{-1} is an integer (if this is not the case, one can fix that by reducing ε by at most a factor 2). Algorithm 7 and its analysis also employ the notation defined in Section E.2, namely, given two vectors $\mathbf{x}, \mathbf{y}$, their coordinate-wise multiplication is denoted by $\mathbf{x} \odot \mathbf{y}$. Additionally, we denote by $\bar{0}$ and $\bar{1}$ the all zeros and all ones vectors, respectively.

Algorithm 7: Non-monotone Frank-Wolfe(ε)

```
1 Let $\mathbf{y}^{(0)} \leftarrow \bar{0}$ and $t = 0$.
2 while $t \leq 1$ do
3     $\mathbf{s}^{(t)} \leftarrow \arg\max_{\mathbf{x} \in P} \mathbf{x} \cdot ((\bar{1} - \mathbf{y}^{(t)}) \odot \nabla F(\mathbf{y}^{(t)}))$.
4     $\mathbf{y}^{(t+\varepsilon)} \leftarrow \mathbf{y}^{(t)} + \varepsilon \cdot (\bar{1} - \mathbf{y}^{(t)}) \odot \mathbf{s}^{(t)}$.
5     $t \leftarrow t + \varepsilon$.
6 return $\mathbf{y}^{(1)}$.
```

To analyze Algorithm 7 we need to define two additional parameters. The first parameter is the diameter $D = \max_{x \in P} \|\mathbf{x}\|_2$ of P, which is a standard parameter. The other parameter is the monotonicity ratio of F, which can be extended to the continuous setting we study in the following natural way.[10]

$$m = \inf_{\substack{\mathbf{x}, \mathbf{y} \in \mathcal{X} \\ \mathbf{x} \leq \mathbf{y}}} \frac{F(\mathbf{y})}{F(\mathbf{x})} \enspace ,$$

where the ratio $F(\mathbf{y})/F(\mathbf{x})$ should be understood to have a value of 1 whenever $F(\mathbf{x}) = 0$. Additionally, let us denote by $\mathbf{o}$ an arbitrary optimal solution for the problem described above. Using these definitions, we are now ready to state the result that we prove for Algorithm 7.

Theorem J.1. *When given a non-negative m-monotone DR-submodular function $F \colon \mathcal{X} \to \mathbb{R}_{\geq 0}$ and a down-closed solvable convex body $P \subseteq \mathcal{X}$, the Measured Greedy Frank-Wolfe algorithm (Algorithm 7) outputs a solution $\mathbf{y} \in P$ such that $F(\mathbf{y}) \geq [m(1 - 1/e) + (1 - m) \cdot (1/e)] \cdot F(\mathbf{o}) - \varepsilon L D^2$.*

[9]In Section E.2, down-closeness of was defined for the special case of $P \subseteq [0, 1]^{\mathcal{N}}$. More generally, a body $P \subseteq \mathcal{X}$ is down-closed if $\mathbf{b} \in P$ implies $\mathbf{a} \in P$ for every vector $\mathbf{a} \in \mathcal{X}$ obeying $\mathbf{a} \leq \mathbf{b}$.

[10]In Appendix 6.2 we showed how the monotonicity ratio can be extended to the particular continuous setting studied in that section. The definition of Appendix 6.2 is obtained from the more general definition we give here by setting $\mathcal{X} = \prod_{i=1}^{n} [0, u_i]$.

Our first objective towards proving Theorem J.1 is to lower bound the expression $F(\mathbf{o}+\mathbf{y}^{(t)} \cdot (\bar{1}-\mathbf{o}))$, which we do in the next two lemmata.

Lemma J.2. *For every integer $i \in [0, \varepsilon^{-1}]$, $\mathbf{y}^{(\varepsilon i)} \geq \bar{0}$ and $\|\mathbf{y}^{(\varepsilon i)}\|_\infty \leq 1-(1-\varepsilon)^{-i}$.*

Proof. We prove the lemma by induction on i. For $i = 0$, the lemma follows directly from the initialization $\mathbf{y}^{(0)} = \bar{0}$ because $1-(1-\varepsilon)^{-0} = 0$. Assume now that the lemma holds for $i-1$, and let us prove it for an integer $0 < i \leq 1$. Observe that, for every $j \in [n]$,

$$y_j^{\varepsilon i} = y_j^{\varepsilon(i-1)} + \varepsilon \cdot \left(1 - y_j^{\varepsilon(i-1)}\right) \cdot s_j^{\varepsilon(i-1)} \geq y_j^{\varepsilon(i-1)} \geq 0 \ ,$$

where the first inequality holds since $y_j^{\varepsilon(i-1)} \leq 1$ by the induction hypothesis and the value of $s_j^{(\varepsilon(i-1))}$ is non-negative by definition. Moreover,

$$\begin{aligned} y_j^{\varepsilon i} &= y_j^{\varepsilon(i-1)} + \varepsilon \cdot \left(1 - y_j^{\varepsilon(i-1)}\right) \cdot s_j^{\varepsilon(i-1)} \leq y_j^{\varepsilon(i-1)} + \varepsilon \cdot \left(1 - y_j^{\varepsilon(i-1)}\right) \\ &= \varepsilon + (1-\varepsilon) \cdot y_j^{\varepsilon(i-1)} \leq \varepsilon + (1-\varepsilon) \cdot \left[1-(1-\varepsilon)^{(i-1)}\right] = 1-(1-\varepsilon)^i \ , \end{aligned}$$

where again the first inequality holds since $s^{(\varepsilon(i-1))} \in \mathcal{X}$, which implies $s_j^i \leq 1$; and the second inequality holds by the induction hypothesis. □

Lemma J.3. *For every integer $i \in [0, \varepsilon^{-1}]$, $F(\mathbf{o}+\mathbf{y}^{(\varepsilon i)} \cdot (\bar{1}-\mathbf{o})) \geq \left[(1-(1-m)\big(1-(1-\varepsilon)^i\big)\right] \cdot F(\mathbf{o}) = \left[m+(1-m)(1-\varepsilon)^i\right] \cdot F(\mathbf{o})$.*

Proof. Observe that

$$\begin{aligned} F(\mathbf{o}+\mathbf{y}^{(\varepsilon i)} \cdot (\bar{1}-\mathbf{o})) &\geq \left(1-\|\mathbf{y}^{(\varepsilon i)}\|_\infty\right) \cdot F(\mathbf{o}) + \|\mathbf{y}^{(\varepsilon i)}\|_\infty \cdot F\left(\mathbf{o} + \frac{\mathbf{y}^{(\varepsilon i)} \cdot (\bar{1}-\mathbf{o})}{\|\mathbf{y}^{(\varepsilon i)}\|_\infty}\right) \\ &\geq \left(1-\|\mathbf{y}^{(\varepsilon i)}\|_\infty\right) \cdot F(\mathbf{o}) + m \cdot \|\mathbf{y}^{(\varepsilon t)}\|_\infty \cdot F(\mathbf{o}) \\ &= \left(1-(1-m) \cdot \|\mathbf{y}^{(\varepsilon i)}\|_\infty\right) \cdot F(\mathbf{o}) \ , \end{aligned}$$

where the first inequality holds since the DR-submodularity of F implies that F is concave along positive directions (such as the direction $\mathbf{y}^{(\varepsilon i)} \cdot (\bar{1}-\mathbf{o})/\|\mathbf{y}^{(\varepsilon i)}\|_\infty$), and the second inequality holds since the monotonicity ratio of F is at least m. Plugging Lemma J.2 into the previous inequality completes the proof of the lemma. □

Using the previous lemma, we can now provide a lower bound on the increase in the value of $\mathbf{y}^{(t)}$ as a function of t.

Lemma J.4. *For every integer $0 \leq i < \varepsilon^{-1}$, $F(\mathbf{y}^{(\varepsilon(i+1))}) - F(\mathbf{y}^{(\varepsilon i)}) \geq \varepsilon \cdot [(m+(1-m) \cdot (1-\varepsilon)^i) \cdot F(\mathbf{o}) - F(\mathbf{y}^{(\varepsilon i)})] - \varepsilon^2 LD^2$.*

Proof. By the chain rule,

$$\begin{aligned} F(\mathbf{y}^{\varepsilon(i+1)}) - F(\mathbf{y}^{(\varepsilon i)}) &= F(\mathbf{y}^{(\varepsilon i)} + \varepsilon \cdot \mathbf{s}^{(\varepsilon i)} \odot (\bar{1}-\mathbf{y}^{(\varepsilon i)})) - F(\mathbf{y}^{(\varepsilon i)}) \\ &= \int_0^\varepsilon \nabla F(\mathbf{y}^{(\varepsilon i)} + r \cdot \mathbf{s}^{(\varepsilon i)} \odot (\bar{1}-\mathbf{y}^{(\varepsilon i)})) \cdot (\mathbf{s}^{(\varepsilon i)} \odot (\bar{1}-\mathbf{y}^{(\varepsilon i)}))\, dr \\ &\geq \int_0^\varepsilon \nabla F(\mathbf{y}^{(\varepsilon i)}) \cdot (\mathbf{s}^{(\varepsilon i)} \odot (\bar{1}-\mathbf{y}^{(\varepsilon i)}))\, dr - \varepsilon^2 LD^2 \\ &= \varepsilon \cdot \nabla F(\mathbf{y}^{(\varepsilon i)}) \cdot (\mathbf{s}^{(\varepsilon i)} \odot (1-\mathbf{y}^{(\varepsilon i)})) - \varepsilon^2 LD^2 \ , \end{aligned}$$

where the first inequality holds by the L-smoothness of F. Furthermore,

$$\begin{aligned} \nabla F(\mathbf{y}^{(\varepsilon i)}) \cdot (\mathbf{s}^{(\varepsilon i)} \odot (\bar{1}-\mathbf{y}^{(\varepsilon i)})) &= ((\bar{1}-\mathbf{y}^{(\varepsilon i)}) \odot \nabla F(\mathbf{y}^{(\varepsilon i)})) \cdot \mathbf{s}^{(\varepsilon i)} \\ &\geq ((\bar{1}-\mathbf{y}^{(\varepsilon i)}) \odot \nabla F(\mathbf{y}^{(\varepsilon i)})) \cdot \mathbf{o} \\ &= \nabla F(\mathbf{y}^{(\varepsilon i)})) \cdot ((\bar{1}-\mathbf{y}^{(\varepsilon i)}) \odot \mathbf{o}) \\ &\geq F(\mathbf{o}+\mathbf{y}^{(\varepsilon i)}(\bar{1}-\mathbf{o})) - F(\mathbf{y}^{(\varepsilon i)}) \\ &\geq \left[m+(1-m) \cdot (1-\varepsilon)^i\right] \cdot F(\mathbf{o}) - F(\mathbf{y}^{(\varepsilon i)}) \ , \end{aligned}$$

where the first inequality holds by the definition of $\mathbf{s}^{(\varepsilon i)}$ since $\mathbf{o}$ is a candidate to be this vector, the second inequality follows from the concavity of F along positive directions, and the last inequality holds by Lemma J.3. The lemma now follows by combining the two above inequalities. $\square$

We are now ready to prove Theorem J.1.

Proof of Theorem J.1. Rearranging the guarantee of Lemma J.4, we get

$$F(\mathbf{y}^{\varepsilon(i+1)}) \geq (1-\varepsilon) \cdot F(\mathbf{y}^{(\varepsilon i)}) + \varepsilon[m + (1-m) \cdot (1-\varepsilon)^i] \cdot F(\mathbf{o}) - \varepsilon^2 LD^2 \ .$$

Since this inequality applies for every integer $0 \leq i < \varepsilon^{-1}$, we can use it repeatedly to obtain

$$\begin{aligned}
F(\mathbf{y}^{(1)}) &\geq \varepsilon \cdot \sum_{i=1}^{1/\varepsilon} (1-\varepsilon)^{1/\varepsilon - i} \cdot \left[(m + (1-m) \cdot (1-\varepsilon)^{i-1}) \cdot F(\mathbf{o}) - \varepsilon LD^2\right] + (1-\varepsilon)^{1/\varepsilon} \cdot F(\bar{0}) \\
&\geq m\varepsilon \cdot \sum_{i=1}^{1/\varepsilon} (1-\varepsilon)^{\frac{1}{\varepsilon} - i} \cdot F(\mathbf{o}) + \varepsilon(1-m) \cdot \sum_{i=1}^{1/\varepsilon} [(1-\varepsilon)^{1/\varepsilon - 1} \cdot F(\mathbf{o}) - \varepsilon LD^2] \\
&= m\varepsilon \cdot \frac{1 - (1-\varepsilon)^{1/\varepsilon}}{\varepsilon} \cdot F(\mathbf{o}) + \varepsilon(1-m) \cdot \frac{(1-\varepsilon)^{1/\varepsilon - 1} \cdot F(\mathbf{o}) - \varepsilon LD^2}{\varepsilon} \\
&\geq \left[m(1-e^{-1}) + (1-m) \cdot e^{-1}\right] \cdot F(\mathbf{o}) - \varepsilon LD^2 \ ,
\end{aligned}$$

where the second inequality follows from the non-negativity of F, and the last inequality holds since $(1-\varepsilon)^{1/\varepsilon} \leq e^{-1} \leq (1-\varepsilon)^{1/\varepsilon - 1}$. $\square$

Statement of Authorship

Principal Author

Title of Paper	Resolving the Approximability of Offline and Online Non-monotone DR-Submodular Maximization over General Convex Sets.		
Publication Status	[✓ Published] Accepted for Publication Submitted for Publication		
Publication Details	In *International Conference on Artificial Intelligence and Statistics*, pp. 2542-2564. PMLR, 2023.		
Name of Principal Author (Candidate)	Loay Mualem		
Contribution to the Paper	Contributed to the theory, implementation and writing of the paper.		
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Name and Signature	Loay Mualem [illegible]	Date	24/6/2025

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the dissertation

Name of Co-Author	Moran Feldman		
Contribution to the Paper	Contributed to the theory and writing of the paper.		
Name and Signature	Moran Feldman	Date	30.6.25

Please cut and paste additional co-author panels here as required.

Chapter 4

Resolving the Approximability of Offline and Online Non-monotone DR-Submodular Maximization over General Convex Sets

In this chapter, we address the problem of maximizing non-monotone DR-submodular functions over general convex constraints, in both offline and online settings. We establish tight approximation guarantees and propose efficient algorithms that resolve several open questions regarding the approximability of this fundamental class of problems.

The following paper [MF23] was published at the *International Conference on Artificial Intelligence and Statistics (AISTATS 2023)*.

Resolving the Approximability of Offline and Online Non-monotone DR-Submodular Maximization over General Convex Sets

Loay Mualem
University of Haifa

Moran Feldman
University of Haifa

Abstract

In recent years, maximization of DR-submodular continuous functions became an important research field, with many real-worlds applications in the domains of machine learning, communication systems, operation research and economics. Most of the works in this field study maximization subject to down-closed convex set constraints due to an inapproximability result by Vondrák (2013). However, Dürr et al. (2021) showed that one can bypass this inapproximability by proving approximation ratios that are functions of m, the minimum ℓ_∞-norm of any feasible vector. Given this observation, it is possible to get results for maximizing a DR-submodular function subject to *general* convex set constraints, which has led to multiple works on this problem. The most recent of which is a polynomial time $\frac{1}{4}(1-m)$-approximation offline algorithm due to Du (2022). However, only a sub-exponential time $\frac{1}{3\sqrt{3}}(1-m)$-approximation algorithm is known for the corresponding online problem. In this work, we present a polynomial time online algorithm matching the $\frac{1}{4}(1-m)$-approximation of the state-of-the-art offline algorithm. We also present an inapproximability result showing that our online algorithm and Du's offline algorithm are both optimal in a strong sense. Finally, we study the empirical performance of our algorithm and the algorithm of Du (which was only theoretically studied previously), and show that they consistently outperform previously suggested algorithms on revenue maximization, location summarization and quadratic programming applications.

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206. Copyright 2023 by the author(s).

1 INTRODUCTION

Optimization of continuous DR-submodular functions has gained prominence in recent times. Such optimization is an important tractable subclass of non-convex optimization, and captures problems at the forefront of machine learning and statistics with many real-world applications (see, e.g., (Bian et al., 2019; Hassani et al., 2017a; Mitra et al., 2021; Soma and Yoshida, 2017)). The majority of the existing works on DR-submodular optimization (and submodular optimization in general) have been focused either on monotone objective functions, or optimization subject to a down-closed convex set constraint.[1] However, many real-world problems are naturally captured as optimization of a non-monotone DR-submodular function over a constraint convex set that is not down-closed. For example, consider a streaming service that would like to produce a summary of recommended movies for a user. Often the design of the user interface places strong bounds on the size of the summary displayed to the user, leading to a non-down-closed constraint. Furthermore, the quality of the summary is often captured by a non-monotone objective since putting very similar films in the summary is detrimental to both its value and professional look.

Motivated by the above-mentioned situation, a few recent works started to consider DR-submodular maximization subject to a general (not necessarily down-closed) convex set constraint $\mathcal{K}$. In general, no constant approximation ratio can be guaranteed for this problem in sub-exponential time due to an hardness result by Vondrák (2013). However, Dürr et al. (2021) showed that this inapproximability result can be bypassed when the convex set constraint $\mathcal{K}$ includes points whose ℓ_∞-norm is less than the maximal value of 1. Specifically, Dürr et al. (2021) presented a sub-exponential time offline algorithm guaranteeing $\frac{1}{3\sqrt{3}}(1-m)$-approximation for this problem, where m is the minimal ℓ_∞-norm of any vector in $\mathcal{K}$. Later, Thắng and Srivastav (2021) showed how to obtain a similar result in an online (regret minimization) setting, and

[1]A set $\mathcal{K} \subseteq [0,1]^n$ is down-closed if, for every two vectors $\mathbf{x}, \mathbf{y} \in [0,1]^n$, $\mathbf{x} \in \mathcal{K}$ whenever $\mathbf{y} \in \mathcal{K}$ and $\mathbf{y}$ coordinate-wise dominates $\mathbf{x}$.

an improved sub-exponential offline algorithm obtaining $\frac{1}{4}(1-m)$-approximation was suggested by Du et al. (2022). Very recently, Du (2022) provided the first polynomial time algorithm for this setting, obtaining the same offline $\frac{1}{4}(1-m)$-approximation as Du et al. (2022). Nevertheless, and despite all the progress described above, there are still important open questions left regarding this setting.

- What is the best approximation ratio that can be obtained by a polynomial time offline algorithm? In particular, can such an algorithm guarantee a better than $\frac{1}{4}(1-m)$-approximation, and if not, how much slower must be an algorithm that improves over this approximation ratio?
- Is there a polynomial time *online* algorithm guaranteeing any constant approximation ratio? Can such an algorithm match the optimal approximation ratio obtainable by an offline algorithm?

In this work we answer all the above questions, essentially settling the problem of maximizing DR-submodular functions over general convex sets in both the offline and online settings. We also study the empirical performance of the theoretically optimal offline and online algorithms, showing that both algorithms consistently outperform previously suggested algorithms. Below we describe our results in more detail.

Online setting. As mentioned above, the state-of-the-art online (regret minimization) algorithm of Thắng and Srivastav (2021) achieves $\frac{1}{3\sqrt{3}}(1-m)$-approximation, which it does with sub-exponential running time and roughly $O(\sqrt{T})$-regret, where T is the number of time steps.[2] In this paper, we describe a new online algorithm improving both the approximation ratio and the time complexity. Specifically, our algorithm achieves $\frac{1}{4}(1-m)$-approximation in polynomial time and roughly $O(\sqrt{T})$-regret. The approximation guarantee of our algorithm matches an inapproximability that we prove for the offline setting (see below), and is thus, optimal. We also study the empirical performance of our algorithm, and show that it outperforms the algorithm of Thắng and Srivastav (2021) on two applications of revenue maximization and location summarization.

Offline setting. Recall that the state-of-the-art offline algorithm is a recent polynomial time $\frac{1}{4}(1-m)$-approximation algorithm due to Du (2022). Our first contribution to the offline setting is an inapproximability result showing that this algorithm is optimal in a very strong sense. Specifically, we show that no sub-exponential time algorithm can significantly improve over this approximation ratio, even when m is fixed to any particular value in $[0,1]$. Furthermore, since Du (2022) analyzed only the theoretical performance of his algorithm, it is interesting to study the empirical performance of this algorithm, which we do by considering revenue maximization and quadratic programming applications.

Coding the algorithm of Du (2022) for the empirical study is somewhat non-trivial because Du (2022) presented his algorithm as part of a general mathematical framework for designing algorithms for various submodular optimization problems. Therefore, our empirical study is based on an explicit version of this algorithm that we give in this paper, which is not fully identical to the algorithm of Du (2022). Beside being explicit, our version of the algorithm also has the advantage of being more tuned towards practical performance. For completeness, we include a full analysis of our version of the algorithm of Du (2022). This full analysis is also used as a warm-up towards the analysis of our own online algorithm.

1.1 Related work

Next, we provide a brief summary of the most relevant results on DR-submodular maximization. Recently, this field has become the work-horse of numerous applications in the fields of statistics and machine learning, which has lead to a dramatic increase in the number of studies related to it.

Offline DR-submodular optimization. The problem of maximizing monotone DR-functions subject to a down-closed convex set was considered by Bian et al. (2017a), who showed a variant of the Frank-Wolfe algorithm (based on the greedy method proposed by Calinescu et al. (2011) for set functions) that guarantees $(1-1/e)$-approximation for this problem, which is optimal (Nemhauser and Wolsey, 1978). Later, Hassani et al. (2017a) showed that the algorithm of Bian et al. (2017a) is not robust in stochastic settings (i.e., when only an unbiased estimator of gradients is available), and proved that gradient methods are robust in such setting while still achieving $1/2$-approximation. When the objective DR-submodular function is not necessarily monotone, the problem becomes harder to approximate. Bian et al. (2019) and Niazadeh et al. (2020) independently provided two algorithms with the same approximation guarantee of $1/2$ for maximizing non-monotone DR-submodular functions over a hypercube, which is optimal (Feige et al., 2011) (the algorithm of Niazadeh et al. (2020) applies also to non-DR submodular functions). For general down-closed convex sets, Bian et al. (2018) provided a $1/e$-approximation algorithm based on the greedy method of Feldman et al. (2011) for set functions. Using the concept of monotonicity ratio, Mualem and Feldman (2022) were able to smoothly interpolate between the

[2]By changing parameter values, it is possible to reduce the time complexity of the algorithm of Thắng and Srivastav (2021) to be polynomial. However, this comes at the cost of a regret that is nearly-linear in T and an error term in the approximation ratio that diminishes very slowly (linearly in $\log T$).

last result and the $(1 - 1/e)$-approximation obtainable for monotone objectives.

Online DR-submodular optimization. Online optimization of monotone DR-submodular functions over general convex sets (for monotone objective functions, there is no difference between optimization subject to down-closed or general convex sets) was first considered by Chen et al. (2018), who provided two algorithms. One guaranteeing $(1 - 1/e)$-approximation using roughly $O(\sqrt{T})$-regret, and another algorithm which is robust to stochastic settings but guarantees only $1/2$-approximation up to the same regret. Later, Chen et al. (2019) presented an algorithm that combines $(1 - 1/e)$-approximation with roughly $O(\sqrt{T})$-regret and robustness, and Zhang et al. (2019) showed how one can reduce the number of gradient calculations per time step to one, at the cost of increasing the regret to roughly $O(T^{4/5})$. Such a reduction is important for bandit versions of the same problem. Online optimization of DR-submodular functions that are not necessarily monotone was studied by Thắng and Srivastav (2021), who provided three algorithms for it. One of these algorithms applies to general convex set constraints, and was already discussed above. Another algorithm applies to maximization over the entire hypercube, and achieves $1/2$-approximation with roughly $O(\sqrt{T})$-regret; and the last algorithm applies to online maximization of non-monotone DR-submodular functions over down-closed convex sets, and achieves $1/e$-approximation with roughly $O(T^{3/4})$-regret.

1.2 Paper organization

In Section 2, we provide some definitions and important properties of DR-submodular functions. Section 3 describes our explicit version of the offline algorithm of Du (2022), which also serves as warm up for our novel online algorithm described in Section 4. Our inapproximability result, showing that the above offline and online algorithms are both optimal, is proved in Section 5. Finally, in Section 6, we study the empirical performance and robustness of our online algorithm and our version of the algorithm of Du (2022) by comparing them with previously suggested algorithms on multiple machine learning applications.

2 PRELIMINARIES

DR-submodularity (first defined by Bian et al. (2017b)) is an extension of the submodularity notion from set functions to continuous functions. Formally speaking, given a domain $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_i$, where $\mathcal{X}_i$ is a closed range in $\mathbb{R}$ for every $i \in [n]$, a function $F\colon \mathcal{X} \to \mathbb{R}$ is *DR-submodular* if for every two vectors $\mathbf{a}, \mathbf{b} \in \mathcal{X}$, positive value k and coordinate $i \in [n]$, the inequality $F(\mathbf{a} + k\mathbf{e}_i) - F(\mathbf{a}) \geq F(\mathbf{b}+k\mathbf{e}_i) - F(\mathbf{b})$ holds whenever $\mathbf{a} \leq \mathbf{b}$ and $\mathbf{b}+k\mathbf{e}_i \in \mathcal{X}$ (here and throughout the paper, $\mathbf{e}_i$ denotes the standard i-th basis vector, and comparison between two vectors should be understood to hold coordinate-wise). Note that if function F is continuously differentiable, then the above definition of DR-submodulrity is equivalent to

$$\nabla F(\mathbf{x}) \leq \nabla F(\mathbf{y}) \quad \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{X}, \mathbf{x} \geq \mathbf{y} \ .$$

Furthermore, when F is twice differentiable, it is DR-submodular if and only if its Hessian is non-positive at every vector $\mathbf{x} \in \mathcal{X}$.

In this work, we study the problem of maximizing a non-negative DR-submodular function $F\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ subject to a general convex body $\mathcal{K} \subseteq \mathcal{X}$ (usually polytope) constraint. For simplicity, we assume that $\mathcal{X} = [0, 1]^n$. Note that this assumption is without loss of generality since there is a natural mapping from $\mathcal{X}$ to $[0, 1]^n$. Additionally, as is standard in the field, we assume that F is β-smooth for some parameter $\beta > 0$. Recall that F is β-smooth if it is continuously differentiable, and for every two vectors $\mathbf{x}, \mathbf{y} \in [0, 1]^n$, the function F obeys $\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\|_2 \leq \beta\|\mathbf{x} - \mathbf{y}\|_2$.

In the online (regret minimization) version of the above problem, there are T time steps. In every time step $t \in [T]$, the adversary selects a non-negative β-smooth DR-submodular function F_t, and then the algorithm should select a vector $\mathbf{y}^{(t)} \in \mathcal{K}$ without knowing F_t (the function F_t is revealed to the algorithm only after $\mathbf{y}^{(t)}$ is selected). The objective of the algorithm is to maximize $\sum_{i=1}^{T} F_t(\mathbf{y}^{(t)})$, and its success in doing so is measured compared to the best fixed vector $\mathbf{x} \in \mathcal{K}$. More formally, we say that the algorithm achieves an approximation ratio of $c \geq 0$ with regret $\mathcal{R}(T)$ if

$$\mathbb{E}\left[\sum_{t=1}^{T} F_t(\mathbf{y}^{(t)})\right] \geq c \cdot \max_{\mathbf{x}\in\mathcal{K}} \mathbb{E}\left[\sum_{t=1}^{T} F_t(\mathbf{x})\right] - \mathcal{R}(T) \ .$$

The nature of the access that the algorithm has to F_t varies between different versions of the above problem. Some previous works assume access to the exact gradient of F. However, our algorithm applies also to a stochastic version of the problem in which only access to an unbiased estimator of this gradient is available.

We conclude this section by introducing some additional notation and two known lemmata that are useful in our proofs. Given two vectors $\mathbf{x}, \mathbf{y} \in [0, 1]^n$, we denote by $\mathbf{x} \vee \mathbf{y}$ and $\mathbf{x} \wedge \mathbf{y}$ their coordinate-wise maximum and minimum, respectively. Using this notation, we can now state the first known lemma, which can be traced back to Hassani et al. (2017a) (see Inequality 7.5 in the arXiv version (Hassani et al., 2017b) of Hassani et al. (2017a)), and is also explicitly stated and proved in (Dürr et al., 2021).

Lemma 2.1 (Lemma 1 of Dürr et al. (2021))**.** *For every two vectors $\mathbf{x}, \mathbf{y} \in [0, 1]^n$ and any continuously differentiable DR-submodular function $F\colon [0, 1]^n \to \mathbb{R}$, $\langle \nabla F(x), y - x\rangle \geq F(\mathbf{x} \vee \mathbf{y}) + F(\mathbf{x} \wedge \mathbf{y}) - 2F(\mathbf{x})$.*

The following lemma originates from a lemma proved by Feldman et al. (2011) for set functions. Extensions of this lemma to continuous domains have appeared in (Bian et al., 2017a; Chekuri et al., 2015), but for completeness, we prove our exact version of the lemma in Appendix A.

Lemma 2.2. *For every two vectors $\mathbf{x}, \mathbf{y} \in [0,1]^n$ and any continuously differentiable non-negative DR-submodular function $F\colon [0,1]^n \to \mathbb{R}_{\geq 0}$, $F(\mathbf{x} \vee \mathbf{y}) \geq (1 - \|\mathbf{x}\|_\infty) F(\mathbf{y})$.*

3 OFFILINE MAXIMIZATION

In this section, we present and analyze an explicit variant of the offline algorithm of Du (2022) for maximizing a non-negative DR-submodular function F over a general convex set $\mathcal{K}$. Since the algorithm of Du (2022) is related to Frank-Wolfe, we name our variant `Non-mon. Frank-Wolfe`, and its pseudocode appears as Algorithm 1. Algorithm 1 gets a non-negative integer parameter T and a quality control parameter $\varepsilon \in (0, 1)$.

Algorithm 1: `Non-mon. Frank-Wolfe` (T, ε)

```
1 Let $\mathbf{y}^{(0)} \leftarrow \arg\min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|_\infty$.
2 for $i = 1$ to $T$ do
3     Let $\mathbf{s}^{(i)} \leftarrow \arg\max_{\mathbf{x} \in \mathcal{K}} \langle \nabla F(\mathbf{y}^{(i-1)}), \mathbf{x} \rangle$
4     Let $\mathbf{y}^{(i)} \leftarrow (1 - \varepsilon) \cdot \mathbf{y}^{(i-1)} + \varepsilon \cdot \mathbf{s}^{(i)}$
5 return the vector maximizing $F$ among
    $\{\mathbf{y}^{(0)}, \dots, \mathbf{y}^{(T)}\}$.
```

For completeness, and as a warmup for Section 4, we present a full analysis of Algorithm 1, independent of the analysis presented by Du (2022). The conclusions of our analysis are summarized by the following theorem. We note that, for the purpose of this theorem, it would have sufficed for Algorithm 1 to return $\mathbf{y}^{(T)}$ rather than the best solution among $\mathbf{y}^{(0)}, \dots, \mathbf{y}^{(T)}$. However, returning the best of these solutions results in a better empirical performance at almost no additional cost.

Theorem 3.1. *Let $\mathcal{K} \subseteq [0,1]^n$ be a general convex set, and let $F\colon [0,1]^n \to \mathbb{R}_{\geq 0}$ be a non-negative β-smooth DR-submodular function. Then,* `Non-mon. Frank-Wolfe` *(Algorithm 1) outputs a solution $\mathbf{w} \in \mathcal{K}$ obeying*

$$F(\mathbf{w}) \geq (1 - 2\varepsilon)^{T-1}[(1+\varepsilon)^T - 1](1 - \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2 \beta D^2 T \enspace ,$$

where D is the diameter of $\mathcal{K}$ and $\mathbf{o} \in \arg\max_{\mathbf{x} \in \mathcal{K}} F(\mathbf{x})$. In particular, when T is set to be $\lfloor \ln 2/\varepsilon \rfloor$,

$$F(\mathbf{w}) \geq (1/4 - 3\varepsilon)(1 - \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon\beta D^2 \enspace .$$

We begin the proof of Theorem 3.1 with the following observation, which bounds the rate in which the infinity norm of the solution maintained by Algorithm 1 can be increase. The proof of this observation is done by induction on the number of iterations, and can be found in Appendix B (like all the other proofs of this section).

Observation 3.2. *For every integer $0 \leq i \leq T$, $1 - \|\mathbf{y}^{(i)}\|_\infty \geq (1 - \varepsilon)^i \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty)$.*

By combining the last observation and Lemma 2.2, we can prove the following lemma about the rate in which the value of $F(y^{(i)})$ increases as a function of i. The proof gives a bound on the rate of increase in terms of $\langle \mathbf{s}^{(i)}, \nabla F(\mathbf{y}^{(i-1)}) \rangle$, and then lower bounds this inner product by observing that $\mathbf{o}$ is one possible candidate to be $\mathbf{s}^{(i)}$.

Lemma 3.3. *For every integer $1 \leq i \leq T$, $F(\mathbf{y}^{(i)}) \geq (1 - 2\varepsilon) \cdot F(\mathbf{y}^{(i-1)}) + \varepsilon(1 - \varepsilon)^{i-1} \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2 \beta D^2$.*

Theorem 3.1 is proved by using Lemma 3.3 repeatedly.

4 ONLINE MAXIMIZATION

In this section, we consider the problem of maximizing a non-negative DR-submodular function F over a general convex set $\mathcal{K}$ in the online setting. The only currently known algorithm for this problem is an algorithm due to Thắng and Srivastav (2021) which guarantees $\frac{1 - \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|_\infty}{3\sqrt{3}}$-approximation. One drawback of this algorithm is that its regret is roughly T over the logarithm of the running time, and therefore, to make this regret less than nearly-linear in T one has to allow for a super-polynomial time complexity (furthermore, a sub-exponential time complexity is necessary to get a regret of T^c for any constant $c \in (0, 1)$). Our algorithm, given as Algorithm 2, combines ideas from our offline algorithm and the Meta-Frank-Wolfe algorithm suggested in (Chen et al., 2018), and guarantees both $\frac{1}{4}(1 - \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x}\|_\infty)$-approximation and roughly $O(\sqrt{T})$-regret in polynomial time.

Like the original Meta-Frank-Wolfe algorithm of Chen et al. (2018), our algorithm uses in a black-box manner multiple instances $\mathcal{E}$ of an online algorithm for linear optimization. More formally, we assume that every instance $\mathcal{E}$ has the following behavior and guarantee. There are T time steps. In every time step $t \in [T]$, $\mathcal{E}$ selects a vector $\mathbf{u}^{(t)} \in \mathcal{K}$, and then an adversary reveals to $\mathcal{E}$ a vector $\mathbf{d}^{(t)}$ that was chosen independently of $\mathbf{u}^{(t)}$. The algorithm $\mathcal{E}$ guarantees that

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle \mathbf{u}^{(t)}, \mathbf{d}^{(t)} \rangle\right] \geq \max_{\mathbf{x} \in \mathcal{K}} \mathbb{E}\left[\sum_{t=1}^{T} \langle \mathbf{x}, \mathbf{d}^{(t)} \rangle\right] - \mathcal{R}(T)$$

for some regret function $\mathcal{R}(T)$ that depends on the particular linear optimization algorithm chosen as the black-box (and may depend on the convex body $\mathcal{K}$ and the bounds available on the adversarially chosen vectors $\mathbf{d}^{(t)}$). One possible choice for an online linear optimization algorithm is Regularized-Follow-the-Leader due to Abernethy et al. (2008) that has $\mathcal{R}(T) \leq DG\sqrt{2T}$, where D is the diameter of $\mathcal{K}$ and $G = \max_{1\leq t\leq T} \|\mathbf{d}^{(t)}\|_2$.

Algorithm 2 runs in each time step a procedure similar to our version of the offline algorithm (`Non-mon. Frank-Wolfe`). However, instead of calculating a point $\mathbf{s}$ that is good with respect to the gradient at the current solution, Algorithm 2 asks an instance of an online linear optimization algorithm to provide such a point. At the end of the time step, the online linear optimization algorithm gets an estimate of the gradient as the adversarial vector, and therefore, on average, the points it produces are a good approximation of the optimal point in retrospect. Algorithm 2 gets three parameters. The parameters L and ε correspond to the parameters T and ε of `Non-mon. Frank-Wolfe` (Algorithm 1),[3] respectively, and the parameter T is the number of time steps.

Algorithm 2: `Non-mon. Meta-Frank-Wolfe` (L, ε, T)

```
1  for $i = 1$ to $L$ do Initialize an instance $\mathcal{E}_i$ of some online algorithm for linear optimization.
2  for $t = 1$ to $T$ do
3      Let $\mathbf{y}^{(0,t)} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{K}} \|\mathbf{x}\|_\infty$.
4      for $i = 1$ to $L$ do
5          Let $\mathbf{s}^{(i,t)} \in \mathcal{K} \leftarrow$ be the vector picked by $\mathcal{E}_\ell$ in time step $t$.
6          Let $\mathbf{y}^{(i,t)} \leftarrow (1-\varepsilon)\cdot \mathbf{y}^{(i-1,t)} + \varepsilon \cdot \mathbf{s}^{(i,t)}$.
7      Play $\mathbf{y}^{(t)} = \mathbf{y}^{(L,t)}$.
8      for $i = 1$ to $L$ do
9          Observe an unbiased estimator $\mathbf{g}^{(i,t)}$ of $\nabla F_t(\mathbf{y}^{(i-1,t)})$.
10         Pass $\mathbf{g}^{(i,t)}$ as the adverserially chosen vector $\mathbf{d}^{(t)}$ for $\mathcal{E}_i$.
```

The main result that we prove regarding the online setting is given by the next theorem.

Theorem 4.1. *Let $\mathcal{K}$ be a general convex set with diameter D. Assume that for every $1 \leq t \leq T$, $F_t\colon [0,1]^n \to \mathbb{R}_{\geq 0}$ is a β-smooth DR-submodular function, then*

$$\sum_{t=1}^{T} \mathbb{E}[F_t(\mathbf{y}^{(t)})] \geq (1-2\varepsilon)^{T-1}[(1+\varepsilon)^T - 1](1 - \min_{\mathbf{x}\in\mathcal{K}}\|\mathbf{x}\|_\infty)\cdot \mathbb{E}\left[\sum_{t=1}^{T} F(\mathbf{o})\right] - \varepsilon L \cdot \mathcal{R}(T) - 0.5\varepsilon^2\beta D^2 TL ,$$

where D is the diameter of $\mathcal{K}$, $\mathbf{o}$ is a vector in $\mathcal{K}$ maximizing $\mathbb{E}[\sum_{t=1}^T F_t(\mathbf{o})]$, and $\mathcal{R}(T)$ is the regret of the online linear optimization algorithm over the domain $\mathcal{K}$ when the adversarial vectors $\mathbf{d}^{(t)}$ are the estimators $\mathbf{g}^{(i,t)}$ calculated by Algorithm 2. In particular, when L is set to be $\lfloor \ln 2/\varepsilon \rfloor$, ε is set to be $1/\sqrt{T}$ and $\mathcal{E}_i$ is chosen as an instance of Regularized-Follow-the-Leader,

$$\sum_{t=1}^{T} \mathbb{E}[F_t(\mathbf{y}^{(t)})] \geq (1/4 - 3\varepsilon)(1 - \min_{\mathbf{x}\in\mathcal{K}}\|\mathbf{x}\|_\infty)\cdot \mathbb{E}\left[\sum_{t=1}^{T} F_t(\mathbf{o})\right] - (G + \beta D)D\sqrt{T} ,$$

where $G = \max_{1\leq i\leq L, 1\leq t\leq T} \|\mathbf{g}^{(i,t)}\|_2$.

Remark: In the last theorem we have set ε to $1/\sqrt{T}$, which requires pre-knowledge of T. This can be avoided by using a dynamic value for ε that changes as a function of the number of time slots that have already passed.

We begin the proof of Theorem 4.1 by observing that a repetition of the first half of the proof of Lemma 3.3 leads to the following lemma.

Lemma 4.2. *For every two integers $1 \leq t \leq T$ and $1 \leq i \leq L$, $F_t(\mathbf{y}^{(i,t)}) \geq F_t(\mathbf{y}^{(i-1,t)}) + \varepsilon \cdot \langle \mathbf{s}^{(i,t)} - \mathbf{y}^{(i-1,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)})\rangle - 0.5\varepsilon^2\beta D^2$.*

Using the guarantee of $\mathcal{E}_i$, it is possible to get the following lemma from the previous one.

Lemma 4.3. *For every integer number $1 \leq i \leq L$, $\mathbb{E}[\sum_{t=1}^T F_t(\mathbf{y}^{(i,t)})] \geq \mathbb{E}[\sum_{t=1}^T F_t(\mathbf{y}^{(i-1,t)}) + \varepsilon\cdot\sum_{t=1}^T\langle \mathbf{o} - \mathbf{y}^{(i-1,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)})\rangle] - \varepsilon\cdot\mathcal{R}(T) - 0.5\varepsilon^2\beta D^2 T$.*

Proof. Summing up Lemma 4.2 over all t values, we get

$$\begin{aligned}\sum_{t=1}^{T} F_t(\mathbf{y}^{(i,t)}) &\geq \sum_{t=1}^{T} F_t(\mathbf{y}^{(i-1,t)}) - 0.5\varepsilon^2\beta D^2 T \\ &\quad + \varepsilon\cdot\sum_{t=1}^{T}\langle \mathbf{s}^{(i,t)} - \mathbf{y}^{(i-1,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)})\rangle \\ &= \sum_{t=1}^{T} F_t(\mathbf{y}^{(i-1,t)}) - 0.5\varepsilon^2\beta D^2 T + \varepsilon\cdot\left[\sum_{t=1}^{T}\langle \mathbf{s}^{(i,t)}, \mathbf{g}^{(i,t)}\rangle\right. \\ &\quad + \sum_{t=1}^{T}\langle \mathbf{s}^{(i,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)}) - \mathbf{g}^{(i,t)}\rangle \\ &\quad \left. - \sum_{t=1}^{T}\langle \mathbf{y}^{(i-1,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)})\rangle\right] .\end{aligned}$$

Additionally, since $\mathbf{g}^{(i,t)}$ is independent of $\mathbf{s}^{(i,t)}$, by the

[3]The parameter T of `Non-mon. Frank-Wolfe` was renamed to L here to accommodate the standard notation in both offline and online algorithms. In offline Frank-Wolfe-like algorithms, the number of iterations is usually denoted by T, and in online algorithms T is reserved to the number of time steps.

guarantee of $\mathcal{E}_i$,

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle \mathbf{s}^{(i,t)}, \mathbf{g}^{(i,t)}\rangle\right] \geq \mathbb{E}\left[\sum_{t=1}^{T}\langle \mathbf{o}, \mathbf{g}^{(i,t)}\rangle\right] - \mathcal{R}(T) \enspace .$$

Finally, since $\mathbf{g}^{(i,t)}$ is chosen after $\mathbf{y}^{(i-1,t)}$,

$$\begin{aligned}&\mathbb{E}[\langle \mathbf{s}^{(i,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)}) - \mathbf{g}^{(i,t)}\rangle \mid \mathbf{s}^{(i,t)}, \mathbf{y}^{(i-1,t)}]\\ &= \langle \mathbf{s}^{(i,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)}) - \mathbb{E}[\mathbf{g}^{(i,t)} \mid \mathbf{y}^{(i-1,t)}]\rangle\\ &= \langle \mathbf{s}^{(i,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)}) - \nabla F_t(\mathbf{y}^{(i-1,t)})\rangle = 0 \enspace ,\end{aligned}$$

which by the law of total expectation implies the equality $\mathbb{E}[\langle \mathbf{s}^{(i,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)}) - \mathbf{g}^{(i,t)}\rangle] = 0$. Combining all the above inequalities yields

$$\begin{aligned}&\mathbb{E}\left[\sum_{t=1}^{T} F_t(\mathbf{y}^{(i,t)})\right]\\ &\geq \mathbb{E}\left[\sum_{t=1}^{T} F_t(\mathbf{y}^{(i-1,t)})\right] + \varepsilon \cdot \left\{\sum_{t=1}^{T}\langle o, \mathbb{E}[\mathbf{g}^{(i,t)}]\rangle - \mathcal{R}(T)\right.\\ &\qquad \left. - \mathbb{E}\left[\sum_{t=1}^{T}\langle \mathbf{y}^{(i-1,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)})\rangle\right]\right\} - 0.5\varepsilon^2\beta D^2 T\\ &= \mathbb{E}\left[\varepsilon \cdot \sum_{t=1}^{T}\langle o - \mathbf{y}^{(i-1,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)})\right.\\ &\qquad \left. + \sum_{t=1}^{T} F_t(\mathbf{y}^{(i-1,t)})\rangle\right] - \varepsilon \cdot \mathcal{R}(T) - 0.5\varepsilon^2\beta D^2 T \enspace . \quad \square\end{aligned}$$

Corollary 4.4. *For every integer number $1 \leq i \leq L$, $\mathbb{E}[\sum_{t=1}^{T} F_t(\mathbf{y}^{(i,t)})] \geq \mathbb{E}[(1-2\varepsilon) \cdot \sum_{t=1}^{T} F_t(\mathbf{y}^{(i-1,t)}) + \varepsilon(1-\varepsilon)^{i-1} \cdot \sum_{t=1}^{T}(1 - \|\mathbf{y}^{(0,t)}\|_\infty) \cdot F_t(\mathbf{o})] - \varepsilon \cdot \mathcal{R}(T) - 0.5\varepsilon^2\beta D^2 T$.*

Proof. To see why this corollary follows from Lemma 4.3, it suffices to observe that, for every integer $1 \leq t \leq T$,

$$\begin{aligned}&\langle \mathbf{o} - \mathbf{y}^{(i-1,t)}, \nabla F_t(\mathbf{y}^{(i-1,t)})\rangle\\ &\geq F_t(\mathbf{o} \vee \mathbf{y}^{(i-1,t)}) + F_t(\mathbf{o} \wedge \mathbf{y}^{(i-1,t)}) - 2F_t(\mathbf{y}^{(i-1,t)})\\ &\geq F_t(\mathbf{o} \vee \mathbf{y}^{(i-1,t)}) - 2F_t(\mathbf{y}^{(i-1)})\\ &\geq (1-\varepsilon)^{i-1} \cdot (1 - \|\mathbf{y}^{(0,t)}\|_\infty) \cdot F_t(\mathbf{o}) - 2F_t(\mathbf{y}^{(i-1,t)}) \enspace ,\end{aligned}$$

where the first inequality follows from Lemma 2.1, the second inequality holds by the non-negativity of F_t, and the last inequality follows from Lemma 2.2 and the observation that the proof of Observation 3.2 extends to Algorithm 2 and yields $1 - \|y^{(i,t)}\|_\infty \leq (1-\varepsilon)^i \cdot (1 - \|\mathbf{y}^{(0,t)}\|_\infty)$. $\square$

One can observe that Corollary 4.4 is very similar to Lemma 3.3 (the main difference between the two is that in Corollary 4.4 the sum $\sum_{t=1}^{T} F_t$ replaces the function F from Lemma 3.3). This similarity means that the proof of Theorem 3.1 can work with Corollary 4.4 instead of Lemma 3.3, which yields Theorem 4.1.

5 INAPPROXIMABILITY

This section includes our inapproximability result, which is given by the following theorem. Our result shows that the known offline result (reproved in Section 3) for maximizing a DR-submodular function subject to a general convex set is optimal. Notice that this implies that our online algorithm from Section 4 is also optimal (at least in terms of the approximation ratio) unless one allows for an exponential time complexity.

Theorem 5.1. *For every two constants $h \in [0,1)$ and $\varepsilon > 0$, no sub-exponential time algorithm can obtain $(1/4(1-h)+\varepsilon)$-approximation for the problem of maximizing a continuously differentiable non-negative DR-submodular function $F\colon [0,1]^n \to \mathbb{R}_{\geq 0}$ subject to a solvable polytope $\mathcal{K}$ obeying $\min_{\mathbf{x}\in\mathcal{K}} \|\mathbf{x}\|_\infty = h$. Furthermore, this is true even if we are guaranteed that $\max_{\mathbf{x}\in\mathcal{K}} F(\mathbf{x}) = \Omega(n^{-1})$ and F is β-smooth for some β that is polynomial in n.*

The last part of Theorem 5.1 specifies some additional conditions under which the inapproximability stated in the theorem still applies. These conditions are important because under them our algorithm from Section 3 can be made to have a clean approximation guarantee of $1/4(1 - \min_{\mathbf{x}\in\mathcal{K}} \|\mathbf{x}\|_\infty) - \varepsilon'$, for any constant $\varepsilon' > 0$, by choosing a polynomially small value for the parameter ε of the algorithm (to see that this is indeed the case, it is important to observe that since $\mathcal{K} \subseteq [0,1]^n$, the diameter D of $\mathcal{K}$ is at most $\sqrt{n}$).

Theorem 5.1 is unconditional, i.e., it does not rely on any complexity assumption. Instead, Theorem 5.1 assumes a constraint on the way in which the algorithm may access the objective F. It is standard in the field to assume that the algorithm can access F only by querying the value or gradient of F at a given point $\mathbf{x}$. Theorem 5.1 applies under this standard assumption, and furthermore, it applies even when the algorithm is allowed any query about F whose output is determined by the values of F in an arbitrarily small neighborhood of a point $\mathbf{x}$. Note that the standard queries of value and gradient at $\mathbf{x}$ both fall within this class of queries, and the same is true for other natural kind of queries (such as higher order derivatives of F).

The proof of Theorem 5.1 is based on the symmetry gap framework of Vondrák (2013). To use this framework, we first need to choose a submodular set function f_k ($k \geq 1$ is an integer parameter of the function). We choose the same function that was used by Vondrák (2013) to prove his hardness for maximizing a submodular function subject to a matroid base constraint. Specifically, the ground set of f_k is the set $\mathcal{N}_k = \{a_i, b_i \mid i \in [k]\}$, and for every set $S \subseteq \mathcal{N}_k$,

$$f_k(S) = \sum_{i=1}^{k} \mathbf{1}[a_i \in S] \cdot \mathbf{1}[b_i \notin S] \enspace .$$

One can verify that f_k is non-negative and submodular since it is the cut function of a directed graph consisting of k vertex-disjoint arcs.

We now would like to convert f_k into two DR-submodular functions, which we do using the following lemma of Vondrák (2013). This lemma refers to the multilinear extension of a set function $f\colon 2^{\mathcal{N}} \to \mathbb{R}$ over a ground set $\mathcal{N}$. This extension is a function $F\colon [0,1]^{\mathcal{N}} \to \mathbb{R}$ defined for every vector $\mathbf{x} \in [0,1]^{\mathcal{N}}$ by $F(\mathbf{x}) = \mathbb{E}[f(\mathtt{R}(\mathbf{x}))]$, where $\mathtt{R}(\mathbf{x})$ is a random subset of $\mathcal{N}$ that includes every element $u \in \mathcal{N}$ with probability x_u, independently.

Lemma 5.2 (Lemma 3.2 of Vondrák (2013))**.** *Consider a function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ invariant under a group of permutations $\mathcal{G}$ on the ground set $\mathcal{N}$. Let $F(\mathbf{x})$ be the multilinear extension of f, define $\bar{x} = \mathbb{E}_{\sigma\in\mathcal{G}}[\mathbf{1}_{\sigma(\mathbf{x})}]$ and fix any $\varepsilon' > 0$. Then, there is $\delta > 0$ and functions $\hat{F}, \hat{G}\colon [0,1]^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ (which are also symmetric with respect to $\mathcal{G}$), satisfying the following:*

1. *For all $\mathbf{x} \in [0,1]^{\mathcal{N}}$, $\hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$.*
2. *For all $\mathbf{x} \in [0,1]^{\mathcal{N}}$, $|\hat{F}(\mathbf{x}) - F(\mathbf{x})| \leq \varepsilon'$.*
3. *Whenever $\|\mathbf{x} - \bar{\mathbf{x}}\|_2 \leq \delta$, $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x})$ and the value depends only on $\bar{\mathbf{x}}$.*
4. *The first partial derivatives of $\hat{F}$ and $\hat{G}$ are absolutely continuous.*
5. *If f is monotone, then, for every element $u \in \mathcal{N}$, $\frac{\partial \hat{F}}{\partial x_u} \geq 0$ and $\frac{\partial \hat{G}}{\partial x_u} \geq 0$ everywhere.*
6. *If f is submodular then, for every two elements $u, v \in \mathcal{N}$, $\frac{\partial^2 \hat{F}}{\partial x_u \partial x_v} \leq 0$ and $\frac{\partial^2 \hat{G}}{\partial x_u \partial x_v} \leq 0$ almost everywhere.*

Observe that f_k is invariant to exchanging the identities of a_i and b_i with a_j and b_j, respectively, for any choice of $i, j \in [k]$. Therefore, we can choose $\mathcal{G}$ in the last lemma as the group of permutations that can be obtained by any number of such exchanges. In the rest of this section, we assume that $\hat{F}_k$ and $\hat{G}_k$ are functions $\hat{F}$ and $\hat{G}$ obtained using Lemma 5.2 for this choice of $\mathcal{G}$, f_k and $\varepsilon' = 1/(2k)$. It is also important to note that for this choice of $\mathcal{G}$ we have for every vector $\mathbf{x} \in [0,1]^{\mathcal{N}_k}$ and $i \in [k]$

$$\bar{x}_{a_i} = \frac{1}{k}\sum_{j=1}^{k} x_{a_j} \qquad \text{and} \qquad \bar{x}_{b_i} = \frac{1}{k}\sum_{j=1}^{k} x_{b_j} \enspace .$$

Let us now define a family of polytopes. The polytope $\mathcal{P}_{h,k}$ is the convex hull of the $k+1$ vectors $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \ldots, \mathbf{v}^{(k)}$ and $\mathbf{u}$ defined as follows. For every $j \in [k]$, $u_{a_j} = 0$ and $u_{b_j} = h$. For every $i, j \in [k]$,

$$v^{(i)}_{a_j} = \begin{cases} 1 & \text{if } i = j \enspace , \\ 0 & \text{otherwise} \enspace , \end{cases} \qquad \text{and} \qquad v^{(i)}_{b_j} = \begin{cases} 1 & \text{if } i \neq j \enspace , \\ 0 & \text{otherwise} \enspace . \end{cases}$$

Using the above definitions, we can state two instances of the problem we consider

$$\begin{array}{l} \max \hat{F}_k(\mathbf{x}) \\ \mathbf{x} \in \mathcal{P}_{h,k} \end{array} \qquad \text{and} \qquad \begin{array}{l} \max \hat{G}_k(\mathbf{x}) \\ \mathbf{x} \in \mathcal{P}_{h,k} \end{array} \enspace .$$

In Appendix C we refer to these instances as the *basic* instances. We show there that by "scrambling" these instances in an appropriate way, they can be made indistinguishable. This yields Theorem 5.1 as we also prove in Appendix C that the scrambled instances obey the properties assumed in the theorem, and furthermore, that there is a large gap between the optimal values of scrambled instances derived from the two basic instances.

6 APPLICATIONS AND EXPERIMENTAL RESULTS

Up until recently, all the algorithms suggested for submodular maximization subject to general convex set constraints had a sub-exponential execution time. As mentioned above, Du (2022) has recently shown the first polynomial time offline algorithm for this problem, and in this paper we have shown another polynomial time algorithm obtaining a similar guarantee for the online (regret minimization) setting. In this section (and Appendix D), we study the empirical performance of these algorithms on the machine learning applications of revenue maximization, location summarization and quadratic programming. We note that these are just a few examples of standard applications to which our results can be applied (other possible applications include, for example, movie recommendation and image summarization).

In the case of the offline algorithm, it is important to note that (i) we analyze our explicit version of the algorithm, rather than the original version of Du (2022); and (ii) it is interesting to study the empirical performance of the algorithm of Du (2022) because only a theoretical analysis of this algorithm appeared in (Du, 2022).

Since the previously suggested algorithms require sub-exponential execution time, and thus cannot be used as is, we allowed all algorithms in our experiments the same number of iterations. This makes all the algorithms terminate in roughly the same amount of time, and allows for a fair comparison between the quality of their solutions. In a nutshell, our experiments show that our online algorithm and the offline algorithm of Du (2022) provide better solutions (often much better) compared to their state-of-the-art sub-exponential time counterparts.

6.1 Revenue Maximization

Following Thắng and Srivastav (2021), our first set of experiments considers revenue maximization in the following setting. The goal of a company is to advertise a product to users so that the revenue increases through the "word-of-mouth" effect. Formally, the input for the problem is a weighted undirected graph $G = (V, E)$ representing a social network graph, where w_{ij} denotes the weight of the edge between vertex i and vertex j ($w_{ij} = 0$ if the edge (i, j) is missing from the graph). If the company invests x_i unit of cost in a user $i \in V$, then this user becomes an advocate of the product with probability $1 - (1 - p)^{x_i}$, where $p \in (0, 1)$ is a parameter. Note that this means that each ε unit of cost invested in the user has an independent chance to make the user an advocate, and that by investing a full unit in the user, she becomes an advocate with probability p (Soma and Yoshida, 2017).

Let $S \subseteq V$ be a set of users who ended up being advocates for the product. Then, the revenue obtained is represented by the total influence of the users of S on non-advociate users, or more formally, by $\sum_{i \in S} \sum_{j \in V \setminus S} w_{ij}$. The objective function $f\colon [0,1]^V \to \mathbb{R}_{\geq 0}$ of the experiments is accordingly defined as the expectation of the above expression, i.e.,

$$\begin{aligned} f(\mathbf{x}) &= \mathbb{E}_S\left[\sum_{i \in S} \sum_{j \in V \setminus S} w_{ij}\right] \\ &= \sum_{i \in V} \sum_{\substack{j \in V \\ i \neq j}} w_{ij}(1 - (1-p)^{x_i})(1-p)^{x_j} \quad . \end{aligned}$$

It has been shown that f is a non-monotone DR-submodular function (Soma and Yoshida, 2017).

In both the online and offline settings, we experimented on instances of the above setting based on two different datasets. The first is a Facebook network (Viswanath et al., 2009), and includes $64K$ users (vertices) and $1M$ unweighted relationships (edges). The second dataset is based on the Advogato network (Massa et al., 2009), and includes $6.5K$ users (vertices) as well as $61K$ weighted relationships (edges).

6.1.1 Online setting

When performing our experiments in the online settings, we tried to closely mimic the experiment of Thắng and Srivastav (2021). Therefore, we chose the number of time steps to be $T = 1000$, and the parameter $p = 0.0001$. In each time step t, the objective function is defined in the following way. A subset $V^t \subseteq V$ is selected, and only edges connecting two vertices of V^t are kept. In the case of the Advogato network, V_t is a uniformly random subset of V of size 200, and in the case of the much larger Facebook network, V_t is a uniformly random subset of V of size 15,000. The optimization is done subject to the constraint $0.1 \leq \sum_i x_i \leq 1$, which represents both minimum and maximum investment requirements. Note that the intersection of this constraint with the implicit box constraint represents a non-down-monotone feasibility polytope.

In our experiments, we have compared our algorithm from Section 4 with the algorithm of Thắng and Srivastav (2021), which is the only other algorithm for the online setting currently known. In both algorithms, we have set the number of online linear optimizers used to be $L = 100$, and in our algorithm we have set the error parameter $\varepsilon = 0.03$ (there is no error parameter in the algorithm of Thắng and Srivastav (2021)). The results of these experiments on the Advogato and Facebook networks can be found in Figures 1a and 1b, respectively. One can observe that our algorithm significantly outperforms the state-of-the-art algorithm for any number of time steps.

6.1.2 Offline setting

Our experiments in the offline setting are similar to the ones done in the online setting, with two differences. First, since there is only one objective function in the offline setting, we base it on the entire network graph rather than on a subset of its vertices. Second, for the sake of diversity, we changed the constraint to be $0.25 \leq \sum_i x_i \leq 1$ (but we note that the results of the experiments remain essentially unchanged if one reuse the constraint from the online setting).

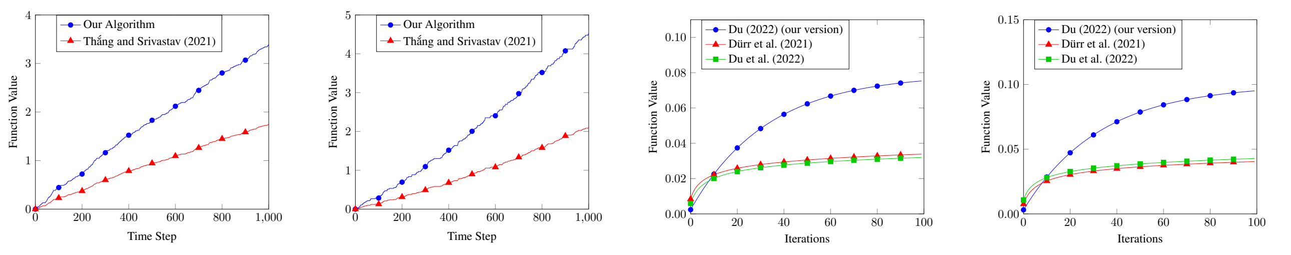
In our experiments, we have compared our explicit version from Section 3 of the algorithm of Du (2022) with the previous algorithms of Dürr et al. (2021) and Du et al. (2022). All the algorithms have been executed for $T = 100$ iterations,[4] and the error parameter ε was set 0.03 in (our version of) the algorithm of Du (2022). The results of these experiments on the Advogato and Facebook networks can be found in Figures 1c and 1d, respectively. One can observe that our version of the polynomial time algorithm of Du (2022) clearly outperforms the two previous algorithms, except when the number of iterations is very low.

6.2 Location Summarization

In this section we consider a location summarization task based on the Yelp dataset (Yelp), which is a subset of Yelp's businesses, reviews and user data. This dataset contains information about local businesses across 11 metropolitan areas, and we have followed the technique of Kazemi et al. (2021) for generating symmetry scores between these locations based on features extracted from the descriptions of the locations and their related user reviews (such as parking options, WiFi access, having vegan menus).

We would like to pick a non-empty set of up to 2 locations that summarizes the existing locations, while not being too far from the current location of the user. A natural objective function for this task (which is very similar to the objective function used in (Kazemi et al., 2021)) is the following set function. Assume that the set of locations is $[n]$, $M_{i,j}$ is

[4]Recall that the number of iterations corresponds to the parameter L in the online setting, which was also set to 100 above.



(a) Online Algorithms on the Advogato network.

(b) Online Algorithms on the Facebook network.

(c) Offline Algorithms on the Advogato network.

(d) Offline Algorithms on the Facebook network.

Figure 1: Results of the Revenue Maximization Experiments

the similarity score between locations i and j, and d_i is the distance of location i from the user (in units of 200KM); then for every set $S \subseteq [n]$, the value of the objective is $f(S) = \frac{1}{n}\sum_{i=1}^{n} \max_{j\in S} M_{i,j} - \sum_{i\in S} d_i$.

Since f is a set function, and the tools we have developed in this work apply only to continuous functions, we optimize the multilinear extension F of f,[5] which is given for every vector $\mathbf{x} \in [0,1]^n$ by

$$F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[x_j M_{i,j} \cdot \prod_{j' | M_{i,j} \prec M_{i,j'}} (1 - x_{j'})\right] - \sum_{i=1}^{n} x_i d_i \enspace .$$

The multilinear extension F is DR-submodular since f is submodular. Moreover, any solution obtained while optimizing F can be rounded into a solution obtaining the same approximation guarantee for f using either pipage or swap rounding (Calinescu et al., 2011; Chekuri et al., 2010).

In our experiment, we restricted attention to a single metropolitan area (Charlotte), and assumed there are 100 time steps. In each time step, a new user u arrives, and her location is determined uniformly at random within the rectangle containing the metropolitan area. Let us denote by F_u the function F when the distances are calculated based on the location of u. When user u arrives, we would like to choose a vector $\mathbf{x}^{(u)}$ maximizing F_u among all vectors obeying $\|\mathbf{x}\|_1 \in [1, 2]$ (recall that we look for solutions that include 1 or 2 locations). Furthermore, we would like to do that before learning the location of u (to speed up the response and for privacy reasons); thus, we need to consider online optimization algorithms. Specifically, like in Section 6.1.1, we compared our algorithm from Section 4 with the algorithm of Thắng and Srivastav (2021). In both algorithms, we have set the number of online linear optimizers used to be $L = 100$, and in our algorithm we have set the error parameter $\varepsilon = 0.03$. The results of the experiment can be found in Figure 2, and they show that our algorithm (again) significantly outperforms the state-of-the-art algorithm for any number of time steps.

[5]See Section 5 for a definition of the multi-linear extension.

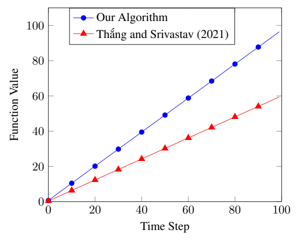


Figure 2: Location Summarization Experiment

7 CONCLUSION

In this work, we have considered the problem of maximizing a DR-submodular function over a general convex set in both the offline and the online (regret minimization) settings. For the online setting we provided the first polynomial time algorithm. Our algorithm matches the approximation guarantee of the only polynomial time algorithm known for the offline setting. Moreover, we presented a hardness result showing that this approximation guarantee is optimal for both settings. Finally, we have run experiments to study the empirical performance of both our algorithm and the (recently suggested) polynomial time offline algorithm. Our experiments show that both these algorithms outperform previous benchmarks.

Acknowledgements

This work was supported in part by Israel Science Foundation (ISF) grant number 459/20.

References

Jacob Abernethy, Elad Hazan, and Alexander Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Conference on Learning Theory (COLT)*, pages 263–273, 2008. URL `https://www.learningtheory.org/colt2008/papers/123-Abernethy.pdf`.

An Bian, Kfir Yehuda Levy, Andreas Krause, and Joachim M. Buhmann. Non-monotone continuous DR-submodular maximization: Structure and algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 486–496, 2017a. URL `https://proceedings.neurips.cc/paper/2017/hash/58238e9ae2dd305d79c2ebc8c1883422-Abstract.html`.

An Bian, Kfir Y Levy, Andreas Krause, and Joachim M Buhmann. Non-monotone continuous DR-submodular maximization: Structure and algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 487–497. Curran, 2018.

Andrew An Bian, Joachim M. Buhmann, Andreas Krause, and Sebastian Tschiatschek. Guarantees for greedy maximization of non-submodular functions with applications. In *International Conference on Machine Learning (ICML)*, pages 498–507, 2017b. URL `http://proceedings.mlr.press/v70/bian17a.html`.

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

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

Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *Foundations of Computer Science (FOCS)*, pages 575–584. IEEE Computer Society, 2010. doi: 10.1109/FOCS.2010.60. URL `https://doi.org/10.1109/FOCS.2010.60`.

Chandra Chekuri, T. S. Jayram, and Jan Vondrák. On multiplicative weight updates for concave and submodular function maximization. In Tim Roughgarden, editor, *Innovation in Theoretical Computer Science (ITCS)*, pages 201–210. ACM, 2015. doi: 10.1145/2688073.2688086. URL `https://doi.org/10.1145/2688073.2688086`.

Lin Chen, Hamed Hassani, and Amin Karbasi. Online continuous submodular maximization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1896–1905. PMLR, 2018.

Lin Chen, Mingrui Zhang, and Amin Karbasi. Projection-free bandit convex optimization. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89 of *Proceedings of Machine Learning Research*, pages 2047–2056. PMLR, 2019. URL `http://proceedings.mlr.press/v89/chen19f.html`.

Donglei Du. Lyapunov function approach for approximation algorithm design and analysis: with applications in submodular maximization. *CoRR*, abs/2205.12442, 2022. doi: 10.48550/arXiv.2205.12442. URL `https://doi.org/10.48550/arXiv.2205.12442`.

Donglei Du, Zhicheng Liu, Chenchen Wu, Dachuan Xu, and Yang Zhou. An improved approximation algorithm for maximizing a DR-submodular function over a convex set. *arXiv preprint arXiv:2203.14740*, 2022.

Christoph Dürr, Nguyên Kim Thǎng, Abhinav Srivastav, and Léo Tible. Non-monotone DR-submodular maximization over general convex sets. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2148–2154, 2021.

Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing non-monotone submodular functions. *SIAM J. Comput.*, 40(4):1133–1153, 2011.

Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *Foundations of Computer Science (FOCS)*, pages 570–579, 2011.

Hamed Hassani, Mahdi Soltanolkotabi, and Amin Karbasi. Gradient methods for submodular maximization. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017a.

S. Hamed Hassani, Mahdi Soltanolkotabi, and Amin Karbasi. Gradient methods for submodular maximization. *CoRR*, abs/1708.03949, 2017b. URL `http://arxiv.org/abs/1708.03949`.

Ehsan Kazemi, Shervin Minaee, Moran Feldman, and Amin Karbasi. Regularized submodular maximization at scale. In Marina Meila and Tong Zhang, editors, *International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 5356–5366. PMLR, 2021. URL `http://proceedings.mlr.press/v139/kazemi21a.html`.

Paolo Massa, Martino Salvetti, and Danilo Tomasoni. Bowling alone and trust decline in social network sites. In *IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC)*, pages 658–663. IEEE Computer Society, 2009. doi: 10.1109/DASC.2009.130. URL `https://doi.org/10.1109/DASC.2009.130`.

Siddharth Mitra, Moran Feldman, and Amin Karbasi. Submodular + concave. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, pages 11577–11591, 2021. URL `https://proceedings.`

neurips.cc/paper/2021/hash/602443a3d6907117d8b4a308844e963e-Abstract.html.

Loay Mualem and Moran Feldman. Using partial monotonicity in submodular maximization. *arXiv preprint arXiv:2202.03051*, 2022.

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

Rad Niazadeh, Tim Roughgarden, and Joshua R Wang. Optimal algorithms for continuous non-monotone submodular and DR-submodular maximization. *Journal of Machine Learning Research*, 21(1):4937–4967, 2020.

Tasuku Soma and Yuichi Yoshida. Non-monotone DR-submodular function maximization. In Satinder Singh and Shaul Markovitch, editors, *AAAI Conference on Artificial Intelligence*, pages 898–904. AAAI Press, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14483.

Nguyễn Kim Thắng and Abhinav Srivastav. Online non-monotone DR-submodular maximization. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 9868–9876. AAAI Press, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17186.

Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P. Gummadi. On the evolution of user interaction in facebook. In *ACM SIGCOMM Workshop on Social Networks (WOSN)*, August 2009.

Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM J. Comput.*, 42(1):265–304, 2013. doi: 10.1137/110832318. URL https://doi.org/10.1137/110832318.

Wei Xia, Juan-Carlos Vera, and Luis F. Zuluaga. Globally solving nonconvex quadratic programs via linear integer programming techniques. *INFORMS J. Comput.*, 32(1):40–56, 2020. doi: 10.1287/ijoc.2018.0883. URL https://doi.org/10.1287/ijoc.2018.0883.

Yelp. Yelp Dataset. https://www.yelp.com/dataset, 2019.

Mingrui Zhang, Lin Chen, Hamed Hassani, and Amin Karbasi. Online continuous submodular maximization: From full-information to bandit feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

A PROOF OF LEMMA 2.2

In this section we prove Lemma 2.2, which we repeat here for convenience.

Lemma 2.2. *For every two vectors* $\mathbf{x}, \mathbf{y} \in [0,1]^n$ *and any continuously differentiable non-negative DR-submodular function* $F\colon [0,1]^n \to \mathbb{R}_{\geq 0}$, $F(\mathbf{x} \vee \mathbf{y}) \geq (1 - \|\mathbf{x}\|_\infty)F(\mathbf{y})$.

Proof. If $\|\mathbf{x}\|_\infty = 0$, then $\mathbf{x}$ is the all zeros vector, and the lemma becomes trivial. Thus, we may assume in the rest of this proof that $\|\mathbf{x}\|_\infty > 0$. Let $\mathbf{z} = \mathbf{x} \vee \mathbf{y} - \mathbf{y}$. Then,

$$\begin{aligned} F(\mathbf{x} \vee \mathbf{y}) - F(\mathbf{y}) &= \int_0^1 \left.\frac{dF(\mathbf{y} + r \cdot \mathbf{z})}{dr}\right|_{r=t} dt = \int_0^1 \sum_{i=1}^n \langle \mathbf{z}, \nabla F(\mathbf{y} + t \cdot \mathbf{z})\rangle dt \\ &= \|\mathbf{x}\|_\infty \cdot \int_0^{1/\|\mathbf{x}\|_\infty} \sum_{i=1}^n \langle \mathbf{z}, \nabla F(\mathbf{y} + \|\mathbf{x}\|_\infty \cdot t' \cdot \mathbf{z})\rangle dt' \\ &\geq \|\mathbf{x}\|_\infty \cdot \int_0^{1/\|\mathbf{x}\|_\infty} \sum_{i=1}^n \langle \mathbf{z}, \nabla F(\mathbf{y} + t' \cdot \mathbf{z})\rangle dt' \enspace , \end{aligned} \tag{1}$$

where the last equality holds by changing the integration variable to $t' = t/\|\mathbf{x}\|_\infty$, and the inequality follows from the DR-submodularity of F because $\mathbf{y} + t' \cdot \mathbf{z} \in [0,1]^n$. To see that the last inclusion holds, note that, for every $i \in [n]$, if $x_i \leq y_i$, then $y_i + t' \cdot z_i = y_i \leq 1$, and if $x_i \geq y_i$, then

$$y_i + t' \cdot z_i \leq y_i + \frac{z_i}{\|\mathbf{x}\|_\infty} = y_i + \frac{x_i - y_i}{\|\mathbf{x}\|_\infty} \leq \frac{x_i}{\|\mathbf{x}\|_\infty} \leq 1 \enspace .$$

Observe now that we also have

$$\begin{aligned} \int_0^{1/\|\mathbf{x}\|_\infty} \sum_{i=1}^n \langle \mathbf{z}, \nabla F(\mathbf{y} + t' \cdot \mathbf{z})\rangle dt' &= \int_0^{1/\|\mathbf{x}\|_\infty} \left.\frac{dF(\mathbf{y} + r \cdot \mathbf{z})}{dr}\right|_{r=t'} dt' \\ &= F\left(\mathbf{y} + \frac{\mathbf{z}}{\|\mathbf{x}\|_\infty}\right) - F(\mathbf{y}) \geq -F(\mathbf{y}) \enspace , \end{aligned}$$

where the inequality follows from the non-negativity of F. The lemma now follows by plugging this inequality into Inequality (1), and rearranging. □

B MISSING PROOFS OF SECTION 3

B.1 Proof of Observation 3.2

In this section we prove observation 3.2, which we repeat here for convenience.

Observation 3.2. *For every integer* $0 \leq i \leq T$, $1 - \|\mathbf{y}^{(i)}\|_\infty \geq (1 - \varepsilon)^i \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty)$.

Proof. To prove the observation, we show by induction that for every fixed coordinate $j \in [n]$, we have $1 - y_j^{(i)} \geq (1 - \varepsilon)^i \cdot (1 - y_j^{(0)})$. For $i = 0$, this inequality trivially holds. Furthermore, assuming this inequality holds for $i - 1$, it also holds for i because

$$\begin{aligned} 1 - y_j^i &= 1 - (1 - \varepsilon) y_j^{(i-1)} - \varepsilon s_j^{(i)} \\ &\geq 1 - (1 - \varepsilon) y_j^{(i-1)} - \varepsilon \\ &= (1 - \varepsilon)(1 - y_j^{(i-1)}) \\ &\geq (1 - \varepsilon)^i \cdot (1 - y_j^{(0)}) \enspace , \end{aligned}$$

where the second inequality follows from the induction hypothesis. □

B.2 Proof of Lemma 3.3

In this section we prove Lemma 3.3, which we repeat here for convenience.

Lemma 3.3. *For every integer* $1 \leq i \leq T$, $F(\mathbf{y}^{(i)}) \geq (1-2\varepsilon) \cdot F(\mathbf{y}^{(i-1)}) + \varepsilon(1-\varepsilon)^{i-1} \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2\beta D^2$.

Proof. By the chain rule,

$$\begin{aligned}
F(\mathbf{y}^{(i)}) - F(\mathbf{y}^{(i-1)}) &= F((1-\varepsilon)\cdot \mathbf{y}^{(i-1)} + \varepsilon \cdot \mathbf{s}^{(i)}) - F(\mathbf{y}^{(i-1)}) \\
&= \int_0^\varepsilon \left.\frac{F((1-z)\cdot \mathbf{y}^{(i-1)} + z\cdot \mathbf{s}^{(i)})}{dz}\right|_{z=r} dr \\
&= \int_0^\varepsilon \langle \mathbf{s}^{(i)} - \mathbf{y}^{(i-1)}, \nabla F((1-r)\cdot \mathbf{y}^{(i-1)} + r\cdot \mathbf{s}^{(i)})\rangle dr \\
&\geq \int_0^\varepsilon \left[\langle \mathbf{s}^{(i)} - \mathbf{y}^{(i-1)}, \nabla F(\mathbf{y}^{(i-1)})\rangle - r\beta D^2\right] dr \\
&= \varepsilon \cdot \langle \mathbf{s}^{(i)} - \mathbf{y}^{(i-1)}, \nabla F(\mathbf{y}^{(i-1)})\rangle - 0.5\varepsilon^2\beta D^2,
\end{aligned}$$

where the inequality follows from the β-smoothness of F. Recall now that $s^{(i)}$ is the maximizer found by Algorithm 1 in its i-th iteration, and $\mathbf{o}$ is one of the values in the domain on which the maximum is calculated. Therefore,

$$\begin{aligned}
F(\mathbf{y}^{(i)}) - F(\mathbf{y}^{(i-1)}) &\geq \varepsilon \cdot \langle \mathbf{s}^{(i)} - \mathbf{y}^{(i-1)}, \nabla F(\mathbf{y}^{(i-1)})\rangle - 0.5\varepsilon^2\beta D^2 \\
&\geq \varepsilon \cdot \langle \mathbf{o} - \mathbf{y}^{(i-1)}, \nabla F(\mathbf{y}^{(i-1)})\rangle - 0.5\varepsilon^2\beta D^2 \\
&\geq \varepsilon \cdot \left[F(\mathbf{o} \vee \mathbf{y}^{(i-1)}) + F(\mathbf{o} \wedge \mathbf{y}^{(i-1)}) - 2F(\mathbf{y}^{(i-1)})\right] - 0.5\varepsilon^2\beta D^2 \\
&\geq \varepsilon \cdot \left[(1-\varepsilon)^{i-1} \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 2F(\mathbf{y}^{(i-1)})\right] - 0.5\varepsilon^2\beta D^2.
\end{aligned}$$

where the third inequality follows from Lemma 2.1, and the last inequality from Lemma 2.2, Observation 3.2 and the non-negativity of F. The lemma now follows by rearranging the last inequality. □

B.3 Proof of Theorem 3.1

In this section we prove Theorem 3.1, which we repeat here for convenience.

Theorem 3.1. *Let $\mathcal{K} \subseteq [0,1]^n$ be a general convex set, and let $F\colon [0,1]^n \to \mathbb{R}_{\geq 0}$ be a non-negative β-smooth DR-submodular function. Then,* `Non-mon. Frank-Wolfe` *(Algorithm 1) outputs a solution $\mathbf{w} \in \mathcal{K}$ obeying*

$$\begin{aligned}
F(\mathbf{w}) \geq (1-2\varepsilon)^{T-1}[(1+\varepsilon)^T - 1](1 - \min_{\mathbf{x}\in\mathcal{K}} \|\mathbf{x}\|_\infty) \cdot F(\mathbf{o}) \\
- 0.5\varepsilon^2\beta D^2 T \ ,
\end{aligned}$$

where D is the diameter of $\mathcal{K}$ and $\mathbf{o} \in \arg\max_{\mathbf{x}\in\mathcal{K}} F(\mathbf{x})$. In particular, when T is set to be $\lfloor \ln 2/\varepsilon \rfloor$,

$$F(\mathbf{w}) \geq (1/4 - 3\varepsilon)(1 - \min_{\mathbf{x}\in\mathcal{K}} \|\mathbf{x}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon\beta D^2 \ .$$

Proof. To see that the second part of the theorem follows from the first part, note that for $T = \lfloor \ln 2/\varepsilon \rfloor$ and $\varepsilon < 1/4$,

$$\begin{aligned}
(1-2\varepsilon)^{T-1}[(1+\varepsilon)^T - 1] &\geq e^{-2\varepsilon T}(1 - 4\varepsilon^2 T)[e^{\varepsilon T}(1 - \varepsilon^2 T) - 1] \\
&\geq e^{-2\ln 2}(1 - 4\varepsilon \ln 2)[e^{\ln 2 - \varepsilon}(1 - \varepsilon \ln 2) - 1] \\
&= \left(\frac{1}{4} - \varepsilon \ln 2\right)\left[\frac{2 - 2\varepsilon \ln 2}{e^\varepsilon} - 1\right] \\
&\geq \left(\frac{1}{4} - \varepsilon\right)\left[\frac{2 - 2\varepsilon}{1 + 2\varepsilon} - 1\right] \\
&= \left(\frac{1}{4} - \varepsilon\right) \cdot \frac{1 - 4\varepsilon}{1 + 2\varepsilon} \\
&\geq \frac{1}{4} - 3\varepsilon \ .
\end{aligned}$$

For $\varepsilon \geq 1/4$, the second part of the theorem is an immediate consequence of the non-negaitivity of F.

It remains to prove the first part of the theorem. We do that by proving by induction the stronger claim that for every integer $0 \leq i \leq T$,

$$F(\mathbf{y}^{(i)}) \geq (1-2\varepsilon)^{i-1}\big[(1+\varepsilon)^i - 1\big] \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2\beta D^2 i \ . \tag{2}$$

Note that the theorem indeed follows from this claim because w is the best vector within a set that includes $\mathbf{y}^{(T)}$, and $\mathbf{y}^{(0)} \in \arg\min_{\mathbf{x}\in\mathcal{K}} \|x\|_\infty$. For $i = 0$, Equation (2) follows directly from the non-negativity of F. Hence, we only need to show that for $1 \leq i \leq T$, if we assume that Equation (2) holds for $i - 1$, then it holds for i as well. This is indeed the case because Lemma 3.3 yields

$$\begin{aligned}
F(\mathbf{y}^{(i)}) &\geq (1-2\varepsilon) \cdot F(\mathbf{y}^{(i-1)}) + \varepsilon(1-\varepsilon)^{i-1} \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2\beta D^2\\
&\geq (1-2\varepsilon) \cdot \{(1-2\varepsilon)^{i-2}\big[(1+\varepsilon)^{i-1} - 1\big] \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2\beta D^2(i-1)\}\\
&\quad + \varepsilon(1-\varepsilon)^{i-1} \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2\beta D^2\\
&\geq \{(1-2\varepsilon)^{i-1}\big[(1+\varepsilon)^i - \varepsilon(1+\varepsilon)^{i-1} - 1\big] + \varepsilon(1-\varepsilon)^{i-1}\} \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2\beta D^2 i\\
&\geq (1-2\varepsilon)^{i-1}\big[(1+\varepsilon)^i - 1\big] \cdot (1 - \|\mathbf{y}^{(0)}\|_\infty) \cdot F(\mathbf{o}) - 0.5\varepsilon^2\beta D^2 i \ ,
\end{aligned}$$

where the second inequality follows from the induction hypothesis, and the last inequality holds since

$$(1-2\varepsilon)^{i-1} \cdot \varepsilon(1+\varepsilon)^{i-1} = \varepsilon(1-\varepsilon-2\varepsilon^2)^{i-1} \leq \varepsilon(1-\varepsilon)^{i-1} \ . \qquad \square$$

C CONTINUING THE PROOF OF THEOREM 5.1

In this section, we complete the proof of Theorem 5.1. As explained in Section 5, the proof of Theorem 5.1 is based on showing that: (i) by "scrambling" the basic instances defined in Section 5 in an appropriate way, they can be made indistinguishable, (ii) the scrambled instances obey the properties assumed in the theorem, and (iii) there is a large gap between the optimal values of scrambled instances derived from the two basic instances. Towards this goal, we first study the properties of the basic instances themselves, and the gap between their optimal values. Let us begin with the following lemma, which gives some properties of the objective functions of the basic instances.

Lemma C.1. *The functions $\hat{F}_k$ and $\hat{G}_k$ are continuously differentiable, non-negative and DR-submodular. Furthermore, they are β-smooth for a value β that is polynomial in k.*

Proof. The non-negativity of $\hat{F}_k$ and $\hat{G}_k$ is explicitly guaranteed by Lemma 5.2, and Part 4 of the lemma shows that $\hat{F}$ and $\hat{G}$ are also continuously differentiable. Finally, Parts 4 and 6 of Lemma 5.2 imply together that $\hat{F}_k$ and $\hat{G}_k$ are DR-submodular (see the proof of Lemma 3.1 of Vondrák (2013) for a formal argument).

It remains to bound the smoothness of $\hat{F}_k$ and $\hat{G}_k$. Notice that the following claim implies that both functions are β-smooth for a β value that is polynomial in k. Unfortunately, the proof of this claim is technically quite involved (and not very insightful) as it requires us to look into the proof Lemma 5.2, and therefore, we defer the proof of this claim to Section C.1.

Claim C.2. *The absolute values of the second order partial derivatives of the functions $\hat{F}_k$ and $\hat{G}_k$ are bounded by $16k+2$ almost everywhere, and therefore, both functions are β-smooth for a β value that is polynomial in k.* $\square$

Next, we observe that the common constraint polytope of the basic instances is solvable since $\mathcal{P}_{h,k}$ is a polytope over $2k$ variables defined as the convex-hall of $k + 1$ vectors. The next observation proves another property of this polytope.

Observation C.3. *If $k \geq 1/(1-h)$, $\min_{\mathbf{x}\in\mathcal{P}_{h,k}} \|\mathbf{x}\|_\infty = h$.*

Proof. Since $\mathbf{u} \in \mathcal{P}_{h,k}$, $\min_{\mathbf{x}\in\mathcal{P}_{h,k}} \|\mathbf{x}\|_\infty \leq h$. Thus, we only need to show that no point in $\mathcal{P}_{h,k}$ has an infinity norm less than h. Recall that every point in $\mathcal{P}_{h,k}$ is a convex combination $\sum_{i=1}^k c_i\mathbf{v}^{(i)} + d\mathbf{u}$ (where c_i is the coefficient of $\mathbf{v}^{(i)}$ in the combination, and d is the coefficient of $\mathbf{u}$), and assume without loss of generality that $c_1 = \min\{c_1, c_2, \ldots, c_k\}$. Then,

$$\left\|\sum_{i=1}^k c_i\mathbf{v}^{(i)} + d\mathbf{u}\right\|_\infty \geq \sum_{i=1}^k c_i v_{b_1}^{(i)} + du_{b_1} = \sum_{i=2}^k c_i + dh \geq \frac{k-1}{k}\sum_{i=1}^k c_i + dh \geq h\sum_{i=1}^k c_i + dh = h \ ,$$

where the last inequality holds by the condition of the observation, and the last equality holds since the fact that $\sum_{i=1}^k c_i\mathbf{v}^{(i)} + d\mathbf{u}$ is a convex combination implies $\sum_{i=1}^k c_i + d = 1$. $\square$

The last properties that we need to prove for the basic instances are about the optimal values of these instances. Specifically, we need to show that both their optimal values are significant (at least $\Omega(k^{-1})$), but there is a large gap between them. The following two lemmata show these properties, respectively.

Lemma C.4. $\max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{F}_k(\mathbf{x}) = \Omega(k^{-1})$ *and* $\max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{G}_k(\mathbf{x}) = \Omega(k^{-1})$.

Proof. We prove the lemma by considering the vector $\mathbf{y} = \frac{1}{k}\sum_{i=1}^{k} \mathbf{u}^{(i)}$. Since $\mathbf{y} \in \mathcal{P}_{h,k}$ and $\bar{\mathbf{y}} = \mathbf{y}$, $\hat{F}_k(\mathbf{y})$ lower bounds both $\max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{F}_k(\mathbf{x})$ and $\max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{G}_k(\mathbf{x})$. Thus, it remains to show that $\hat{F}_k(\mathbf{y}) = \Omega(k^{-1})$. By Lemma 5.2,

$$\hat{F}_k(\mathbf{y}) \geq F_k(\mathbf{y}) - \varepsilon' = \sum_{i=1}^{k} y_{a_i}(1-b_i) - \varepsilon' = \sum_{i=1}^{k} \frac{1}{k}\cdot\left(1-\left(1-\frac{1}{k}\right)\right) - \varepsilon' = \frac{1}{k} - \varepsilon' = \frac{1}{2k} \enspace ,$$

where F_k is the multilinear extension of f_k. □

Lemma C.5. $\max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{F}_k(\mathbf{x}) \geq 1 - 1/(2k)$ *and* $\max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{G}_k(\mathbf{x}) \leq (1-h)/4 + 3/(2k)$.

Proof. To prove the first part of the lemma, it suffices to observe that $\mathbf{v}^{(1)} \in \mathcal{P}_{h,k}$ and

$$\hat{F}_k(\mathbf{v}^{(1)}) \geq F_k(\mathbf{v}^{(1)}) - \varepsilon' = f_k(\{a_1\} \cup \{b_i \mid 2 \leq i \leq k\}) - \varepsilon' = 1 - 1/(2k) \enspace ,$$

where F_k is the multilinear extension of f_k.

Let us now prove the second part of the lemma. Fix an arbitrary vector $\mathbf{x} \in \mathcal{P}_{h,k}$, and let d be the coefficient of $\mathbf{u}$ in the convex combination that shows that $\mathbf{x}$ belongs to $\mathcal{P}_{h,k}$. Then,

$$\sum_{i=1}^{k} x_{a_i} = 1 - d \qquad \text{and} \qquad \sum_{i=1}^{k} x_{b_i} = dkh + (1-d)(k-1) = k(dh + 1 - d) + d - 1 \enspace .$$

Thus,

$$\begin{aligned}\hat{G}_k(\mathbf{x}) = \hat{F}_k(\bar{\mathbf{x}}) \leq F_k(\bar{\mathbf{x}}) + \varepsilon' &= \sum_{1=1}^{k} \frac{\sum_{i=1}^{k} x_{a_i}}{k}\left(1 - \frac{\sum_{i=1}^{k} x_{b_i}}{k}\right) + \varepsilon' \\ &= (1-d)\left(d - dh + \frac{1-d}{k}\right) + \varepsilon' \leq d(1-d)(1-h) + \frac{1}{k} + \varepsilon' \leq \frac{1-h}{4} + \frac{3}{2k} \enspace .\end{aligned}$$

□

We now would like to describe how the two basic instances are scrambled. Intuitively, the constraint polytope $\mathcal{K}_{h,k,\ell}$ of a scrambled instance is obtained by combining ℓ orthogonal instances of $\mathcal{P}_{h,k}$. Each element a_i or b_i has a copy in all the orthogonal instances, and the objective function treats every such copy as representing ℓ^{-1} of the original element. For example, if one would like to construct a solution assigning a value of $1/2$ to a_i, then the copies of a_i in $\mathcal{K}_{h,k,\ell}$ should get an average value of $1/2$. By randomly permuting the names of the elements in each orthogonal instance of $\mathcal{P}_{h,k}$, we make it difficult for the algorithm to construct solutions that do not correspond to symmetric vectors in $\mathcal{P}_{h,k}$. More formally, the constraint polytope $\mathcal{K}_{h,k,\ell}$ is a subset of $[0,1]^{\mathcal{M}_{k,\ell}}$, where

$$\mathcal{M}_{k,\ell} = \{a_{i,j}, b_{i,j} \mid i \in [k], j \in [\ell]\} \enspace .$$

A vector $\mathbf{x} \in [0,1]^{\mathcal{M}_{k,\ell}}$ belongs to $\mathcal{K}_{h,k,\ell}$ if for every $j \in [\ell]$ we have $\mathbf{x}^{(j)} \in \mathcal{P}_{h,k}$, where the vector $\mathbf{x}^{(j)} \in [0,1]^{\mathcal{N}_k}$ is defined by

$$\mathbf{x}^{(j)}_{a_i} = \mathbf{x}_{a_{i,j}} \qquad \text{and} \qquad \mathbf{x}^{(j)}_{b_i} = \mathbf{x}_{b_{i,j}} \enspace .$$

The following lemma is an immediate corollary of the definition of $\mathcal{K}_{h,k,\ell}$, Observation C.3 and the discussion before this observation.

Lemma C.6. *When* $k \geq 1/(1-h)$, $\mathcal{K}_{h,k,\ell}$ *is solvable and* $\max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \|\mathbf{x}\|_\infty = h$.

The objective functions of the scrambled instances are formally defined using a vector $\boldsymbol{\sigma}$ of ℓ permutations over $[k]$ (in other words, $\sigma_1, \sigma_2, \ldots, \sigma_\ell$ are all permutations over $[k]$). Given such a vector $\boldsymbol{\sigma}$ and a vector $\mathbf{x} \in [0,1]^{\mathcal{M}_{k,\ell}}$, we define the vector $\mathbf{x}^{(\boldsymbol{\sigma})} \in [0,1]^{\mathcal{N}_k}$ as follows.

$$\mathbf{x}^{(\boldsymbol{\sigma})}_{a_i} = \tfrac{1}{\ell}\sum_{j=1}^{\ell} \mathbf{x}_{a_{\sigma_j(i),j}} \qquad \text{and} \qquad \mathbf{x}^{(\boldsymbol{\sigma})}_{b_i} = \tfrac{1}{\ell}\sum_{j=1}^{\ell} \mathbf{x}_{b_{\sigma_j(i),j}} \enspace .$$

Then, the functions $\bar{F}_{k,\boldsymbol{\sigma}} \colon [0,1]^{\mathcal{M}_{k,\ell}} \to \mathbb{R}_{\geq 0}$ and $\bar{G}_{k,\boldsymbol{\sigma}} \colon [0,1]^{\mathcal{M}_{k,\ell}} \to \mathbb{R}_{\geq 0}$ are defined for every vector $\mathbf{x} \in [0,1]^{\mathcal{M}_{k,\ell}}$ by

$$\bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) = \hat{F}(\mathbf{x}^{(\boldsymbol{\sigma})}) \qquad \text{and} \qquad \bar{G}_{k,\boldsymbol{\sigma}}(\mathbf{x}) = \hat{G}(\mathbf{x}^{(\boldsymbol{\sigma})}) \enspace .$$

The following lemma shows that the functions $\bar{F}_{k,\boldsymbol{\sigma}}$ and $\bar{G}_{k,\boldsymbol{\sigma}}$ inherit all the good properties of $\hat{F}_k$ and $\hat{G}_k$ promised by Lemma C.1. Since the proof of this lemma is technical and quite straightforward given Lemma C.1, we defer it to Section C.1.

Lemma C.7. *The functions $\bar{F}_{k,\boldsymbol{\sigma}}$ and $\bar{G}_{k,\boldsymbol{\sigma}}$ are continuously differentiable, non-negative and DR-submodular. Furthermore, they are β-smooth for a value β that is polynomial in k and ℓ.*

We can now formally state the scrambled instances that we consider.

$$\begin{array}{l} \max \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) \\ \mathbf{x} \in \mathcal{K}_{h,k,\ell} \end{array} \qquad \text{and} \qquad \begin{array}{l} \max \bar{G}_{k,\boldsymbol{\sigma}}(\mathbf{x}) \\ \mathbf{x} \in \mathcal{K}_{h,k,\ell} \end{array} \enspace .$$

The next lemma shows that these scrambled instances inherit the values of their optimal solutions from the basic instances, which in particular, implies that they also inherit the gap between these solutions.

Lemma C.8. *We have both* $\max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) = \max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{F}_k(\mathbf{x})$ *and* $\max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \bar{G}_{k,\boldsymbol{\sigma}}(\mathbf{x}) = \max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{G}_k(\mathbf{x})$.

Proof. We prove below only the first equality of the lemma. The proof of the other equality is analogous. We begin by arguing that $\max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) \geq \max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{F}_k(\mathbf{x})$. To show this inequality, we start with an arbitrary vector $\mathbf{x} \in \mathcal{P}_{h,k}$, and we construct a vector $\mathbf{y} \in \mathcal{K}_{h,k,\ell}$ such that $\bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{y}) = \hat{F}_k(\mathbf{x})$. Formally, the vector $\mathbf{y}$ is defined as follows. For every $i \in [k]$ and $j \in [\ell]$,

$$\mathbf{y}_{a_{i,j}} = \mathbf{x}_{a_{\sigma_j^{-1}(i)}} \qquad \text{and} \qquad \mathbf{y}_{b_{i,j}} = \mathbf{x}_{b_{\sigma_j^{-1}(i)}} \enspace .$$

One can observe that $\mathbf{x} = \mathbf{y}^{(\boldsymbol{\sigma})}$, and therefore, we indeed have $\bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{y}) = \hat{F}_k(\mathbf{x})$; which means that we are only left to show that $\mathbf{y} \in \mathcal{K}_{h,k,\ell}$. Recall that, by the definition of $\mathcal{K}_{h,k,\ell}$, to prove this inclusion, we need to argue that $\mathbf{y}^{(j)} \in \mathcal{P}_{h,k}$ for every $j \in [\ell]$, where $\mathbf{y}^{(j)}$ is the restriction of $\mathbf{y}$ to elements of $\{a_{i,j}, b_{i,j} \mid i \in [k]\}$.

Below, given a vector $\mathbf{z} \in \mathcal{P}_{h,k}$, we denote by $\sigma_j(\mathbf{z})$ the following vector.

$$(\sigma_j(\mathbf{z}))_{a_i} = \mathbf{z}_{a_{\sigma_j^{-1}(i)}} \qquad \text{and} \qquad (\sigma_j(\mathbf{z}))_{b_i} = \mathbf{z}_{b_{\sigma_j^{-1}(i)}} \enspace .$$

Observe that this definition implies $\sigma_j(\mathbf{u}) = \mathbf{u}$ and $\sigma_j(\mathbf{v}^{(i)}) = \mathbf{v}^{(\sigma_j(i))}$, where $\mathbf{u}, \mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \ldots, \mathbf{v}^{(k)}$ are the vectors whose convex-hall defines $\mathcal{P}_{h,k}$. Since $\mathbf{x} \in \mathcal{P}_{h,k}$, it must be given by some convex combination of the vectors $\mathbf{u}, \mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \ldots, \mathbf{v}^{(k)}$. In other words,

$$\mathbf{x} = \sum_{i=1}^{k} c_i \cdot \mathbf{v}^{(i)} + d \cdot \mathbf{u} \enspace .$$

Thus,

$$\mathbf{y}^{(j)} = \sigma_j(\mathbf{x}) = \sigma_j\left(\sum_{i=1}^{k} c_i \cdot \mathbf{v}^{(i)} + d \cdot \mathbf{u}\right) = \sum_{i=1}^{k} c_i \cdot \mathbf{v}^{(\sigma_j(i))} + d \cdot \mathbf{u} \enspace .$$

The rightmost side of the last equality is another convex combination of the vectors $\mathbf{u}, \mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \ldots, \mathbf{v}^{(k)}$, and thus, the equality shows that $\mathbf{y}^{(j)} \in \mathcal{P}_{h,k}$, as desired.

We now get to the proof that $\max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) \leq \max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{F}_k(\mathbf{x})$. Consider an arbitrary vector $\mathbf{x} \in \mathcal{K}_{h,k,\ell}$. By the definition of $\bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x})$, $\bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) = \hat{F}_k(\mathbf{x}^{(\boldsymbol{\sigma})})$. Thus, to prove the last inequality, it suffices to show that $\mathbf{x}^{(\boldsymbol{\sigma})} \in \mathcal{P}_{h,k}$, which is done by the next claim. Since the proof of this claim is very similar to the above proof that $\mathbf{y} \in \mathcal{K}_{h,k,\ell}$, we defer it to Section C.1.

Claim C.9. *For every vector* $\mathbf{x} \in \mathcal{K}_{h,k,\ell}$, $\mathbf{x}^{(\boldsymbol{\sigma})} \in \mathcal{P}_{h,k}$. □

Corollary C.10. *It holds that* $\max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) \geq 1 - 1/(2k) = \Omega(k^{-1})$ *and* $(1-h)/4 + 3/(2k) \geq \max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \bar{G}_{k,\boldsymbol{\sigma}}(\mathbf{x}) = \Omega(k^{-1})$.

Lemmata C.6, C.7 and C.8 show that the scrambled instances we have constructed have all the properties stated in Theorem 5.1 when $k \geq 1/(1-h)$). Therefore, to prove the theorem it suffices to show that no sub-exponential time algorithm can obtain a good approximation guarantee given these instances when ℓ is large enough compared to k. We do this by showing that when $\boldsymbol{\sigma}$ is chosen uniformly at random, it is difficult to distinguish between the two scrambled instances, and therefore, no sub-exponential time algorithm can obtain an approximation ratio better than the (large) gap between their optimal values. The first step in this proof is done by the next lemma, which shows that any single access to the objective function almost always returns the same answer given either of the two scrambled instances. To understand why the lemma implies this, it is important to recall that we assume that the algorithm is able to access F only by making queries whose outputs are determined by the values of F in an arbitrary small neighborhood of a given point $\mathbf{x}$ (this kind of queries includes the standard value and gradient queries).

Lemma C.11. *Assume $\boldsymbol{\sigma}$ is drawn uniformly at random, i.e., σ_j is an independently chosen uniformly random permutation of $[k]$ for every $j \in [\ell]$. Given any vector $\mathbf{x} \in [0,1]^{\mathcal{M}_k}$, with probability at least $1 - 4k \cdot e^{-\ell \cdot \frac{\delta_k}{6\sqrt{2k}}}$ we have $\bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{y}) = \bar{G}_{k,\boldsymbol{\sigma}}(\mathbf{y})$ for every vector $\mathbf{y}$ such that $\|\mathbf{x}-\mathbf{y}\|_2 \leq (\sqrt{\ell}/4)\cdot\delta_k$, where δ_k is the value of δ when Lemma 5.2 is applied to f_k.*

Proof. Below, we show that $\|\mathbf{x}^{(\boldsymbol{\sigma})} - \bar{\mathbf{x}}^{(\boldsymbol{\sigma})}\|_2 \leq \delta_k/2$ with probability at least $1 - 4k\cdot e^{-\ell\delta_k/(6\sqrt{2k})}$. However, before getting to this proof, let us show that, whenever this inequality holds, we also have $\bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{y}) = \bar{G}_{k,\boldsymbol{\sigma}}(\mathbf{y})$. By the definitions of $\bar{F}_{k,\boldsymbol{\sigma}}$ and $\bar{G}_{k,\boldsymbol{\sigma}}$, the last equality is equivalent to $\hat{F}_k(\mathbf{y}^{(\boldsymbol{\sigma})}) = \hat{G}_k(\mathbf{y}^{(\boldsymbol{\sigma})})$, and this equality holds by Lemma 5.2 since

$$\|\mathbf{y}^{(\boldsymbol{\sigma})} - \bar{\mathbf{y}}^{(\boldsymbol{\sigma})}\|_2 \leq \|\mathbf{y}^{(\boldsymbol{\sigma})} - \mathbf{x}^{(\boldsymbol{\sigma})}\|_2 + \|\bar{\mathbf{y}}^{(\boldsymbol{\sigma})} - \bar{\mathbf{x}}^{(\boldsymbol{\sigma})}\|_2 + \|\mathbf{x}^{(\boldsymbol{\sigma})} - \bar{\mathbf{x}}^{(\boldsymbol{\sigma})}\|_2 \leq 2\|\mathbf{y}^{(\boldsymbol{\sigma})} - \mathbf{x}^{(\boldsymbol{\sigma})}\|_2 + \delta_k/2 \leq \delta_k \enspace,$$

where the first inequality is the triangle inequality, the second inequality holds since averaging two vectors in the same way can only decrease their distance from each other, and the last inequality holds because Sedrakyan's inequality (or Cauchy-Schwarz inequality) implies

$$\begin{aligned}\|\mathbf{y}^{(\boldsymbol{\sigma})} - \mathbf{x}^{(\boldsymbol{\sigma})}\|_2^2 &= \frac{\sum_{i=1}^{k}[\sum_{j=1}^{\ell}(\mathbf{y}_{a_{\sigma_j(i),j}} - \mathbf{x}_{a_{\sigma_j(i),j}})]^2 + \sum_{i=1}^{k}[\sum_{j=1}^{\ell}(\mathbf{y}_{b_{\sigma_j(i),j}} - \mathbf{x}_{b_{\sigma_j(i),j}})]^2}{\ell^2}\\ &\leq \frac{\sum_{i=1}^{k}\sum_{j=1}^{\ell}(\mathbf{y}_{a_{\sigma_j(i),j}} - \mathbf{x}_{a_{\sigma_j(i),j}})^2 + \sum_{i=1}^{k}\sum_{j=1}^{\ell}(\mathbf{y}_{b_{\sigma_j(i),j}} - \mathbf{x}_{b_{\sigma_j(i),j}})^2}{\ell} = \frac{\|\mathbf{x}-\mathbf{y}\|_2^2}{\ell} \enspace.\end{aligned}$$

It now remains to prove that the inequality $\|\mathbf{x}^{(\boldsymbol{\sigma})} - \bar{\mathbf{x}}^{(\boldsymbol{\sigma})}\|_2 \leq \delta_k/2$ holds with probability at least $1 - 4k\cdot e^{-\ell\delta_k/(6\sqrt{2k})}$. By the union bound, to prove this inequality it suffices to show that, for every $i \in [k]$, the probabilities of the two inequalities $|\mathbf{x}^{(\boldsymbol{\sigma})}_{a_i} - \bar{\mathbf{x}}^{(\boldsymbol{\sigma})}_{a_i}| > \delta_k/\sqrt{8k}$ and $|\mathbf{x}^{(\boldsymbol{\sigma})}_{b_i} - \bar{\mathbf{x}}^{(\boldsymbol{\sigma})}_{b_i}| > \delta_k/\sqrt{8k}$ to hold are both at most $2e^{-\ell\delta_k/(6\sqrt{2k})}$. The rest of this proof is devoted to showing that this is indeed the case for the first inequality as the proof for the second inequality is analogous. Recall that

$$\mathbf{x}^{(\boldsymbol{\sigma})}_{a_i} = \tfrac{1}{\ell}\sum_{j=1}^{\ell}\mathbf{x}_{a_{\sigma_j(i),j}} \enspace. \tag{3}$$

Thus,

$$\bar{\mathbf{x}}^{(\boldsymbol{\sigma})}_{a_i} = \frac{1}{k}\sum_{i'=1}^{k}\mathbf{x}^{(\boldsymbol{\sigma})}_{a_{i'}} = \frac{1}{k}\sum_{i'=1}^{k}\left(\tfrac{1}{\ell}\sum_{j=1}^{\ell}\mathbf{x}_{a_{\sigma_j(i'),j}}\right) = \frac{1}{k\ell}\sum_{i'=1}^{k}\sum_{j=1}^{\ell}\mathbf{x}_{a_{\sigma_j(i'),j}} = \frac{1}{k\ell}\sum_{i'=1}^{k}\sum_{j=1}^{\ell}\mathbf{x}_{a_{i',j}} \enspace, \tag{4}$$

where the last equality holds since σ_j is a permutation over $[k]$. Similarly, we also have

$$\mathbb{E}[\mathbf{x}^{(\boldsymbol{\sigma})}_{a_i}] = \frac{1}{\ell}\sum_{j=1}^{\ell}\mathbb{E}[\mathbf{x}_{a_{\sigma_j(i),j}}] = \frac{1}{\ell}\sum_{j=1}^{\ell}\left(\tfrac{1}{k}\sum_{i'=1}^{k}\mathbb{E}[\mathbf{x}_{a_{i',j}}]\right) = \bar{\mathbf{x}}^{(\boldsymbol{\sigma})}_{a_i} \enspace.$$

Hence, the claim that we want to prove bounds the probability that $\mathbf{x}^{(\boldsymbol{\sigma})}_{a_i}$ significantly deviates from its expectation. Furthermore, Equation (3) shows that $\ell\cdot\mathbf{x}^{(\boldsymbol{\sigma})}_{a_i}$ is the sum of ℓ random variables taking values from the range $[0,1]$. Since σ_j

is chosen independently for every $j \in [\ell]$, these ℓ random variables are independent, which allows us to use Chernoff's inequality to bound their sum. Therefore,

$$\begin{aligned}
\Pr\left[|\mathbf{x}^{(\boldsymbol{\sigma})}_{a_i} - \bar{\mathbf{x}}^{(\boldsymbol{\sigma})}_{a_i}| > \frac{\delta_k}{\sqrt{8k}}\right] &= \Pr\left[\left|\sum_{j=1}^{\ell} \mathbf{x}_{a_{\sigma_j(i),j}} - \mathbb{E}\left[\sum_{j=1}^{\ell} \mathbf{x}_{a_{\sigma_j(i),j}}\right]\right| > \frac{\ell\delta_k}{\sqrt{8k}}\right] \\
&\leq 2e^{-\frac{\mathbb{E}[\sum_{j=1}^{\ell} \mathbf{x}_{a_{\sigma_j(i),j}}]\cdot \min\left\{\frac{\ell\delta_k}{\sqrt{8k}\cdot\mathbb{E}[\sum_{j=1}^{\ell} \mathbf{x}_{a_{\sigma_j(i),j}}]}, \frac{\ell^2\delta_k^2}{8k\cdot\mathbb{E}[\sum_{j=1}^{\ell} \mathbf{x}_{a_{\sigma_j(i),j}}]^2}\right\}}{3}} \\
&= 2e^{-\frac{\min\left\{\frac{\ell\delta_k}{\sqrt{8k}}, \frac{\ell^2\delta_k^2}{8k\cdot\mathbb{E}[\sum_{j=1}^{\ell} \mathbf{x}_{a_{\sigma_j(i),j}}]}\right\}}{3}} \leq 2e^{-\frac{\frac{\ell\delta_k}{\sqrt{8k}}\cdot\min\left\{1, \frac{\delta_k}{\sqrt{8k}}\right\}}{3}} = 2e^{-\ell\cdot\frac{\delta_k}{6\sqrt{2k}}} \enspace . \qquad \square
\end{aligned}$$

Equation (4) in the last proof has another interesting consequence. This equation shows that $\bar{\mathbf{x}}^{(\boldsymbol{\sigma})}$ is independent of $\boldsymbol{\sigma}$. Since Lemma 5.2 shows that $\hat{G}_k(\mathbf{x}) = \hat{F}_k(\bar{\mathbf{x}})$ for every $\mathbf{x} \in [0,1]^{\mathcal{N}_k}$, this implies the following observation.

Observation C.12. *For every $\mathbf{x} \in [0,1]^{\mathcal{M}_{k,\ell}}$, the value of $\bar{G}_{k,\boldsymbol{\sigma}}(\mathbf{x}) = \hat{G}_k(\mathbf{x}^{(\boldsymbol{\sigma})}) = \hat{F}_k(\bar{\mathbf{x}}^{\boldsymbol{\sigma}})$ is independent of $\boldsymbol{\sigma}$.*

In light of the above observation, we use below $\bar{G}_k$ to denote the function $\bar{G}_{k,\boldsymbol{\sigma}}$. We are now ready to prove Theorem 5.1.

Proof of Theorem 5.1. Fix an arbitrary sub-exponential function $P(\cdot)$. Below, we show that there is a distribution of instances on which no deterministic algorithm making at most $P(n)$ accesses to the objective function, where n is the dimension, can obtain an approximation ratio of $(1-h)/4+\varepsilon$. By Yao's principle, this will imply the same result also for randomized algorithms running in time $P(n)$ (notice that running in time $P(n)$ implies making at most $P(n)$ accesses to the objective function).

The distribution of instances we consider is the scrambled instance $\max_{vx\in\mathcal{K}_{h,k,\ell}} \mathcal{F}_{k,\boldsymbol{\sigma}}$, where $k \geq 1/(1-h)$ and ℓ are deterministic values to be determined below, and $\boldsymbol{\sigma}$ is chosen at random according to the distribution defined in Lemma C.11. Assume towards a contradiction that there exists a deterministic algorithm ALG that accesses the objective function at most $P(|\mathcal{M}_{k,\ell}|) = P(2k\ell)$ times, and given a random instance from the above distribution obtains an approximation ratio of $(1-h)/4+\varepsilon$. More formally, if we denote $OPT = \max_{\mathbf{x}\in\mathcal{P}_{h,k}} \hat{F}_k(\mathbf{x})$, then ALG guarantees that its output vector $\mathbf{a}$ obeys

$$\mathbb{E}[\mathcal{F}_{k,\boldsymbol{\sigma}}(\mathbf{a})] \geq [(1-h)/4+\varepsilon]\cdot\mathbb{E}\left[\max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \hat{F}_{k,\boldsymbol{\sigma}}(\mathbf{x})\right] = [(1-h)/4+\varepsilon]\cdot OPT \enspace , \tag{5}$$

where the equality holds by Lemma C.8.

Consider now an execution of ALG on the instance $\max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \bar{G}_k(\mathbf{x})$, and let us denote by $A_1, A_2, \ldots, A_r$ the accesses made by ALG (each access A_i consists of a vector $\mathbf{x}$ and the type of access, namely whether ALG evaluates the objective function at $\mathbf{x}$ or calculates the gradient of the objective function at $\mathbf{x}$). It is convenient to assume that the last access made by ALG is to evaluate the value of its output set $\mathbf{a}$. If this is not the case, we can add such an access to the end of the execution of ALG, and still have $r \leq P(2k\ell)+1$. Let $\mathcal{E}$ be the event that all the accesses $A_1, A_2, \ldots, A_r$ return the same value given that the objective is either $\bar{G}_k$ or $\bar{F}_{k,\boldsymbol{\sigma}}$. Clearly, ALG follows the same execution path given either $\bar{G}_k$ or $\bar{F}_{k,\boldsymbol{\sigma}}$ when the event $\mathcal{E}$ happens, and therefore, it outputs the same vector $\mathbf{a} \in \mathcal{K}_{h,k,\ell}$ in this case. Furthermore, $\mathcal{E}$ also implies that $\bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{a}) = \bar{G}_k(\mathbf{a})$, and thus, conditioned on $\mathcal{E}$,

$$\begin{aligned}
\mathcal{F}_{k,\boldsymbol{\sigma}}(\mathbf{a}) \leq \max_{\mathbf{x}\in\mathcal{K}_{h,k,\ell}} \hat{G}_k(\mathbf{x}) \leq (1-h)/4 + 3/(2k) \leq \frac{(1-h)/4+3/(2k)}{1-1/(2k)}\cdot OPT \\
\leq \left[\frac{1-h}{4-2/k} + \frac{3}{k}\right]\cdot OPT \leq \left[\frac{1-h}{4} + \frac{4}{k}\right]\cdot OPT \enspace ,
\end{aligned}$$

where the second inequality holds by Corollary C.10, the third inequality follows from Lemma C.5, and two last inequalities hold since $k \geq 1$ and $h \in [0,1]$.

We would like to use the last inequality to upper bound $\mathbb{E}[\mathcal{F}_{k,\boldsymbol{\sigma}}(\mathbf{a})]$. For that purpose, we need to lower bound the probability of the event $\mathcal{E}$. By Lemma C.11 and the union bound,

$$\Pr[\mathcal{E}] \geq 1 - 4kr\cdot e^{-\ell\cdot\frac{\delta_k}{6\sqrt{2k}}} \geq 1 - 4k[P(2k\ell)+1]\cdot e^{-\ell\cdot\frac{\delta_k}{6\sqrt{2k}}} \enspace .$$

Consider the second term in the rightmost side of the last inequality. This term is a function of k and ℓ alone, and for a fixed value of k it is the product of a sub-exponential function of ℓ and an exponentially decreasing function of ℓ. Therefore, for any fixed value of k, we can choose a large enough value for ℓ to guarantee that $2k[P(2k\ell)+1]\cdot e^{-\ell\cdot\frac{\delta_k}{6\sqrt{k}}} \leq \varepsilon/2$. In the rest of the proof we assume that ℓ is chosen in such a way. Then, since we always have $\mathcal{F}_{k,\boldsymbol{\sigma}}(\mathbf{a}) \leq OPT$ and $\Pr[\mathcal{E}] \leq 1$, we get by the law of total expectation,

$$\mathbb{E}[\mathcal{F}_{k,\boldsymbol{\sigma}}(\mathbf{a})] \leq \Pr[\bar{\mathcal{E}}]\cdot OPT + \mathbb{E}[\mathcal{F}_{k,\boldsymbol{\sigma}}(\mathbf{a}) \mid \mathcal{E}] \leq \frac{\varepsilon}{2}\cdot OPT + [(1-h)/4 + 4/k]\cdot OPT \enspace ,$$

which contradicts Equation (5) (and thus, the existence of ALG) when k is chosen to be $\max\{\lceil 1/(h-1)\rceil, 8/\varepsilon\}$. $\square$

C.1 Missing Proofs

C.1.1 Proof of Claim C.2

In this section we prove Claim C.2, which we repeat here for convenience.

Claim C.2. *The absolute values of the second order partial derivatives of the functions $\hat{F}_k$ and $\hat{G}_k$ are bounded by $16k+2$ almost everywhere, and therefore, both functions are β-smooth for a β value that is polynomial in k.*

Proof. Recall that $\hat{F}_k$ and $\hat{G}_k$ are the functions $\hat{F}$ and $\hat{G}$ whose existence is guaranteed by Lemma 5.2 for $f = f_k$. The functions $\hat{F}$ and $\hat{G}$ are obtained in the proof of Lemma 5.2 in a series of steps involving multiple intermediate functions. The first of these functions are F (the multilinear extension of f), the function $G(\mathbf{x}) = F(\bar{\mathbf{x}})$ and the function $H(\mathbf{x}) = F(\mathbf{x}) - G(\mathbf{x})$. The proof of Lemma 3.5 of Vondrák (2013) shows that the absolute values of the second partial derivatives of these functions are bounded by $4M$, $4M$ and $8M$, respectively, where M is the maximum value that the function f can take. Since in our case f is f_k, the maximum value it can take is k, and therefore, the absolute values of the second partial derivatives of all three functions can be upper bounded by $8k$.

The next function we consider is a function denoted by $\tilde{F}$ in the proof of Lemma 5.2. The proof of Lemma 3.8 of Vondrák (2013) shows that for every two elements $u, v \in \mathcal{N}$, this function obeys almost everywhere the inequality

$$\left|\frac{\partial^2 \tilde{F}(\mathbf{x})}{\partial u \partial v} - \frac{\partial^2 F(\mathbf{x})}{\partial u \partial v} + \phi(D(\mathbf{x}))\cdot\frac{\partial^2 H(\mathbf{x})}{\partial u \partial v}\right| \leq 512M|\mathcal{N}|\alpha = \frac{512\varepsilon'}{2000|\mathcal{N}|^2} \leq 1 \enspace ,$$

where ϕ is a function defined by Vondrák (2013) whose range is $[0,1]$, $D(\mathbf{x})$ is another function defined by Vondrák (2013) and $\alpha = \varepsilon'/(2000M|\mathcal{N}|^3)$. Since $|\phi(D(\mathbf{x}))| \leq 1$, the last inequality implies that the absolute values of the second partial derivatives of $\tilde{F}$ are upper bounded by $16k+1$ because the second partial derivatives of F and H have absolute values bounded by $8k$.

The functions $\hat{F}$ and $\hat{G}$ are obtained from $\tilde{F}$ and G, respectively, by adding $256M|\mathcal{N}|\alpha J(\mathbf{x}) = \frac{256\varepsilon'}{2000|\mathcal{N}|^2}\cdot J(\mathbf{x})$, where

$$J(\mathbf{x}) = |\mathcal{N}|^2 + 3|\mathcal{N}|\|\mathbf{x}\|_1 - (\|\mathbf{x}\|_1)^2 \enspace .$$

Since the second order partial derivatives of $J(\mathbf{x})$ are all -2, and the coefficient of $J(\mathbf{x})$ is $\frac{256\varepsilon'}{2000|\mathcal{N}|^2} \leq 1/2$, adding $\frac{256\varepsilon'}{2000|\mathcal{N}|^2}\cdot J(\mathbf{x})$ cannot increase the absolute value of the second order partial derivatives by more than 1. $\square$

C.1.2 Proof of Lemma C.7

In this section we prove Lemma C.7, which we repeat here for convenience.

Lemma C.7. *The functions $\bar{F}_{k,\boldsymbol{\sigma}}$ and $\bar{G}_{k,\boldsymbol{\sigma}}$ are continuously differentiable, non-negative and DR-submodular. Furthermore, they are β-smooth for a value β that is polynomial in k and ℓ.*

Proof. We prove the lemma below for $\bar{F}_{k,\boldsymbol{\sigma}}$. The proof for $\bar{G}_{k,\boldsymbol{\sigma}}$ is analogous. The non-negativity of $\bar{F}_{k,\boldsymbol{\sigma}}$ follows immediately from their definitions and the non-negativity of $\hat{F}_k$ and $\hat{G}_k$. Furthermore, by the chain-rule, for every pair of $i \in [k]$ and $j \in [\ell]$, we have

$$\frac{\partial \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x})}{\partial \mathbf{x}_{a_{i,j}}} = \frac{1}{\ell}\cdot\left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{a_{\sigma_j(i)}}}\right|_{\mathbf{z}=\mathbf{x}^{(\boldsymbol{\sigma})}} \qquad \text{and} \qquad \frac{\partial \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x})}{\partial \mathbf{x}_{b_{i,j}}} = \frac{1}{\ell}\cdot\left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{b_{\sigma_j(i)}}}\right|_{\mathbf{z}=\mathbf{x}^{(\boldsymbol{\sigma})}} \enspace . \tag{6}$$

Thus, the continuous differentiability of $\hat{F}_k$ implies that $\bar{F}_{k,\boldsymbol{\sigma}}$ is also continuously differentiable.

Taking the derivative of the last equalities with respect to $a_{i',b'}$ for another pair $i' \in [k], j' \in [\ell]$, the chain-rule gives us the equalities

$$\frac{\partial^2 \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x})}{\partial \mathbf{x}_{a_{i',j'}} \partial \mathbf{x}_{a_{i,j}}} = \frac{1}{\ell^2} \cdot \left. \frac{\partial^2 \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{a_{\sigma_{j'}(i')}} \partial \mathbf{z}_{a_{\sigma_j(i)}}} \right|_{\mathbf{z}=\mathbf{x}^{(\boldsymbol{\sigma})}}$$

and

$$\frac{\partial^2 \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x})}{\partial \mathbf{x}_{a_{i',j'}} \partial \mathbf{x}_{b_{i,j}}} = \frac{1}{\ell^2} \cdot \left. \frac{\partial^2 \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{a_{\sigma_{j'}(i')}} \partial \mathbf{z}_{b_{\sigma_j(i)}}} \right|_{\mathbf{z}=\mathbf{x}^{(\boldsymbol{\sigma})}} .$$

Since similar equalities hold also when we take the derivative of the equalities in Equation (6) with respect to $b_{i',j'}$, the DR-submodularity of $\hat{F}_k$ implies the same property for $\bar{F}_{k,\boldsymbol{\sigma}}$.

It remains to bound the smoothness of $\bar{F}_{k,\boldsymbol{\sigma}}$. For every two vectors $\mathbf{x}, \mathbf{y} \in [0,1]^{\mathcal{M}_k}$, we have by Equation (6) that

$$\begin{aligned}
\|\nabla \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) - \nabla \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{y})\|_2^2 &= \sum_{i=1}^{k}\sum_{j=1}^{\ell}\left(\frac{1}{\ell}\cdot\left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{a_{\sigma_j(i)}}}\right|_{\mathbf{z}=\mathbf{x}^{(\boldsymbol{\sigma})}} - \frac{1}{\ell}\cdot\left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{a_{\sigma_j(i)}}}\right|_{\mathbf{z}=\mathbf{y}^{(\boldsymbol{\sigma})}}\right)^2 \\
&+ \sum_{i=1}^{k}\sum_{j=1}^{\ell}\left(\frac{1}{\ell}\cdot\left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{b_{\sigma_j(i)}}}\right|_{\mathbf{z}=\mathbf{x}^{(\boldsymbol{\sigma})}} - \frac{1}{\ell}\cdot\left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{b_{\sigma_j(i)}}}\right|_{\mathbf{z}=\mathbf{y}^{(\boldsymbol{\sigma})}}\right)^2 = \frac{1}{\ell}\cdot\sum_{i=1}^{k}\left(\left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{a_i}}\right|_{\mathbf{z}=\mathbf{x}^{(\boldsymbol{\sigma})}} - \left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{a_i}}\right|_{\mathbf{z}=\mathbf{y}^{(\boldsymbol{\sigma})}}\right)^2 \\
&+ \frac{1}{\ell}\cdot\sum_{i=1}^{k}\left(\left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{b_i}}\right|_{\mathbf{z}=\mathbf{x}^{(\boldsymbol{\sigma})}} - \left.\frac{\partial \hat{F}_k(\mathbf{z})}{\partial \mathbf{z}_{b_i}}\right|_{\mathbf{z}=\mathbf{y}^{(\boldsymbol{\sigma})}}\right)^2 = \frac{\|\nabla \hat{F}_k(\mathbf{x}^{(\boldsymbol{\sigma})}) - \nabla \hat{F}_k(\mathbf{y}^{(\boldsymbol{\sigma})})\|_2^2}{\ell} \leq \frac{\beta^2\|\mathbf{x}^{(\boldsymbol{\sigma})} - \mathbf{y}^{(\boldsymbol{\sigma})}\|_2^2}{\ell} \\
&= \frac{\beta^2 \cdot \sum_{i=1}^{k}[(\sum_{j=1}^{\ell}\mathbf{x}_{a_{\sigma_j(i),j}} - \sum_{j=1}^{\ell}\mathbf{y}_{a_{\sigma_j(i),j}})^2 + (\sum_{j=1}^{\ell}\mathbf{x}_{b_{\sigma_j(i),j}} - \sum_{j=1}^{\ell}\mathbf{y}_{b_{\sigma_j(i),j}})^2]}{\ell^3} ,
\end{aligned}$$

where β is the smoothness parameter of $\hat{F}_k$, and the second equality holds since the entries of $\boldsymbol{\sigma}$ are permutations. Using Sedrakyan's inequality (or Cauchy–Schwarz inequality), we also have, for every $i \in [k]$,

$$\left(\sum_{j=1}^{\ell}\mathbf{x}_{a_{\sigma_j(i),j}} - \sum_{j=1}^{\ell}\mathbf{y}_{a_{\sigma_j(i),j}}\right)^2 \leq \ell \cdot \sum_{j=1}^{\ell}(\mathbf{x}_{a_{\sigma_j(i),j}} - \sum_{j=1}^{\ell}\mathbf{y}_{a_{\sigma_j(i),j}})^2$$

and

$$\left(\sum_{j=1}^{\ell}\mathbf{x}_{b_{\sigma_j(i),j}} - \sum_{j=1}^{\ell}\mathbf{y}_{b_{\sigma_j(i),j}}\right)^2 \leq \ell \cdot \sum_{j=1}^{\ell}(\mathbf{x}_{b_{\sigma_j(i),j}} - \sum_{j=1}^{\ell}\mathbf{y}_{b_{\sigma_j(i),j}})^2 .$$

Combining all the above inequalities yields

$$\begin{aligned}
\|\nabla \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{x}) - \nabla \bar{F}_{k,\boldsymbol{\sigma}}(\mathbf{y})\|_2 &\leq \frac{\beta \cdot \sqrt{\sum_{i=1}^{k}[\sum_{j=1}^{\ell}(\mathbf{x}_{a_{\sigma_j(i),j}} - \mathbf{y}_{a_{\sigma_j(i),j}})^2 + \sum_{j=1}^{\ell}(\mathbf{x}_{b_{\sigma_j(i),j}} - \mathbf{y}_{b_{\sigma_j(i),j}})^2]}}{\ell} \\
&= \frac{\beta \cdot \|\mathbf{x} - \mathbf{y}\|_2}{\ell} ,
\end{aligned}$$

which completes the proof of the lemma since the smoothness parameter β of $\hat{F}_k$ is polynomial in k. □

C.1.3 Proof of Claim C.9

In this section we prove Claim C.9, which we repeat here for convenience.

Claim C.9. *For every vector* $\mathbf{x} \in \mathcal{K}_{h,k,\ell}$, $\mathbf{x}^{(\boldsymbol{\sigma})} \in \mathcal{P}_{h,k}$.

Proof. By the definition of $\mathcal{K}_{h,k,\ell}$, the membership of $\mathbf{x}$ in $\mathcal{K}_{h,k,\ell}$ implies that for every $j \in [\ell]$ we must have $\mathbf{x}^{(j)} \in \mathcal{P}_{h,k}$. Thus, $\mathbf{x}^{(j)}$ can be represented by a convex combination of the vectors $\mathbf{u}, \mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \dots, \mathbf{v}^{(k)}$ as follows.

$$\mathbf{x}^{(j)} = \sum_{i=1}^{k} c_{i,j} \cdot \mathbf{v}^{(j)} + d_j \cdot \mathbf{u} \enspace .$$

Similarly to the proof of Lemma C.8, let us define $\sigma_j^{-1}(\mathbf{x}^{(j)})$ to be the following vector. For every $i \in [k]$,

$$(\sigma_j^{-1}(\mathbf{x}^{(j)}))_{a_i} = \mathbf{x}^{(j)}_{a_{\sigma(i)}} \qquad \text{and} \qquad (\sigma_j^{-1}(\mathbf{x}^{(j)}))_{b_i} = \mathbf{x}^{(j)}_{b_{\sigma(i)}} \enspace .$$

Using the above notation, we get

$$\begin{aligned}
\mathbf{x}^{(\boldsymbol{\sigma})} &= \tfrac{1}{\ell}\sum_{j=1}^{\ell} \sigma_j^{-1}(\mathbf{x}^{(j)}) = \tfrac{1}{\ell}\sum_{j=1}^{\ell} \sigma_j^{-1}\left(\sum_{i=1}^{k} c_{i,j} \cdot \mathbf{v}^{(i)} + d_j \cdot \mathbf{u}\right) \\
&= \tfrac{1}{\ell}\sum_{j=1}^{\ell}\left[\sum_{i=1}^{k} c_{i,j} \cdot \sigma_j^{-1}(\mathbf{v}^{(i)}) + d_j \cdot \sigma_j^{-1}(\mathbf{u})\right] = \tfrac{1}{\ell}\sum_{j=1}^{\ell}\left[\sum_{i=1}^{k} c_{i,j} \cdot \mathbf{v}^{(\sigma_j^{-1}(i))} + d_j \cdot \mathbf{u}\right] \\
&= \sum_{i=1}^{k} \frac{\sum_{j=1}^{\ell} c_{\sigma_j(i),j}}{\ell} \cdot \mathbf{v}^{(i)} + \frac{\sum_{j=1}^{\ell} d_j}{\ell} \cdot \mathbf{u} \enspace .
\end{aligned}$$

The last step in the proof of the claim is to show that the rightmost side is a convex combination, which implies $\mathbf{x}^{(\boldsymbol{\sigma})} \in \mathcal{P}_{h,k}$ by the definition of $\mathcal{P}_{h,k}$. To see that this is indeed the case, we observe that the coefficients of all the vectors in this rightmost side are averages of non-negative numbers, and therefore, are non-negative as well. Furthermore,

$$\sum_{i=1}^{k} \frac{\sum_{j=1}^{\ell} c_{\sigma_j(i),j}}{\ell} + \frac{\sum_{j=1}^{\ell} d_j}{\ell} = \tfrac{1}{\ell}\sum_{j=1}^{\ell}\left[\sum_{i=1}^{k} c_{\sigma_j(i),j} + d_j\right] = \tfrac{1}{\ell}\sum_{j=1}^{\ell}\left[\sum_{i=1}^{k} c_{i,j} + d_j\right] = \tfrac{1}{\ell}\sum_{j=1}^{\ell} 1 = 1 \enspace ,$$

where the second equality holds since σ_j is a permutation for every $j \in \ell$. ☐

D QUADRATIC PROGRAMMING

In this section, we complement the study of (our version) of the offline algorithm of Du (2022), by checking its empirical performance for down-closed polytopes. Algorithms with better approximation guarantees are known when one is guaranteed to have such a constraint (Bian et al., 2017a). However, it is still important to understand the performance of algorithms designed for general polytope constraint when they happen to get a down-closed polytope. In particular, we note that Dürr et al. (2021) studied the empirical performance of their algorithm compared to the performance of the algorithm of Bian et al. (2017a) subject to such constraints, and we extend here their work by comparing the performance of their algorithm with that of newer algorithms. All the experiments presented in this section closely follow settings studied in (Dürr et al., 2021).

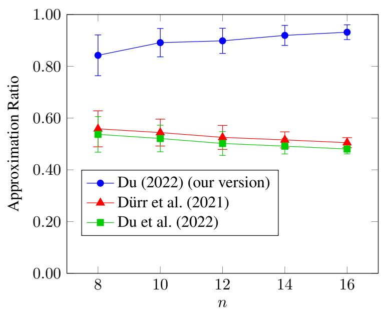
Consider the down-closed polytope given by

$$\mathcal{K} = \{\mathbf{x} \in \mathbb{R}^n_{\geq 0} \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \leq \mathbf{u}, \mathbf{A} \in \mathbb{R}^{m \times n}_{\geq 0}, \mathbf{b} \in \mathbb{R}^m_{\geq 0}\} \enspace ,$$

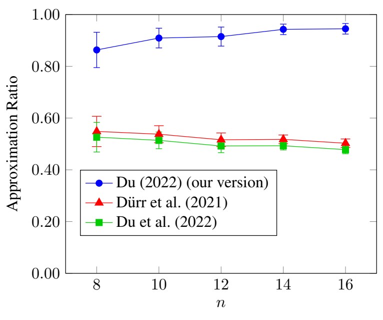
where $\mathbf{A}$ is a non-negative matrix chosen in a way described below, $\mathbf{b}$ is the all ones vector, and $\mathbf{u}$ is a vector that acts as an upper bound on $\mathcal{K}$ and is given by $u_j = \min_{j \in [m]} b_i / A_{i,j}$ for every $j \in [n]$. We now describe a function F that we would like to maximize subject to $\mathcal{K}$. For every vector $\bar{0} \leq \mathbf{x} \leq \mathbf{u}$ (where $\bar{0}$ is the all zeros vector),

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T \mathbf{H} \mathbf{x} + \mathbf{h}^T \mathbf{x} + c \enspace ,$$

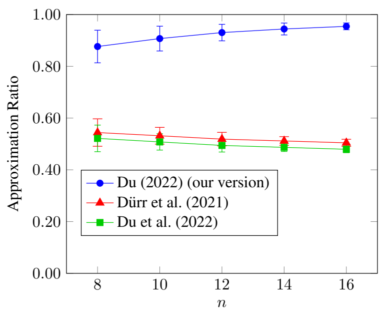
where $\mathbf{H}$ is a matrix, $\mathbf{h}$ is a vector and c is a scalar. The matrix $\mathbf{H}$ is chosen in a way described below, and it is always non-positive, which guarantees that F is DR-submodular. Furthermore, once $\mathbf{H}$ is chosen, we follow Bian et al. (2017a) and set $\mathbf{h} = -0.1 \cdot \mathbf{H}^T \mathbf{u}$. Finally, to make sure that F is also non-negative, the value of c should be at least $M =$



(a) $m = \lfloor 0.5n \rfloor$



(b) $m = n$



(c) $m = \lfloor 1.5n \rfloor$

Figure 3: Quadratic Programming with Uniform Distribution

$-\min_{\bar{0}\leq\mathbf{x}\leq\mathbf{u}} \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x}$. The value of M can be approximately obtained using QUADPROGIP[6] (Xia et al., 2020), and c is chosen to be $M + 0.1|M|$, which is a bit larger than the necessary minimum.

It remains to describe the way in which the entries of the matrices $\mathbf{H}$ and $\mathbf{A}$ are chosen. Below we describe two different random ways in which this can be done, and study the performance of the various algorithms on the instances generated in this way.

D.1 Uniform distribution

The first way to choose the matrices $\mathbf{H}$ and $\mathbf{A}$ is using a uniform distribution. Here, the matrix $\mathbf{H} \in \mathbb{R}^{n\times n}$ is a randomly generated symmetric matrix whose entries are drawn uniformly at random (and independently) from $[-1, 0]$, and $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a randomly generated matrix whose entries are drawn uniformly at random from $[v, v + 1]$ for $v = 0.01$ (this choice of v guarantees that the entries of $\mathbf{A}$ are strictly positive).

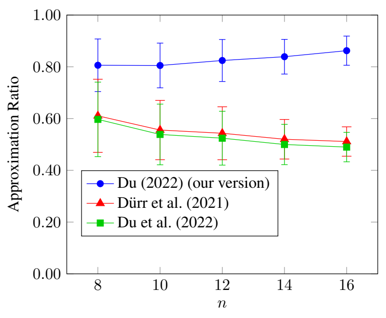
In each one of our experiments, we chose a different set of values for the dimensions n and m, and then drew an instance from the above distribution and executed on it 100 iterations of three algorithms: our explicit version from Section 3 of the algorithm of Du (2022) (with $\varepsilon = 0.03$), and the previous algorithms of Dürr et al. (2021) and Du et al. (2022). Each such experiment was repeated 100 times, and the results are depicted in Figure 3. In each plot of this figure, the x-axis represents the value of n, and the caption of the plot specifies how the value of m was calculated based on the value of n. The y-axis of the plots represents the approximation ratios obtained by the various algorithms compared to the optimum computed using a quadratic programming solver. One can observe that the two sub-exponential time algorithms of Dürr et al. (2021) and Du et al. (2022) exhibit similar performance, and (our version) of the newer algorithm of Du (2022) consistently and significantly outperforms them.

D.2 Exponential distribution

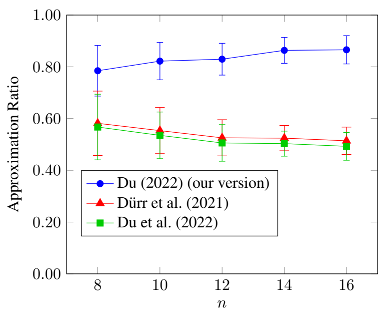
The other way to choose the matrices $\mathbf{H}$ and $\mathbf{A}$ is using an exponential distribution. Recall that given $\lambda > 0$, the exponential distribution $\exp(\lambda)$ is given by a density function assigning a density of $\lambda e^{-\lambda y}$ for every $y \geq 0$ and density 0 for negative y values. Then, $\mathbf{H} \in \mathbb{R}^{n\times n}$ is randomly generated symmetric matrix whose entries are drawn independently from $-\exp(1)$, and $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a randomly generated matrix whose entries are drawn independently from $\exp(0.25) + 0.01$.

For this way of generating $\mathbf{H}$ and $\mathbf{A}$, we repeated that same set of experiments as for the previous way of generating these matrices. The results of these experiments (averaged over 100 repetitions) are depicted in Figure 4. Again, we note that the two sub-exponential time algorithms of Dürr et al. (2021) and Du et al. (2022) exhibit similar performance, and (our version) of the newer algorithm of Du (2022) significantly outperforms them, especially as the dimension n grows.

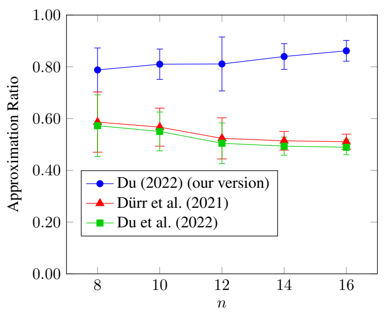
[6]We have used IBM CPLEX optimization studio `https://www.ibm.com/products/ilog-cplex-optimization-studio`.



(a) $m = \lfloor 0.5n \rfloor$



(b) $m = n$



(c) $m = \lfloor 1.5n \rfloor$

Figure 4: Quadratic Programming with Exponential Distribution

Statement of Authorship

Principal Author

Title of Paper	Bridging the Gap Between General and Down-Closed Convex Sets in Submodular Maximizaiton		
Publication Status	✓ Published; ☐ Accepted for Publication; ☐ Submitted for Publication		
Publication Details	In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 1926-1934. 2024.		
Name of Principal Author (Candidate)	Loay Mualem		
Contribution to the Paper	Contributed to the theory, implementation and writing of the paper.		
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Name and Signature	Loay Mualem	Date	24/6/2025

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the dissertation

Name of Co-Author	Morad Tukan		
Contribution to the Paper	Contributed to the writing and implementation of the paper.		
Name and Signature	Morad Tukan [illegible]	Date	25/06/2025

Name of Co-Author	Moran Feldman		
Contribution to the Paper	Contributed to the theory and writing of the paper.		
Name and Signature	Moran Feldman	Date	25/06/2025

Please cut and paste additional co-author panels here as required.

Chapter 5

Bridging the Gap Between General and Down-Closed Convex Sets in Submodular Maximization

In this chapter, we study the impact of constraint structure on the approximability of submodular maximization. Specifically, in the previous literature, there was a gap between general convex sets and down-closed convex sets. To bridge this gap, we present new hardness results alongside improved approximation algorithms that adapt to the geometric properties of the constraint set, and thus, provide a smooth interpolation between previously known results for general and down-closed convex sets.

The following paper [MTF24] was published at the *International Joint Conference on Artificial Intelligence (IJCAI 2024)*.

Bridging the Gap Between General and Down-Closed Convex Sets in Submodular Maximization

Loay Mualem[1] , **Murad Tukan**[2] and **Moran Feldman**[1]
[1]University of Haifa
[2]DataHeroes
loaymua@gmail.com, murad@dataheroes.ai, moranfe@cs.haifa.ac.il

Abstract

Optimization of DR-submodular functions has experienced a notable surge in significance in recent times, marking a pivotal development within the domain of non-convex optimization. Motivated by real-world scenarios, some recent works have delved into the maximization of non-monotone DR-submodular functions over general (not necessarily down-closed) convex set constraints. Up to this point, these works have all used the minimum L-infinity norm of any feasible solution as a parameter. Unfortunately, a recent hardness result due to Mualem and Feldman shows that this approach cannot yield a smooth interpolation between down-closed and non-down-closed constraints. In this work, we suggest novel offline and online algorithms that provably provide such an interpolation based on a natural decomposition of the convex body constraint into two distinct convex bodies: a down-closed convex body and a general convex body. We also empirically demonstrate the superiority of our proposed algorithms across three offline and two online applications.

1 Introduction

Optimization of continuous DR-submodular functions (and the strongly related discrete submodular set functions) has experienced a notable surge in significance in recent times, marking a pivotal development within the domain of non-convex optimization. The tools developed in this context adaptively tackle challenges related to real-world applications at the forefront of various fields such as data summarization [Mualem *et al.*, 2023; Hassani *et al.*, 2017; Bian *et al.*, 2019; Mitra *et al.*, 2021; Soma and Yoshida, 2017], robotics [Shi *et al.*, 2021; Tukan *et al.*, 2023], and human brain mapping [Salehi *et al.*, 2017], among many others. Most of the works in the literature focused either on DR-submodular optimization for monotone objective functions, or subject to a down-closed convex set constraint.[1] However, real-world problems are often naturally captured as optimization of a non-monotone DR-submodular function over a constraint convex set that is not down-closed. For example, imagine an online shopping platform optimizing its product recommendations within strict interface size constraints. The challenge faced by the store involves designing concise summaries that respect upper and lower bounds on product inclusion (these bounds are imposed by the user interface, and they form a non-down-closed constraint), and are good with respect to an objective function balancing between diversity and relevance in the displayed recommendations (which naturally yields a non-monotone objective).

Motivated by scenarios similar to the one given above, and optimization under fairness constraints [El Halabi *et al.*, 2023; Halabi *et al.*, 2023; Yuan and Tang, 2023], some recent works have delved into the maximization of non-monotone DR-submodular functions over general (not necessarily down-closed) convex set constraints. Unfortunately, in general, no constant approximation can be obtained for this problem in sub-exponential time due to a hardness result by Vondrák [2013]. However, Durr et al. [2021] observed that Vondrák's proof of the hardness result was based on a class of instances whose convex sets $\mathcal{K}$ include no point whose ℓ_∞-norm is less than 1.[2] In light of this observation, Durr et al. [2021] considered a parametrization of the problem based on the minimum ℓ_∞ norm of any vector in $\mathcal{K}$ (this minimum is usually denoted by m), and were able to provide a sub-exponential time $\frac{1}{3\sqrt{3}}(1-m)$-approximation for the problem of optimizing a non-monotone DR-submodular function subject to a general convex set $\mathcal{K}$.

The work Durr et al. [2021] has inspired a new line of research. Thắng & Srivastav [2021] showed how to obtain a similar result in an online (regret minimization) setting, and Du et al. [2022] improved the approximation ratio in the of-

[*]A full version of this paper, including all details and proofs omitted here, is available in [Mualem *et al.*, 2024].

[1]A set $\mathcal{K}$ is *down-closed* with respect to a domain $\mathcal{X}$ if, for every two vectors $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, $\mathbf{x} \in \mathcal{K}$ whenever $\mathbf{y} \in \mathcal{K}$ and $\mathbf{y}$ coordinate-wise dominates $\mathbf{x}$. As is standard, we usually omit the domain when talking about down-closed sets, and implicitly assume it to be the domain of the objective function.

[2]For simplicity, we implicitly assume that the domain of the objective function and the convex set constraint is $[0, 1]^n$. This assumption is without loss of generality (see Section 2).

fline setting to $\frac{1}{4}(1-m)$ (while still using sub-exponential time). More recently, the first polynomial time algorithm for the problem, guaranteeing the same approximation ratio as [Du *et al.*, 2022], has been provided by Du [2022]; and subsequently, Mualem & Feldman [2023] obtained the same result in the online setting. In their work, Mualem & Feldman also proved a hardness result, showing that $\frac{1}{4}(1-m)$-approximation is the best approximation ratio that can be obtained in sub-exponential time, which shows that the last algorithms are optimal, and settles the approximability of the problem with respect to this parametrization.

Our contribution. Every down-closed convex body has $m = 0$. The reverse is not true, but one could hope that convex bodies having $m = 0$ admit as good approximation as down-closed convex bodies. Unfortunately, the hardness result of Mualem & Feldman disproves this hope since the state-of-the-art approximation ratio for down-closed bodies is 0.401 [Buchbinder and Feldman, 2023]. This prompts the question of whether there is a different way to look at the problem (beyond parametrization by m) that will provide a smooth interpolation between the approximability obtainable for down-closed and general convex bodies. In this work, we suggest such an interpolation based on a decomposition of the convex body constraint into two distinct convex bodies: a down-closed convex body and a general convex body. Our key results based on this decomposition are as follows.

- We provide a novel polynomial time (offline) algorithm for maximizing DR-submodular functions over convex sets given as a composition of a down-closed convex body $\mathcal{K}_{\mathrm{D}}$ and a general convex body $\mathcal{K}_{\mathrm{N}}$. Our algorithm always recovers at least the $\frac{1}{4}(1-m)$-approximation of Du [2022]. The approximation guarantee smoothly improves when a significant fraction of the value of the optimal solution belongs to the down-closed part $\mathcal{K}_{\mathrm{D}}$ of the decomposition. In particular, when the convex body happens to be entirely down-closed, our algorithm guarantees $e^{-1} \approx 0.367$-approximation, which recovers the approximation ratio for down-closed convex bodies obtained by the Measured Continuous Greedy technique [Bian *et al.*, 2017a; Feldman *et al.*, 2011].[3]
- We provide a novel online (regret minimization) algorithm for the same problem that replicates the guarantees of our offline algorithm, up to a regret term that is proportional to the square root of the number of time steps.
- We demonstrate that a decomposition of the kind that we use can be naturally obtained for various machine-learning applications, and use this fact to empirically demonstrate the superiority of our proposed algorithms compared to existing methods across various offline and online applications, namely, offline and online revenue maximization, quadratic programming and location summarization. In the full version of this paper [Mualem *et al.*, 2024], we also provide theoretical implications for fairness settings.

[3]As mentioned above, the state-of-the-art approximation ratio for down-closed convex body constraints is 0.401. However, all current algorithms for such convex bodies with ratios better than e^{-1} are based on the Measured Continuous Greedy technique (with additional, often impractical, components). To keep our algorithms simple and our message clear, we only aim to recover e^{-1}-approximation for down-closed convex bodies.

1.1 Related Work

DR-submodular maximization has recently emerged as a key tool in numerous applications within the realms of machine learning and statistics. This surge in relevance has prompted a growing number of studies in this field. In what follows, we provide a brief overview of the main results in this field.

Offline DR-submodular optimization. The study of maximization of DR-submodular functions over down-closed convex sets was initiated by Bian et al. [2017a]. Their work showcased the effectiveness of a modified Frank-Wolfe algorithm, grounded in the greedy method introduced by [Calinescu *et al.*, 2011] in the context of set functions, ensuring $(1 - 1/e)$-approximation for the problem when the objective function is guaranteed to be monotone (this is optimal due to [Nemhauser and Wolsey, 1978]). Hassani et al. [2017] subsequently identified a limitation in the algorithm proposed by [Bian *et al.*, 2017a], revealing its lack of robustness in stochastic settings where only an unbiased estimator of the gradient is available. To mitigate this limitation, Hassani et al. [2017] demonstrated that gradient methods exhibit robustness in such scenarios, achieving $1/2$-approximation.

When the DR-submodular function is not guaranteed to be monotone, the approximation task becomes notably more challenging. In separate works, Bian et al. [2019] and Niazadeh et al. [2020] introduced distinct algorithms, both ensuring $1/2$-approximation for the maximization of non-monotone DR-submodular functions over a hypercube constraint, which is the best possible [Feige *et al.*, 2011]. It is noteworthy that (one version of) the algorithm proposed by [Niazadeh *et al.*, 2020] applies also to (non-DR) submodular functions. As mentioned above, the state-of-the-art approximation for maximizing non-monotone DR-submdoular functions subject to a down-closed convex body constraint is 0.401 [Buchbinder and Feldman, 2023].

Recently, Pedramfar et al. [2023] introduced a unified approach for maximization of continuous DR-submodular functions that encompasses a range of settings and oracle access types, while Mualem & Feldman [2022] suggested a parameter termed *monotonicity-ratio* allowing for a smooth interpolation between e^{-1}-approximation for non-monotone objectives and $(1 - 1/e)$-approximation for monotone objectives.

Online (regret minimization) DR-submodular optimization. Chen et al. [2018] were the first to address online maximization of monotone DR-submodular functions over a convex set (for monotone objective functions the distinction between down-closed and general convex sets is irrelevant). They introduced two algorithms: one ensuring $(1 - 1/e)$-approximation with approximately $O(\sqrt{T})$-regret, and another algorithm that is resilient to stochastic settings, but guarantees only $1/2$-approximation up to the same regret. Later, Chen et al. [2019] proposed an algorithm that combines $(1 - 1/e)$-approximation with roughly $O(\sqrt{T})$-regret and robustness, and Zhang et al. [2019] demonstrated how to reduce the number of gradient calculations per time step to one at the expense of increasing the regret to roughly $O(T^{4/5})$. The last reduction is particularly relevant for bandit versions of the problem (we refer the reader to [Pedramfar *et al.*, 2023] for a detailed overview of such versions).

For online optimization of DR-submodular functions that are not guaranteed to be monotone, Thắng & Srivastav [2021] introduced three algorithms. One of these algorithms is applicable to general convex set constraints, and was later improved over by Mualem and Feldman [2023], as discussed above. Another was designed for maximization over the entire hypercube, achieving $1/2$-approximation with approximately $O(\sqrt{T})$-regret. The last algorithm of [Thắng and Srivastav, 2021] addresses down-closed convex set constraints, and attains e^{-1}-approximation with roughly $O(T^{2/3})$-regret.

1.2 Paper Organization

In Section 2, we formally describe the problem we consider. Then, in Section 3, we discuss the technique underlying our results. Our offline and online algorithms, which are based on this technique, can be found in Sections 4 and 5, respectively. Finally, Section 6 compares the empirical performance of our algorithms on multiple machine learning applications with the performance of previously suggested algorithms from the literature.

2 Preliminaries

In this section, we formally present the problem we consider in this paper and the notation that we use. Let us begin with the definition of DR-submodular functions, which are continuous analogs of submodular set functions first defined by Bian et al. [2017b]. Formally, given a domain $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_i$, where $\mathcal{X}_i$ is a closed range in $\mathbb{R}$ for every $i \in [n]$, a function $F\colon \mathcal{X} \to \mathbb{R}$ is called *DR-submodular* if the inequality

$$F(\mathbf{a} + k\mathbf{e}_i) - F(\mathbf{a}) \geq F(\mathbf{b} + k\mathbf{e}_i) - F(\mathbf{b})$$

holds for every two vectors $\mathbf{a}, \mathbf{b} \in \mathcal{X}$, positive value k and coordinate $i \in [n]$ obeying $\mathbf{a} \leq \mathbf{b}$ and $\mathbf{b} + k\mathbf{e}_i \in \mathcal{X}$ (here and throughout the paper, $\mathbf{e}_i$ denotes the standard i-th basis vector, and comparison between two vectors should be understood to hold coordinate-wise). Bian et al. [2017b] observed that for continuously differentiable functions F, the above definition of DR-submodulrity is equivalent to

$$\nabla F(\mathbf{x}) \leq \nabla F(\mathbf{y}) \quad \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{X}, \mathbf{x} \geq \mathbf{y} \ ,$$

and for twice differentiable functions F, it is equivalent to the Hessian being non-positive at every vector $\mathbf{x} \in \mathcal{X}$.

In this work, we study the problem of maximizing a non-negative DR-submodular function $F\colon \mathcal{X} \to \mathbb{R}_{\geq 0}$ subject to a convex body $\mathcal{K} \subseteq \mathcal{X}$ constraint. We are interested in the approximation guarantee that can be obtained for this problem based on a particular decomposition of $\mathcal{K}$ into two other convex bodies: a convex body $\mathcal{K}_\mathrm{D}$ that is down-closed with respect to $\mathcal{X}$ and a (not necessary down-closed) convex body $\mathcal{K}_\mathrm{N}$. Formally, by saying that $\mathcal{K}_\mathrm{D}$ and $\mathcal{K}_\mathrm{N}$ are a *decomposition* of $\mathcal{K}$, we mean that $\mathcal{K} = (\mathcal{K}_\mathrm{N} + \mathcal{K}_\mathrm{D}) \cap \mathcal{X}$, where $\mathcal{K}_\mathrm{N} + \mathcal{K}_\mathrm{D} \triangleq \{\mathbf{y} + \mathbf{z} \mid \mathbf{y} \in \mathcal{K}_\mathrm{N}, \mathbf{z} \in \mathcal{K}_\mathrm{D}\}$.

For simplicity, we assume (throughout the paper) that the domain $\mathcal{X}$ of our objective functions is $[0,1]^n$. This assumption is without loss of generality since there is a natural linear mapping from $\mathcal{X}$ to $[0,1]^n$ preserving the above discussed properties of F and $\mathcal{K}$. Additionally, as is standard in the field, we assume that F is β-smooth for some $\beta > 0$. A function $F\colon [0,1]^n \to \mathbb{R}$ is call *β-smooth* if it is continuously differentiable, and obeys

$$\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\|_2 \leq \beta \|\mathbf{x} - \mathbf{y}\|_2 \quad \forall\, \mathbf{x}, \mathbf{y} \in [0,1]^n \ .$$

Another standard assumption in the field is that the relevant convex-bodies ($\mathcal{K}_\mathrm{N}$ and $\mathcal{K}_\mathrm{D}$ in our case) are solvable, i.e., that one can efficiently optimize linear functions over them. We take a step further, and assume the ability to optimize linear functions over any convex body defined by the intersection of a polynomial number of linear constraints and constraints requiring particular vectors to belong either to $\mathcal{K}_\mathrm{N}$ or to $\mathcal{K}_\mathrm{D}$. This assumption appeared (often implicitly) in many previous works (see, for example, [Buchbinder and Feldman, 2023; Ene and Nguyen, 2016; Mualem and Feldman, 2023]), and is theoretically justified by the well-known equivalence between separability and solvability.

We often refer below to the *diameter* D of $\mathcal{K}$. This diameter is defined as $D \triangleq \max_{\mathbf{x},\mathbf{y} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|_2$.

2.1 Additional Vector Operations

Following Buchbinder and Feldman [2023], we use the following coordinate-wise vector operations. To reduce the number of parentheses necessary, we assume that both these operations have a higher precedence compared to vector addition and subtraction.

Definition 2.1. *Given two vectors $\mathbf{x}, \mathbf{y} \in [0,1]^n$,*

- *we denote by $\mathbf{x} \odot \mathbf{y}$ their coordinate-wise multiplication (also known as the Hadamard product).*
- *we denote by $\mathbf{x} \oplus \mathbf{y}$ their coordinate-wise probabilistic sum. In other words, for every $i \in [n]$, $(\mathbf{x} \oplus \mathbf{y})_i \triangleq x_i + y_i - x_i y_i = 1 - (1 - x_i)(1 - y_i)$.*

As was noted by [Buchbinder and Feldman, 2023], the operation $\oplus$ is symmetric and associative. We also use $\bar{0}$ and $\bar{1}$ to represent the all-zeros and all-ones vectors, respectively.

2.2 Online Optimization

In the online (regret minimization) version of the problem we consider in this work, there are L time steps.[4] In every time step $\ell \in [L]$, the adversary selects a non-negative β-smooth DR-submodular function F_t, and then the algorithm should select a distribution P_ℓ of points in $\mathcal{K} = (\mathcal{K}_\mathrm{N} + \mathcal{K}_\mathrm{D}) \cap [0,1]^n$ without knowing F_t (the function F_t is revealed to the algorithm only after P_ℓ is selected). The objective of the algorithm is to maximize $\sum_{\ell=1}^{L} \mathbb{E}_{\mathbf{x} \sim P_\ell}[F_\ell(\mathbf{x})]$, and its success in doing so is measured compared to the best fixed solution (i.e., any two vectors $\mathbf{o}_\mathrm{N} \in \mathcal{K}_\mathrm{N}$ and $\mathbf{o}_\mathrm{D} \in \mathcal{K}_\mathrm{D}$ such that $\mathbf{o}_\mathrm{N} + \mathbf{o}_\mathrm{D} \in [0,1]^n$).

[4]The number of time steps is usually denoted by T in the literature. However, we use L in this paper to avoid confusion with the parameter T traditionally used by continuous submodular maximization algorithms (including our own algorithms).

Let us elaborate a bit on the last point. If the functions $F_1, F_2, \dots, F_L$ were all known upfront, one could execute the offline algorithm we develop to get a set of solutions $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(L)}$ such that $\sum_{\ell=1}^{L} F_\ell(\mathbf{x}^{(\ell)}) \geq \psi(\sum_{\ell=1}^{L} F_\ell(\mathbf{o}_\mathrm{N} + \mathbf{o}_\mathrm{D}), \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_\mathrm{D}))$ for some function ψ (the structure of the function ψ is determined by the guarantee of Theorem 4.1 below). Since an online algorithm has to select the output distribution P_ℓ before seeing the function F_ℓ, it can only guarantee

$$\sum_{\ell=1}^{L} \mathbb{E}_{\mathbf{x}\sim P_\ell}[F_\ell(\mathbf{x})] \geq \psi\Big(\sum_{\ell=1}^{L} F_\ell(\mathbf{o}_\mathrm{N}+\mathbf{o}_\mathrm{D}), \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_\mathrm{D})\Big) - \mathcal{R}(L)$$

for some regret function $\mathcal{R}(L)$. Asymptotically, for our online algorithm, $\mathcal{R}(L)$ grows proportionally to $\sqrt{L}$, and therefore, for large L values, the average guarantee of our online algorithm per function F_ℓ approaches the one of our offline algorithm. As usual for online settings, we assume that the range of the functions $F_1, F_2, \dots, F_L$ is $[0, 1]$.

3 Our Technique

Our algorithms maintain vectors $\mathbf{y} \in \mathcal{K}_\mathrm{N}$ and $\mathbf{z} \in \mathcal{K}_\mathrm{D}$. Intuitively, the vector $\mathbf{y}$ is maintained by the Frank-Wolfe variant developed by Mualem and Feldman [2023] for non-down-closed polytopes, and the vector $\mathbf{z}$ is maintained by the continuous-greedy-like variant of Frank-Wolfe developed by Bian [2017a] for down-closed polytopes. Combining the two algorithms requires us to solve some technical issues. For example, it is necessary to run the two algorithms in parallel since they both depend on the coordinates of the solution growing at a bounded rate, and it is necessary to create a correlation between the algorithms to guarantee that $\mathbf{y} + \mathbf{z}$ remains within $[0, 1]^n$. However, it turns out that the more interesting question is regarding the best way to combine the two vectors $\mathbf{y}$ and $\mathbf{z}$ into the output solution of the algorithm.

The most natural approach is to consider the sum $\mathbf{y} + \mathbf{z}$ as the output solution. Unfortunately, this does not work well since it results in coordinates of the solution growing too fast. To make this more concrete, we note that our algorithms, as well as the algorithms of [Bian *et al.*, 2017a] and [Mualem and Feldman, 2023], simulate continuous algorithms working from time $t = 0$ until time $t = 1$. Consider now a particular coordinate $j \in [n]$. Up until time $t \in [0, 1]$, our algorithms spend (up to) t units of "energy" on this coordinate. A fraction $x \in [0, t]$ of this "energy" is invested in growing y_j, and the remaining $t - x$ "energy" is invested in growing z_j. By the properties of the algorithms of [Bian *et al.*, 2017a] and [Mualem and Feldman, 2023], this investment of "energy" leads to $y_i = 1 - e^{-x}$ and $z_i = 1 - e^{x-t}$, which in the worst case can make $(\mathbf{y} + \mathbf{z})_i$ as large as $2(1 - e^{-t/2})$. To get a better upper bound on the coordinates of the solution, we have to use $\mathbf{y} \oplus \mathbf{z}$ as the output solution. Note that this choice guarantees that $(\mathbf{y} \oplus \mathbf{z})_j \leq 1 - [1 - (1 - e^{-x})] \cdot [1 - (1 - e^{x-t})] = 1 - e^{-t}$, which is always better (for $t > 0$) compared to the bound of $2(1 - e^{-t/2})$ obtained above.

While the use of $\mathbf{y} \oplus \mathbf{z}$ is useful, it does not come without a cost. As mentioned above, our algorithms simulate continuous algorithms, which is a common practice in the literature about submodular maximization. To discretize these algorithms, one has to split time into steps, and then do in each step a single modification of the vectors $\mathbf{y}$ and $\mathbf{z}$ simulating all the modifications done by the continuous algorithm throughout the step. The standard way in which this is done is as follows. Assume that, at the beginning of the step, the continuous algorithm increases $\mathbf{y}$ at a rate of $\mathbf{y}'$ and $\mathbf{z}$ at a rate of $\mathbf{z}'$, then the discrete algorithm should increase $\mathbf{y}$ by $\varepsilon\mathbf{y}'$ and $\mathbf{z}$ by $\varepsilon\mathbf{z}'$, where ε is the size of the step. Unfortunately, this standard practice results in $\mathbf{y} \oplus \mathbf{z}$ changing by $\varepsilon\mathbf{y}' \odot (\bar{1} - \mathbf{z}) + \varepsilon\mathbf{z}' \odot (\bar{1} - \mathbf{y}) - \varepsilon^2 \cdot (\mathbf{y}' \odot \mathbf{z}')$. To see why this is problematic, note that in the continuous algorithm, when $\mathbf{y}$ and $\mathbf{z}$ increase at rates of $\mathbf{y}'$ and $\mathbf{z}'$, respectively, $\mathbf{y} \oplus \mathbf{z}$ increases at a rate of $\mathbf{y}' \odot (\bar{1} - \mathbf{z}) + \mathbf{z}' \odot (\bar{1} - \mathbf{y})$. Thus, the term $-\varepsilon^2 \cdot (\mathbf{y}' \odot \mathbf{z}')$ from the previous expression represents a new kind of discretization error that we need to handle.

Another hurdle worth mentioning is that the vectors $\mathbf{y}$ and $\mathbf{z}$ are updated using two different update rules inherited from the algorithms of [Bian *et al.*, 2017a] and [Mualem and Feldman, 2023], and the interaction between these update rules results in a guarantee on the output of the algorithm that depends also on the value of $F(\mathbf{z})$. Thus, it is necessary to make sure that our algorithms maintain $\mathbf{z}$ in a way that also guarantees that $F(\mathbf{z})$ has a good value. In the first version of our offline algorithm, we do that by assuming that we know the value v of the part of the optimal solution that belongs to $\mathcal{K}_\mathrm{D}$. This knowledge allows us to force the algorithm to increase $\mathbf{z}$ in a way guaranteed to make $F(\mathbf{z})$ competitive with v. In the other versions of our offline algorithm and in our online algorithm, we use a potential function argument to avoid the need to know v. This potential function argument is similar to an argument used by Feldman [2021] in a different submodular maximization setting.

4 Offline Maximization

In this section, we present and analyze our offline algorithm, whose guarantee is given by the next theorem.

Theorem 4.1. *Let $\mathcal{K}_\mathrm{N} \subseteq [0, 1]^n$ be a general solvable convex set, $\mathcal{K}_\mathrm{D} \subseteq [0, 1]^n$ be a down-closed solvable convex set, and $F\colon [0, 1]^n \to \mathbb{R}_{\geq 0}$ be a non-negative β-smooth DR-submodular function. Then, there exists a polynomial time algorithm that, given an error parameter $\varepsilon \in (0, 1)$, outputs vectors $\mathbf{w} \in (\mathcal{K}_\mathrm{N} + \mathcal{K}_\mathrm{D}) \cap [0, 1]^n$ such that*

$$F(\mathbf{w}) \geq (1 - m) \cdot \max_{t_s\in[0,1]} \max_{T\in[t_s,1]} \Big\{((T - t_s)e^{-T} - O(\varepsilon)) \cdot F(\mathbf{o}_\mathrm{D}^{(2)}) + \Big(\frac{t_s^2 \cdot e^{-t_s-T}}{2} - O(\varepsilon)\Big) \cdot F(\mathbf{o}_\mathrm{D}^{(1)}) + (e^{-T} - e^{-t_s-T} - O(\varepsilon)) \cdot F(\mathbf{o}_\mathrm{N} + \mathbf{o}_\mathrm{D}^{(1)})\Big\} - O(\tfrac{\varepsilon\beta D^2}{1-m}) \enspace,$$

where $m = \min_{\mathbf{x}\in\mathcal{K}_\mathrm{N}} \|\mathbf{x}\|_\infty$, D is the diameter of $(\mathcal{K}_\mathrm{N} + \mathcal{K}_\mathrm{D}) \cap [0, 1]^n$, $\mathbf{o}_\mathrm{N} \in \mathcal{K}_\mathrm{N}$ and $\mathbf{o}_\mathrm{D}^{(1)} \in \mathcal{K}_\mathrm{D}$ are any vectors whose sum belongs to $(\mathcal{K}_\mathrm{N} + \mathcal{K}_\mathrm{D}) \cap [0, 1]^n$, and $\mathbf{o}_\mathrm{D}^{(2)}$ is any vector in $\mathcal{K}_\mathrm{D}$.[5]

[5]The vectors $\mathbf{o}_\mathrm{D}^{(1)}$ and $\mathbf{o}_\mathrm{D}^{(2)}$ can be identical.

It is interesting to note that Theorem 4.1 recovers two guarantees of previous works. Specifically, by setting $T = 1, t_s = 0$ and $\mathcal{K}_{\mathrm{N}} = \{\bar{0}\}$, the theorem implies e^{-1}-approximation for maximizing a DR-submodular function subject to a down-closed polytope $\mathcal{K}_{\mathrm{D}}$, recovering the result of [Bian *et al.*, 2017a]. Similarly, by setting $T = t_s = \ln 2$ and $\mathcal{K}_{\mathrm{D}} = \{\bar{0}\}$, Theorem 4.1 implies $1/4(1-m)$-approximation for maximizing a DR-submodular function subject to a general polytope $\mathcal{K}_{\mathrm{N}}$, recovering the result of [Mualem and Feldman, 2023].

For ease of the presentation, we have three versions of our offline algorithm. The first version, appearing below as Algorithm 1, proves Theorem 4.1 under the assumption that $F(\mathbf{o}_{\mathrm{D}}^{(1)})$ is known. In the full version of this paper [Mualem *et al.*, 2024], we present the two other versions of our offline algorithm that prove Theorem 4.1 without making this assumption. One version is theoretically natural, and is the base for our online algorithm described in Section 5. However, to get the best results in practice, it is natural to make some modifications to this natural version (including ones motivated by the work of [Bian *et al.*, 2017a]), which do not improve the theoretical guarantee of the algorithm and cannot be extended to the online version of the algorithm. Both the natural and the modified version of our algorithm are studied in the offline experiments described in Section 6.

Recall that Algorithm 1 proves Theorem 4.1 under the assumption that $F(\mathbf{o}_{\mathrm{D}}^{(1)})$ is known. In its description, we assume for simplicity that ε^{-1} is integral. If this is not the case, ε can be replaced with $\lceil \varepsilon^{-1} \rceil^{-1}$, which is smaller than ε by at most a factor of 2.

Algorithm 1 `Frank-Wolfe/Continuous-Greedy Hybrid for Known` $F(\mathbf{o}_{\mathrm{D}}^{(1)})$

```
1: Let $\mathbf{y}^{(0)} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{K}_{\mathrm{N}}} \|\mathbf{x}\|_\infty$ and $\mathbf{z}^{(0)} \leftarrow \bar{0}$.
2: for $i = 1$ to $\varepsilon^{-1}$ do
3:     Solve the following linear program:
           max   $\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}),$
                 $\mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \rangle$
           s.t.  $\mathbf{a}^{(i)} \in \mathcal{K}_{\mathrm{N}}, \mathbf{b}^{(i)} \in \mathcal{K}_{\mathrm{D}}$
                 $\langle \mathbf{b}^{(i)} \odot (\mathbf{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{z}^{(i-1)}) \rangle$
                     $\geq (1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)})$
                 $\mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \leq \bar{1}$
4:     Let $\mathbf{y}^{(i)} \leftarrow (1-\varepsilon) \cdot \mathbf{y}^{(i-1)} + \varepsilon \cdot \mathbf{a}^{(i)}$.
5:     Let $\mathbf{z}^{(i)} \leftarrow \mathbf{z}^{(i-1)} + \varepsilon \cdot (1 - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}$.
6: end for
7: Return a vector maximizing $F$ among all the vectors in
   $\{\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} \mid i \in \mathbb{Z}, 0 \leq i \leq \varepsilon^{-1}\}$.
```

5 Online Maximization

In this section, we consider the online (regret minimization) setting described in Section 2.2. The algorithm we present and analyze has the guarantee given by the next theorem.

Theorem 5.1. *Let $\mathcal{K}_{\mathrm{N}} \subseteq [0,1]^n$ be a general solvable convex set, and $\mathcal{K}_{\mathrm{D}} \subseteq [0,1]^n$ be a down-closed solvable convex set. If the adversary chooses only non-negative β-smooth DR-submodular functions $F_\ell \colon [0,1]^n \to [0,1]$, then there exists a polynomial time online algorithm that, given value $\varepsilon \in (0,1)$, outputs at every time step $\ell \in [L]$ a distribution P_ℓ over vectors in $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$ guaranteeing that*

$$\sum\nolimits_{\ell=1}^{L} \mathbb{E}_{\mathbf{x}\sim P_\ell}[F_\ell(\mathbf{x})] \geq (1-m) \cdot \max_{t_s\in[0,1]} \max_{T\in[t_s,1]} \Big\{ \Big((T-t_s)e^{-T} + \frac{t_s^2 \cdot e^{-t_s-T}}{2} - O(\varepsilon) \Big) \cdot \sum\nolimits_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) + (e^{-T} - e^{-t_s-T} - O(\varepsilon)) \cdot \sum\nolimits_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}) \Big\} - O(\varepsilon\beta L D^2) - O(DG\sqrt{L}) - O(\sqrt{L \ln \varepsilon^{-1}}) \enspace ,$$

where $m = \min_{\mathbf{x}\in\mathcal{K}_{\mathrm{N}}} \|\mathbf{x}\|_\infty$, D is the diameter of $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$, $G = \max\{\max_{\mathbf{x}\in(\mathcal{K}_{\mathrm{N}}+\mathcal{K}_{\mathrm{D}})\cap[0,1]^n} \|\nabla F_t(\mathbf{x})\|_2, \max_{\mathbf{x}\in\mathcal{K}_{\mathrm{D}}} \|\nabla F_t(\mathbf{x})\|_2,\}$, and $\mathbf{o}_{\mathrm{N}} \in \mathcal{K}_{\mathrm{N}}$ and $\mathbf{o}_{\mathrm{D}} \in \mathcal{K}_{\mathrm{D}}$ are any vectors whose sum belongs to $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$.

Note the following two remarks about Theorem 5.1.

- Section 2.2 states that the regret of our online algorithm (compared to the offline algorithm) asymptotically grows as $\sqrt{L}$. This might seem to contradict the presence of the error term $O(\varepsilon\beta L D^2)$ in the guarantee of Theorem 5.1. However, this is not the case since this error term is part of the ψ functions mentioned in Section 2.2 (because the term $O(\frac{\varepsilon\beta D^2}{1-m})$ appears in the guarantee of Theorem 4.1).
- Theorem 4.1 considers two vectors $\mathbf{o}_{\mathrm{D}}^{(1)}, \mathbf{o}_{\mathrm{D}}^{(2)} \in \mathcal{K}_{\mathrm{D}}$. A similar result could be proved in Theorem 5.1. However, in the online setting, the algorithm is required to be competitive against any fixed solution. Thus, we felt that it is more natural to state Theorem 5.1 for the case in which $\mathbf{o}_{\mathrm{D}}^{(1)}$ and $\mathbf{o}_{\mathrm{D}}^{(2)}$ are the same vector (denoted by $\mathbf{o}_{\mathrm{D}}$).

The central component in the proof of Theorem 5.1 is the following proposition, which is a variant of Theorem 5.1 in which (i) t_s is a parameter of the algorithm, and (ii) the algorithm outputs in each time step a single vector rather than a distribution over vectors.

Proposition 5.2. *Let $\mathcal{K}_{\mathrm{N}} \subseteq [0,1]^n$ be a general solvable convex set, and $\mathcal{K}_{\mathrm{D}} \subseteq [0,1]^n$ be a down-closed solvable convex set. If the adversary chooses only non-negative β-smooth DR-submodular functions $F_\ell \colon [0,1]^n \to [0,1]$, then there exists a polynomial time online algorithm that, given parameters $t_s \in [0,1]$ and $\varepsilon \in (0,1)$, outputs at every time step $\ell \in [L]$ a vector $\mathbf{w}^{(\ell)} \in (\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$ guaranteeing that*

$$\sum\nolimits_{\ell=1}^{L} F_\ell(\mathbf{w}^{(\ell)}) \geq (1-m) \cdot \max_{T\in[t_s,1]} \Big\{ \Big((T-t_s)e^{-T} + \frac{t_s^2 \cdot e^{-t_s-T}}{2} - O(\varepsilon) \Big) \cdot \sum\nolimits_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) + (e^{-T} - e^{-t_s-T} - O(\varepsilon)) \cdot \sum\nolimits_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}) \Big\} - O(\varepsilon\beta L D^2) - O(DG\sqrt{L}) \enspace ,$$

where $m = \min_{\mathbf{x}\in\mathcal{K}_{\mathrm{N}}} \|\mathbf{x}\|_\infty$, D is the diameter of $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$, $G = \max\{\max_{\mathbf{x}\in(\mathcal{K}_{\mathrm{N}}+\mathcal{K}_{\mathrm{D}})\cap[0,1]^n} \|\nabla F_t(\mathbf{x})\|_2,$

$\max_{\mathbf{x}\in\mathcal{K}_D} \|\nabla F_t(\mathbf{x})\|_2\}$*, and* $\mathbf{o}_N \in \mathcal{K}_N$ *and* $\mathbf{o}_D \in \mathcal{K}_D$ *are any vectors whose sum belongs to* $(\mathcal{K}_N + \mathcal{K}_D) \cap [0,1]^n$.

Algorithm 2 `Online Frank-Wolfe/Continuous-Greedy Hybrid`

```
1: for $i = 1$ to $\varepsilon^{-1}$ do
2:     Initialize an instance $\mathcal{E}_i$ of Regularized-Follow-the-Leader for the convex body $\{(\mathbf{a},\mathbf{b}) \mid \mathbf{a} \in \mathcal{K}_N, \mathbf{b} \in \mathcal{K}_D, \mathbf{a}+\mathbf{b} \in [0,1]^n\}$.
3: end for
4: Let $m \leftarrow \min_{\mathbf{x}\in\mathcal{K}_N}\|\mathbf{x}\|_\infty$.
5: for $\ell = 1$ to $L$ do
6:     Let $\mathbf{y}^{(0,\ell)} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{K}_N}\|\mathbf{x}\|_\infty$, $\mathbf{z}^{(0,\ell)} \leftarrow \bar{0}$.
7:     for $i = 1$ to $\varepsilon^{-1}t_s$ do
8:         Let $(\mathbf{a}^{(i,\ell)}, \mathbf{b}^{(i,\ell)})$ be the vector picked by $\mathcal{E}_i$ in time step $\ell$.
9:         Let $\mathbf{y}^{(i,\ell)} \leftarrow (1-\varepsilon)\cdot\mathbf{y}^{(i-1,\ell)} + \varepsilon\cdot\mathbf{a}^{(i,\ell)}$.
10:        Let $\mathbf{z}^{(i,\ell)} \leftarrow \mathbf{z}^{(i-1,\ell)} + \varepsilon\cdot(1-\mathbf{z}^{(i-1,\ell)})\odot\mathbf{b}^{(i,\ell)}$.
11:    end for
12:    for $i = \varepsilon^{-1}t_s + 1$ to $\varepsilon^{-1}$ do
13:        Let $\mathbf{a}^{(i,\ell)} \leftarrow \mathbf{y}^{(i-1,\ell)}$.
14:        Let $\mathbf{b}^{(i,\ell)}$ be a vector consisting of the last $n$ coordinates of the vector picked by $\mathcal{E}_i$ in time step $\ell$.
15:        Let $\mathbf{y}^{(i,\ell)} \leftarrow \mathbf{y}^{(i-1,\ell)}$.
16:        Let $\mathbf{z}^{(i,\ell)} \leftarrow \mathbf{z}^{(i-1,\ell)} + \varepsilon\cdot(1-\mathbf{z}^{(i-1,\ell)})\odot\mathbf{b}^{(i,\ell)}$.
17:    end for
18:    Set the vector $\mathbf{y}^{(\varepsilon^{-1},\ell)} \oplus \mathbf{z}^{(\varepsilon^{-1},\ell)}$ as the output for time step $\ell$.
19:    for $i = 1$ to $\varepsilon^{-1}t_s$ do
20:        Let $\mathbf{g}^{(i,\ell)}$ be the vector in $\mathbb{R}^{2n}$ obtained as follows. The first $n$ coordinates of this vector are given by $e^{2\varepsilon i}\nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)})$, and the other $n$ coordinates of $\mathbf{g}^{(i,\ell)}$ are equal to $e^{2\varepsilon i}\nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{y}^{(i-1,\ell)}) + (1-m)\cdot e^{\varepsilon i}(t_s - \varepsilon i)\cdot\nabla F_\ell(\mathbf{z}^{(i-1,\ell)}) \odot (1-\mathbf{z}^{(i-1,\ell)})$.
21:        Pass $\mathbf{g}^{(i,\ell)}$ as the adversarially chosen vector $\mathbf{d}^{(\ell)}$ for $\mathcal{E}_i$.
22:    end for
23:    for $i = \varepsilon^{-1}t_s + 1$ to $\varepsilon^{-1}$ do
24:        Let $\mathbf{g}^{(i,\ell)}$ be a vector in $\mathbb{R}^{2n}$ obtained as follows. The first $n$ coordinates of this vector are all zeros, and the other $n$ coordinates of $\mathbf{g}^{(i,\ell)}$ are equal to $\nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{y}^{(i-1,\ell)})$.
25:        Pass $\mathbf{g}^{(i,\ell)}$ as the adversarially chosen vector $\mathbf{d}^{(\ell)}$ for $\mathcal{E}_i$.
26:    end for
27: end for
```

Let us explain why Proposition 5.2 implies Theorem 5.1. The guarantee given in Theorem 5.1 remains unchanged (up to the constants hidden by the big O notation) if the maximum over $t_s \in [0,1]$ is restricted to the set of $O(\varepsilon^{-1})$ values that are integer multiples of ε between 0 and 1. If we knew upfront what value from this set leads to the best guarantee for Proposition 5.2, then we could use the algorithm of this proposition to get the guarantee of Theorem 5.1.

Unfortunately, in reality, we do not usually know upfront the best value for t_s. Nevertheless, since there are only $O(\varepsilon^{-1})$ such values that need to be considered, we can use a regret minimization algorithm (such as the one of [Cesa-Bianchi *et al.*, 2007]) to get in each time step a distribution over solutions whose expected value is at least as good as the guarantee of Proposition 5.2 for the best value of t_s, up to an error term of $O(\sqrt{L \ln \varepsilon^{-1}})$. Thus, Theorem 5.1 indeed follows from Proposition 5.2.

At this point, we would like to describe the algorithm that we use to prove Proposition 5.2, which is given as Algorithm 2. Similarly to the Meta-Frank-Wolfe algorithm of [Chen *et al.*, 2018], Algorithm 2 uses multiple instances of an online algorithm for linear optimization. Specifically, we use the algorithm Regularized-Follow-the-Leader due to [Abernethy *et al.*, 2008], which has the following behavior. There are L time steps. In every time step $\ell \in [L]$, the algorithm selects a vector $\mathbf{u}^{(\ell)} \in \mathcal{K}$ for some given convex body $\mathcal{K}$, and then an adversary reveals to the algorithm a vector $\mathbf{d}^{(\ell)}$ that was chosen independently of $\mathbf{u}^{(\ell)}$. Regularized-Follow-the-Leader guarantees that

$$\sum_{t=1}^{L} \langle \mathbf{u}^{(\ell)}, \mathbf{d}^{(\ell)} \rangle \geq \max_{\mathbf{x}\in\mathcal{K}} \sum_{\ell=1}^{L} \langle \mathbf{x}, \mathbf{d}^{(\ell)} \rangle - D'G'\sqrt{2L} \ ,$$

where $G' = \max_{1\leq\ell\leq L} \|\mathbf{d}^{(\ell)}\|_2$ and D' is the diameter of $\mathcal{K}$.

Algorithm 2 follows the same general structure of our offline algorithm, with two main modifications: the linear programs of the offline algorithm are replaced by instances of the online linear optimization algorithm that we use, and the vector corresponding to $i = T^{-1}$ is used as the output instead of the best vector among multiple options.

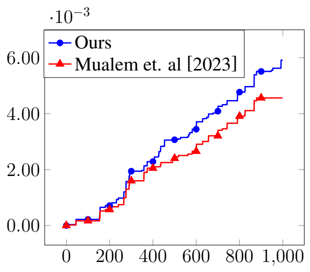
In the pseudocode of Algorithm 2, we implicitly assume that $\varepsilon \leq 1/70$ and εt_s is integral. The first assumption is without loss of generality since we can decrease ε to be $1/70$ if its original value is larger, and the second assumption is without loss of generality because the coefficients in the guarantee of Proposition 5.2 change only by $O(\varepsilon)$ when t_s is modified by up to ε to make $t_s\varepsilon$ integral.

6 Applications and Experimental Results

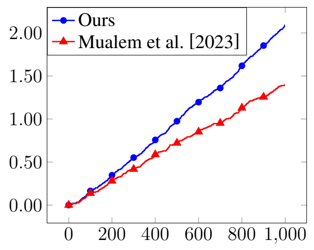
In this work, we study the empirical performance of the offline and online algorithms described in the previous sections on three machine learning tasks. We present one application here, and defer the rest to the full version of this paper [Mualem *et al.*, 2024]. The full version also includes empirical stability and ablation studies of our offline algorithm.

6.1 Revenue Maximization

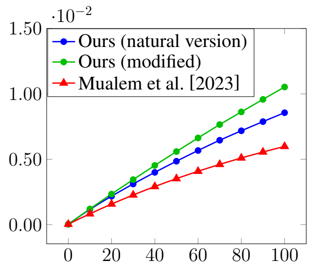
As our first experimental setting, we consider the following revenue maximization setting, which was also considered by [Mualem and Feldman, 2023; Soma and Yoshida, 2017; Thắng and Srivastav, 2021]. The objective of some company is to promote a product to users to boost revenue through the "word-of-mouth" effect. The problem of optimizing this objective can be formalized as follows. The input is a weighted undirected graph $G = (V, E)$ representing a social network, where w_{ij} represents the weight of the edge between vertex



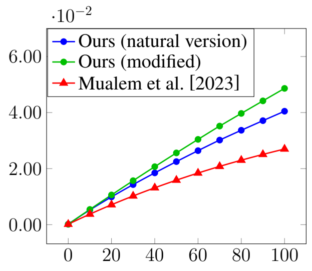
(a) Online Algorithms on the Advogato network.



(b) Online Algorithms on the Facebook network.



(c) Offline Algorithms on the Advogato network.



(d) Offline Algorithms on the Facebook network.

Figure 1: Results of the revenue maximization experiments. In each plot, the x-axis is the timestep, and the y-axis is the corresponding function value.

i and vertex j (with $w_{ij} = 0$ if the edge (i, j) is absent from the graph). If the company allocates a cost of x_i units to a user $i \in V$, then that user becomes an advocate of the product with a probability of $1 - (1 - p)^{x_i}$, where $p \in (0, 1)$ is a parameter. Note that this formula implies that each ε unit of cost invested in the user independently contributes to the chance of making the user an advocate. Furthermore, by investing a full unit in the user, the user becomes an advocate with a probability of p [Soma and Yoshida, 2017].

Given a set $S \subseteq V$ of users who have become advocates for the product, the expected revenue is related to the total influence of the users in S on non-advocate users, which is formally expressed as $\sum_{i \in S} \sum_{j \in V \setminus S} w_{ij}$. Hence, the objective function $F\colon [0, 1]^V \to \mathbb{R}_{\geq 0}$ in this setting is defined as the expectation of the aforementioned expression, i.e.,

$$F(\mathbf{x}) = \mathbb{E}_S\left[\sum_{i \in S} \sum_{j \in V \setminus S} w_{ij}\right] = \sum_{i \in V} \sum_{j \in V, i \neq j} w_{ij}(1 - (1 - p)^{x_i})(1 - p)^{x_j} \quad .$$

It has been demonstrated that F is a non-monotone DR-submodular function [Soma and Yoshida, 2017].

We conducted experiments in both online and offline scenarios based on instances of the aforementioned setting derived from two distinct datasets. The first dataset is sourced from a Facebook network [Viswanath *et al.*, 2009], encompassing $64K$ users (vertices) and $1M$ unweighted relationships (edges). The second dataset is based on the Advogato network [Massa *et al.*, 2009], comprising $6.5K$ users (vertices) and $61K$ weighted relationships (edges).

Online Setting. In our online experiments, inspired by [Mualem and Feldman, 2023], we set the number of time steps to $L = 1000$, with $p = 0.0001$. At each time step ℓ, the objective function is defined by selecting a uniformly random subset $V_\ell \subseteq V$ of a given size, and then retaining only edges connecting two vertices of V_ℓ. For the Advogato network, V_ℓ is of size 200, and for the larger Facebook network, V_ℓ is of size 15,000. The above objective functions are optimized subject to the constraint $0.1 \leq \sum_i x_i \leq 1$, which represents both minimum and maximum investment requirements. Notably, the intersection of this constraint with the implicit box constraint forms a non-down-monotone feasibility polytope. However, this polytope can be decomposed into two polytopes: (i) $\mathcal{K}_{\mathrm{N}}$, a polytope defined by the equality $\sum_{i=1}^n x_i = 0.1$, and (ii) $\mathcal{K}_{\mathrm{D}}$, a down-closed polytope defined by $\sum_{i=1}^n x_i \leq 0.9$. Observe that $(\mathcal{K}_{\mathrm{D}} + \mathcal{K}_{\mathrm{N}}) \cap [0, 1]^n$ is indeed the original polytope, and thus, this is a valid decomposition of this polytope.

In our experiments, we have compared the performance of our algorithm from Section 5 with the online algorithm of Mualem & Feldman [2023], which is the current state-of-the-art algorithm for the online setting. In both algorithms, we have set the number of online linear optimizers used to be 100 (which corresponding to setting the error parameter ε to 0.01 in our algorithm and to $\ln 2/100$ in the algorithm of Mualem & Feldman). The results of our experiments on the Advogato and Facebook networks can be found in Figures 1a and 1b, respectively. In both experiments, our algorithm significantly outperforms the state-of-the-art algorithm.

Offline Setting. Our offline experiments are similar to their online counterparts. However, since there is only one objective function in this setting, we run the experiment on the entire network graph. In this setting, we compared our offline algorithm (see Section 4 for a discussion of the versions of this algorithm used in the experiments) with the current state-of-the-art algorithm from [Mualem and Feldman, 2023] (which is an explicit version of the algorithm of Du [2022]). Both algorithms have been executed for $T = 100$ iterations, and the error parameter ε was set accordingly (which again means 0.01 in our algorithm and $\ln 2/100$ in the algorithm of Mualem & Feldman). The results of the offline experiments on the Advogato and Facebook networks can be found in Figures 1c and 1d, respectively. One can observe that our method consistently outperforms the previous state-of-the-art.

7 Conclusion

We have presented novel offline and online algorithms for DR-submodular maximization subject to a general convex body constraint. Our algorithms are able to provide a smooth interpolation between the approximability of general and down-closed convex bodies by considering a decomposition of the convex body constraint into a down-closed convex body and a general convex body. In addition to giving a theoretical analysis of our algorithms, we have demonstrated their empirical superiority (compared to state-of-the-art methods) in various online and offline machine learning applications.

Acknowledgements

The work of Loay Mualem and Moran Feldman has been partially supported by Israel Science Foundation (ISF) grant number 459/20.

References

[Abernethy *et al.*, 2008] Jacob Abernethy, Elad Hazan, and Alexander Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Conference on Learning Theory (COLT)*, pages 263–273, 2008.

[Bian *et al.*, 2017a] An Bian, Kfir Yehuda Levy, Andreas Krause, and Joachim M. Buhmann. Non-monotone continuous DR-submodular maximization: Structure and algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 486–496, 2017.

[Bian *et al.*, 2017b] Andrew An Bian, Baharan Mirzasoleiman, Joachim M. Buhmann, and Andreas Krause. Guaranteed non-convex optimization: Submodular maximization over continuous domains. In Aarti Singh and Xiaojin (Jerry) Zhu, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54 of *Proceedings of Machine Learning Research*, pages 111–120. PMLR, 2017.

[Bian *et al.*, 2019] Yatao Bian, Joachim Buhmann, and Andreas Krause. Optimal continuous DR-submodular maximization and applications to provable mean field inference. In *International Conference on Machine Learning (ICML)*, pages 644–653. PMLR, 2019.

[Buchbinder and Feldman, 2018] Niv Buchbinder and Moran Feldman. Submodular functions maximization problems. In Teofilo F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics*, volume 1, chapter 42. CRC Press, 2nd edition, 2018.

[Buchbinder and Feldman, 2023] Niv Buchbinder and Moran Feldman. Constrained submodular maximization via new bounds for dr-submodular functions. *CoRR*, abs/2311.01129, 2023.

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

[Cesa-Bianchi *et al.*, 2007] Nicolò Cesa-Bianchi, Yishay Mansour, and Gilles Stoltz. Improved second-order bounds for prediction with expert advice. *Mach. Learn.*, 66(2-3):321–352, 2007.

[Chekuri *et al.*, 2010] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *Foundations of Computer Science (FOCS)*, pages 575–584. IEEE Computer Society, 2010.

[Chen *et al.*, 2018] Lin Chen, Hamed Hassani, and Amin Karbasi. Online continuous submodular maximization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1896–1905. PMLR, 2018.

[Chen *et al.*, 2019] Lin Chen, Mingrui Zhang, and Amin Karbasi. Projection-free bandit convex optimization. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89 of *Proceedings of Machine Learning Research*, pages 2047–2056. PMLR, 2019.

[Du *et al.*, 2022] Donglei Du, Zhicheng Liu, Chenchen Wu, Dachuan Xu, and Yang Zhou. An improved approximation algorithm for maximizing a DR-submodular function over a convex set. *arXiv preprint arXiv:2203.14740*, 2022.

[Du, 2022] Donglei Du. Lyapunov function approach for approximation algorithm design and analysis: with applications in submodular maximization. *CoRR*, abs/2205.12442, 2022.

[Dürr *et al.*, 2021] Christoph Dürr, Nguyên Kim Thắng, Abhinav Srivastav, and Léo Tible. Non-monotone DR-submodular maximization over general convex sets. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2148–2154, 2021.

[El Halabi *et al.*, 2023] Marwa El Halabi, Federico Fusco, Ashkan Norouzi-Fard, Jakab Tardos, and Jakub Tarnawski. Fairness in streaming submodular maximization over a matroid constraint. In *International Conference on Machine Learning*, pages 9150–9171. PMLR, 2023.

[Ene and Nguyen, 2016] Alina Ene and Huy L. Nguyen. Constrained submodular maximization: Beyond 1/e. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 248–257. IEEE Computer Society, 2016.

[Feige *et al.*, 2011] Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing non-monotone submodular functions. *SIAM J. Comput.*, 40(4):1133–1153, 2011.

[Feldman *et al.*, 2011] Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *Foundations of Computer Science (FOCS)*, pages 570–579, 2011.

[Feldman, 2021] Moran Feldman. Guess free maximization of submodular and linear sums. *Algorithmica*, 83(3):853–878, 2021.

[Halabi *et al.*, 2023] Marwa El Halabi, Jakub Tarnawski, Ashkan Norouzi-Fard, and Thuy-Duong Vuong. Fairness in submodular maximization over a matroid constraint. *arXiv preprint arXiv:2312.14299*, 2023.

[Hassani *et al.*, 2017] Hamed Hassani, Mahdi Soltanolkotabi, and Amin Karbasi. Gradient methods for submodular maximization. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

[Kazemi *et al.*, 2021] Ehsan Kazemi, Shervin Minaee, Moran Feldman, and Amin Karbasi. Regularized submodular maximization at scale. In Marina Meila and Tong Zhang, editors, *International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 5356–5366. PMLR, 2021.

[Massa *et al.*, 2009] Paolo Massa, Martino Salvetti, and Danilo Tomasoni. Bowling alone and trust decline in social network sites. In *IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC)*, pages 658–663. IEEE Computer Society, 2009.

[Mitra *et al.*, 2021] Siddharth Mitra, Moran Feldman, and Amin Karbasi. Submodular + concave. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, pages 11577–11591, 2021.

[Mualem and Feldman, 2022] Loay Mualem and Moran Feldman. Using partial monotonicity in submodular maximization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[Mualem and Feldman, 2023] Loay Mualem and Moran Feldman. Resolving the approximability of offline and online non-monotone DR-submodular maximization over general convex sets. In Francisco J. R. Ruiz, Jennifer G. Dy, and Jan-Willem van de Meent, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 206 of *Proceedings of Machine Learning Research*, pages 2542–2564. PMLR, 2023.

[Mualem *et al.*, 2023] Loay Mualem, Ethan R Elenberg, Moran Feldman, and Amin Karbasi. Submodular minimax optimization: Finding effective sets. *arXiv preprint arXiv:2305.16903*, 2023.

[Mualem *et al.*, 2024] Loay Mualem, Murad Tukan, and Moran Fledman. Bridging the gap between general and down-closed convex sets in submodular maximization. *arXiv preprint arXiv:2401.09251*, 2024.

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

[Niazadeh *et al.*, 2020] Rad Niazadeh, Tim Roughgarden, and Joshua R Wang. Optimal algorithms for continuous non-monotone submodular and DR-submodular maximization. *Journal of Machine Learning Research*, 21(1):4937–4967, 2020.

[Pedramfar *et al.*, 2023] Mohammad Pedramfar, Christopher John Quinn, and Vaneet Aggarwal. A unified approach for maximizing continuous DR-submodular functions. *arXiv preprint arXiv:2305.16671*, 2023.

[Salehi *et al.*, 2017] Mehraveh Salehi, Amin Karbasi, Dustin Scheinost, and R. Todd Constable. A submodular approach to create individualized parcellations of the human brain. In Maxime Descoteaux, Lena Maier-Hein, Alfred M. Franz, Pierre Jannin, D. Louis Collins, and Simon Duchesne, editors, *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, volume 10433 of *Lecture Notes in Computer Science*, pages 478–485. Springer, 2017.

[Shi *et al.*, 2021] Guangyao Shi, Ishat E Rabban, Lifeng Zhou, and Pratap Tokekar. Communication-aware multi-robot coordination with submodular maximization. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8955–8961. IEEE, 2021.

[Soma and Yoshida, 2017] Tasuku Soma and Yuichi Yoshida. Non-monotone DR-submodular function maximization. In Satinder Singh and Shaul Markovitch, editors, *AAAI Conference on Artificial Intelligence*, pages 898–904. AAAI Press, 2017.

[Thắng and Srivastav, 2021] Nguyễn Kim Thắng and Abhinav Srivastav. Online non-monotone DR-submodular maximization. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 9868–9876. AAAI Press, 2021.

[Tukan *et al.*, 2023] Murad Tukan, Fares Fares, Yotam Grufinkle, Ido Talmor, Loay Mualem, Vladimir Braverman, and Dan Feldman. Orbslam3-enhanced autonomous toy drones: Pioneering indoor exploration. *arXiv preprint arXiv:2312.13385*, 2023.

[Viswanath *et al.*, 2009] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P. Gummadi. On the evolution of user interaction in facebook. In *ACM SIGCOMM Workshop on Social Networks (WOSN)*, August 2009.

[Vondrák, 2013] Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM J. Comput.*, 42(1):265–304, 2013.

[Xia *et al.*, 2020] Wei Xia, Juan-Carlos Vera, and Luis F. Zuluaga. Globally solving nonconvex quadratic programs via linear integer programming techniques. *INFORMS J. Comput.*, 32(1):40–56, 2020.

[Yelp, 2019] Yelp Dataset. https://www.yelp.com/dataset, 2019.

[Yuan and Tang, 2023] Jing Yuan and Shaojie Tang. Group fairness in non-monotone submodular maximization. *J. Comb. Optim.*, 45(3):88, 2023.

[Zhang *et al.*, 2019] Mingrui Zhang, Lin Chen, Hamed Hassani, and Amin Karbasi. Online continuous submodular maximization: From full-information to bandit feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

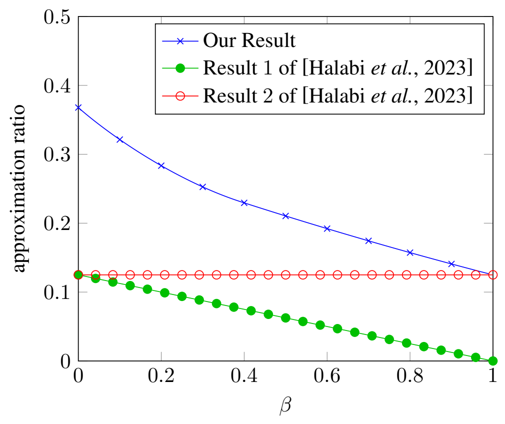


Figure 2: Results for the Fairness Setting of [Halabi *et al.*, 2023] (for $r = 1/2$)

A Implication for a Fairness Setting

El Halabi et al. [2023] considered the following fairness setting. The input for this setting is a ground set $\mathcal{N}$, a non-negative discrete submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$, a matroid $\mathcal{M} = (\mathcal{N}, \mathcal{I})$,[6] a partition of the elements in $\mathcal{N}$ into k disjoint classes $C_1, C_2, \ldots, C_k$ and integral lower and upper bounds ℓ_i and u_i for each class C_i, respectively. The objective is to find a set $S \in \mathcal{I}$ maximizing f subject to fairness constraints requiring that $\ell_i \leq |S \cap C_i| \leq u_i$ for every class C_i. If one only requires the fairness constraints to hold in expectation, then El Halabi et al. [2023] showed that is possible to reduce their setting to the following continues settings.[7] In this continuous settings, one has to find a vector $\mathbf{x}$ in the matroid polytope $P_{\mathcal{M}}$ that maximizing the multilinear extension F of f subject to fairness constraints requiring that $\ell_i \leq \|\mathbf{x} \cap \chi_{C_i}\|_1 \leq u_i$ for every class C_i, where χ_{C_i} is the characteristic vector of the set C_i. Since multilinear extensions of discrete submodular functions are DR-submodular [Bian *et al.*, 2017b], our offline results can be applied to this continues setting.

Some of the results of El Halabi et al. [2023] are bi-criteria approximation algorithms that are allowed to output solutions $\mathbf{x} \in P_{\mathcal{M}}$ that, for some parameter $\beta \in (0, 1)$, only obey $\beta\ell_i \leq \|\mathbf{x} \cap \chi_{C_i}\|_1 \leq u_i$ for every class C_i. In our terminology, allowing such solutions implies two things.

- $F(\mathbf{o}_{\mathrm{D}}^{(1)})$ and $F(\mathbf{o}_{\mathrm{D}}^{(2)})$ can be both made to be at least $(1 - \beta) \cdot F(\mathbf{o})$, where $\mathbf{o}$ is the optimal solution; and
- $m = \beta r$, where $r = \min_{\mathbf{x} \in P_{\mathcal{M}}} \|\mathbf{x}\|_\infty$.

Thus, our offline algorithms can be used to get a solution whose approximation ratio is at least[8]

$$(1 - \beta r) \cdot \max_{t_s \in [0,1]} \max_{T \in [t_s, 1]} \left\{ (T - t_s)e^{-T} \cdot (1 - \beta) + \frac{t_s^2 \cdot e^{-t_s - T}}{2} \cdot (1 - \beta) + e^{-T} - e^{-t_s - T} \right\} .$$

In Figure 2, we compare the above approximation ratio with two results of [Halabi *et al.*, 2023] (for $r = 1/2$). The first result is an approximation ratio of $(1 - \beta)/8$, which is worse than the approximation ratio that we obtain for any β, but is not based on the above mentioned reduction to the continuous setting (and thus, guarantees that the fairness constraints $\beta\ell_i \leq \|\mathbf{x} \cap \chi_{C_i}\|_1 \leq u_i$ hold always, and not just in expectation). The second result of [Halabi *et al.*, 2023] used in our comparison is an approximation of $(1 - r)/4$, which applies even for $\beta = 1$ (but like our result, satisfies the fairness constraints only in expectation). Our result can be viewed as a generalization of this second result.

B Known Results

In this section, we review a few known results used in our proofs. Bian et al. [2017b] observed that DR-submodular functions are concave along non-negative directions. This implies the following important lemma.

Lemma B.1. *Let $F\colon [0, 1]^n \to \mathbb{R}_{\geq 0}$ be a non-negative differentiable DR-submodular function. Then,*

[6] We refer the reader to [Halabi *et al.*, 2023] for the definition of matroids and the other terms used in this section.

[7] If the classes are large, then, in addition to guaranteeing that the fairness constraints hold in expectation, the reduction also guarantees that, with high probability, each fairness constraint is violated by at most a small amount.

[8] We omitted the error term in the approximation ratio. Since the smoothness of multilinear extensions of discrete submodular functions is polynomial, this error term can be made an arbitrarily small constant. For details, see, for example, Appendix A of [Buchbinder and Feldman, 2023].

1. $\langle \nabla F(\mathbf{x}), \mathbf{y}\rangle \geq F(\mathbf{x}+\mathbf{y}) - F(\mathbf{x})$ *for every* $\mathbf{x} \in [0,1]^n$ *and* $\mathbf{y} \geq \bar{0}$ *such that* $\mathbf{x}+\mathbf{y} \leq \bar{1}$.

2. $\langle \nabla F(\mathbf{x}), \mathbf{y}\rangle \leq F(\mathbf{x}) - F(\mathbf{x}-\mathbf{y})$ *for every* $\mathbf{x} \in [0,1]^n$ *and* $\mathbf{y} \geq \bar{0}$ *such that* $\mathbf{x}-\mathbf{y} \geq \bar{0}$.

We also need the following lemma, which generalizes Lemma 2.3 of [Feige *et al.*, 2011].

Lemma B.2 (Lemma 4.3 of [Buchbinder and Feldman, 2023])**.** *Given a DR-submodular function* $F\colon [0,1]^n \to \mathbb{R}$*, integer value* $r \geq 1$*, vectors* $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(r)} \in [0,1]^n$*, and values* $p_1, p_2, \ldots, p_r \in [0,1]$*,*

$$F\left(\bigoplus_{i=1}^{r}(p_i \cdot \mathbf{x}^{(i)})\right) \geq \sum_{S\subseteq[r]}\left[\prod_{i\in S} p_i \cdot \prod_{i\in[r]\setminus S}(1-p_i)\cdot F\left(\bigoplus_{i\in S}\mathbf{x}^{(i)}\right)\right] .$$

One important consequence of the last lemma is given by the next corollary. We note that this corollary can be viewed as an extension of Lemma 2.2 of [Feldman *et al.*, 2011].

Corollary B.3. *Given a non-negative DR-submodular function* $F\colon [0,1]^n \to \mathbb{R}_{\geq 0}$ *and two vectors* $\mathbf{x}, \mathbf{y} \in [0,1]^n$*,* $F(\mathbf{x} \oplus \mathbf{y}) \geq (1 - \|\mathbf{y}\|_\infty) \cdot F(\mathbf{x})$*.*

Proof. If $\|\mathbf{y}\|_\infty = 0$, then $\mathbf{y} = \bar{0}$, which makes the corollary trivial. Otherwise, Lemma B.2 and the non-negativity of F imply together that

$$F(\mathbf{x}\oplus\mathbf{y}) = F\Big(\mathbf{x}\oplus\Big(\|\mathbf{y}\|_\infty\cdot\frac{\mathbf{y}}{\|\mathbf{y}\|_\infty}\Big)\Big) \geq (1-\|\mathbf{y}\|_\infty)F(\mathbf{x}). \qquad \square$$

C Omitted Offline Algorithms and Proofs

In this section, we provide the full omitted analysis of Algorithm 1, and then present two additional versions of our offline algorithm (as discussed in Section 4). For ease of reading, we use $\mathbf{o}$ below to denote the sum $\mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}^{(1)}$. We also assume, without loss of generality, that $F(\mathbf{o}_{\mathrm{D}}^{(2)}) \geq F(\mathbf{o}_{\mathrm{D}}^{(1)})$. If this inequality is violated, then the guarantee of Theorem 4.1 follows from the guarantee of the same theorem for the case in which $\mathbf{o}_{\mathrm{D}}^{(2)}$ is replaced with $\mathbf{o}_{\mathrm{D}}^{(1)}$ (which is a case in which the inequality $F(\mathbf{o}_{\mathrm{D}}^{(2)}) \geq F(\mathbf{o}_{\mathrm{D}}^{(1)})$ trivially holds).

C.1 Analysis of Algorithm 1

It is clear that algorithm Algorithm 1 runs in polynomial time. Therefore, we concentrate on proving that the output vector of Algorithm 1 obeys the properties stated in Theorem 4.1. We begin by showing that this vector belongs to $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$.

Lemma C.1. *For every integer* $0 \leq i \leq \varepsilon^{-1}$*,* $\mathbf{y}^{(i)} \in \mathcal{K}_{\mathrm{N}}$ *and* $\mathbf{z}^{(i)} \in \varepsilon i \cdot \mathcal{K}_{\mathrm{D}}$*, where* $\varepsilon i \cdot \mathcal{K}_{\mathrm{D}} \triangleq \{\varepsilon i \cdot \mathbf{x} \mid \mathbf{x} \in \mathcal{K}_{\mathrm{D}}\}$*. Hence,* $\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} \in (\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$*.*

Proof. We begin the proof by showing that the first part of the lemma implies its second part. Assume that $\mathbf{y}^{(i)} \in \mathcal{K}_{\mathrm{N}}$ and $\mathbf{z}^{(i)} \in \varepsilon i \cdot \mathcal{K}_{\mathrm{D}} \subseteq \mathcal{K}_{\mathrm{D}}$ (the inclusion holds by the down-monotonicity of $\mathcal{K}_{\mathrm{D}}$). The definition of $\oplus$ guarantees that we always have $\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} \in [0,1]^n$. Thus, to prove that $\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} \in (\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$, it suffices to show that $\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}$ is the sum of a vector in $\mathcal{K}_{\mathrm{N}}$ and a vector in $\mathcal{K}_{\mathrm{D}}$, which is the case since $\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} = \mathbf{y}^{(i)} + (\bar{1} - \mathbf{y}^{(i)}) \odot \mathbf{z}^{(i)}$ and $(\bar{1} - \mathbf{y}^{(i)}) \odot \mathbf{z}^{(i)} \in \mathcal{K}_{\mathrm{D}}$ by the down-closeness of $\mathcal{K}_{\mathrm{D}}$.

In the rest of the proof, we prove the first part of the lemma by induction. The base of the induction holds by the initializations of $\mathbf{y}^{(0)}$ and $\mathbf{z}^{(0)}$. Assume now that both $\mathbf{y}^{(i-1)} \in \mathcal{K}_{\mathrm{N}}$ and $\mathbf{z}^{(i-1)} \in \varepsilon(i-1) \cdot \mathcal{K}_{\mathrm{D}}$ hold for some integer $1 \leq i \leq \varepsilon^{-1}$, and let us prove that we also have $\mathbf{y}^{(i)} \in \mathcal{K}_{\mathrm{N}}$ and $\mathbf{z}^{(i)} \in \mathcal{K}_{\mathrm{D}}$. For $\mathbf{y}^{(i)}$ this is true since $\mathcal{K}_{\mathrm{N}}$ is convex and $\mathbf{y}^{(i)}$ is defined as a convex combination of $\mathbf{y}^{(i-1)}$ and $\mathbf{a}^{(i)}$, which are both vectors in $\mathcal{K}_{\mathrm{N}}$. Additionally, since $\mathbf{z}^{(i-1)} \in \varepsilon(i-1) \cdot \mathcal{K}_{\mathrm{D}}$, there must exist a vector $\mathbf{x} \in \mathcal{K}_{\mathrm{D}}$ such that $\mathbf{z}^{(i-1)} = \varepsilon(i-1) \cdot \mathbf{x}$. Hence,

$$\begin{aligned}\mathbf{z}^{(i)} &= \mathbf{z}^{(i-1)} + \varepsilon \cdot (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)} \leq \mathbf{z}^{(i-1)} + \varepsilon \cdot \mathbf{b}^{(i)}\\ &= \varepsilon(i-1)\cdot\mathbf{x} + \varepsilon\cdot\mathbf{b}^{(i)} = \varepsilon i \cdot [(1-i^{-1})\cdot\mathbf{x} + i^{-1}\cdot\mathbf{b}^{(i)}] \in \varepsilon i \cdot \mathcal{K}_{\mathrm{D}} \enspace ,\end{aligned}$$

where the inclusion holds since the convexity of $\mathcal{K}_{\mathrm{D}}$ and the fact that both $\mathbf{x}$ and $\mathbf{b}^{(i)}$ are vectors in $\mathcal{K}_{\mathrm{D}}$ imply together that $(1-i^{-1})\cdot\mathbf{x} + i^{-1}\cdot\mathbf{b}^{(i)} \in \mathcal{K}_{\mathrm{D}}$. Thus, $\mathbf{z}^{(i)}$ is upper bounded by a vector in $\varepsilon i \cdot \mathcal{K}_{\mathrm{D}}$, which implies that $\mathbf{z}^{(i)}$ itself also belongs to $\varepsilon i \cdot \mathcal{K}_{\mathrm{D}}$ because the down-closeness of $\mathcal{K}_{\mathrm{D}}$ implies that $\varepsilon i \cdot \mathcal{K}_{\mathrm{D}}$ is also down-closed. $\square$

Our next goal is to lower bound the value of the output vector of Algorithm 1. We begin with the following lemma, which bounds the infinity norm of $\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}$.

Lemma C.2. *For every integer* $0 \leq i \leq \varepsilon^{-1}$*,* $\|\mathbf{z}^{(i)}\|_\infty \leq 1 - (1-\varepsilon)^i$ *and* $\|\mathbf{y}^{(i)}\|_\infty \leq \|\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}\|_\infty \leq 1 - (1-\varepsilon)^i(1-m)$*.*

Proof. We prove the lemma by induction. For $i = 0$, the lemma holds since our choice of values for $\mathbf{y}^{(0)}$ and $\mathbf{z}^{(0)}$ guarantees that $\|\mathbf{z}^{(0)}\|_\infty = \|\bar{0}\|_\infty = 0$ and $\|\mathbf{y}^{(0)} \oplus \mathbf{z}^{(0)}\|_\infty = \|\mathbf{y}^{(0)}\|_\infty = m$. Let us now prove the lemma for $i \geq 1$ assuming it holds for $i - 1$. Note that

$$\begin{aligned}
\bar{1} - \mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} &= (\bar{1} - \mathbf{y}^{(i)}) \odot (\bar{1} - \mathbf{z}^{(i)}) \\
&= \left(\bar{1} - (1-\varepsilon) \cdot \mathbf{y}^{(i-1)} - \varepsilon \mathbf{a}^{(i)}\right) \odot \left(\bar{1} - \mathbf{z}^{(i-1)} - \varepsilon(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}\right) \\
&\geq (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \varepsilon(\bar{1} - \mathbf{b}^{(i)})) \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \varepsilon \mathbf{b}^{(i)}) \\
&\geq (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}) \cdot (1 - \varepsilon) = (1-\varepsilon) \cdot (\bar{1} - \mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace ,
\end{aligned}$$

where the first inequality uses the fact that the inequality $\mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \leq \bar{1}$ is one of the conditions of the linear program of Algorithm 1, which implies $\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)} \leq (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{b}^{(i)})$. Hence, by the induction hypothesis,

$$\begin{aligned}
\|\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}\|_\infty = \max_{j \in [n]} (\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)})_j &\leq \max_{j \in [n]} [1 - (1-\varepsilon) \cdot (1 - (\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})_j)] \\
&= 1 - (1-\varepsilon) \cdot (1 - \max_{j \in [n]} (\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})_j) = 1 - (1-\varepsilon) \cdot (1 - \|\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}\|_\infty) \\
&\leq 1 - (1-\varepsilon) \cdot [(1-\varepsilon)^{i-1}(1-m)] = 1 - (1-\varepsilon)^i (1-m) \enspace .
\end{aligned}$$

Similarly, the induction hypothesis also implies that

$$\begin{aligned}
\|\mathbf{z}^{(i)}\|_\infty &= \|\mathbf{z}^{(i-1)} + \varepsilon(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}\|_\infty \leq \|\mathbf{z}^{(i-1)} + \varepsilon(\bar{1} - \mathbf{z}^{(i-1)})\|_\infty \\
&= \varepsilon + (1-\varepsilon) \cdot \|\mathbf{z}^{(i-1)}\|_\infty \leq \varepsilon + (1-\varepsilon) \cdot (1 - (1-\varepsilon)^{i-1}) = 1 - (1-\varepsilon)^i \enspace . \qquad \square
\end{aligned}$$

Using the last lemma, we prove two lower bounds on the optimal value of the linear program solved by Algorithm 1. Each one of these lower bounds is based on one possible solution for this linear program. The first such solution is given by the next lemma.

Lemma C.3. *For every integer $1 \leq i \leq \varepsilon^{-1}$, the assignment $\mathbf{a}^{(i)} = \mathbf{o}_{\mathrm{N}}$ and $\mathbf{b}^{(i)} = \mathbf{o}_{\mathrm{D}}^{(1)}$ is a feasible solution for the linear program solved by Algorithm 1 in iteration number i.*

Proof. The definitions of $\mathbf{o}_{\mathrm{N}}$ and $\mathbf{o}_{\mathrm{D}}^{(1)}$ immediately implies that the first two constraints of the linear program hold for the solution stated in the lemma. Thus, we concentrate on proving that this solution obeys also the other two constraints of the linear program. The third constraint of the linear program is

$$\langle \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{z}^{(i-1)}) \rangle \geq (1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)}) \enspace .$$

To see that this constraint is satisfied by our proposed solution, notice that, by Property 1 of Lemma B.1,

$$\begin{aligned}
\langle \mathbf{o}_{\mathrm{D}}^{(1)} \odot (\bar{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{z}^{(i-1)}) \rangle &\geq F(\mathbf{o}_{\mathrm{D}}^{(1)} \oplus \mathbf{z}^{(i-1)}) - F(\mathbf{z}^{(i-1)}) \\
&\geq (1 - \|\mathbf{z}^{(i-1)}\|_\infty) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)}) \geq (1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)}) \enspace ,
\end{aligned}$$

where the second inequality holds by Corollary B.3, and the last inequality follows from Lemma C.2. The last constraint of the linear program is

$$\mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \leq \bar{1} \enspace .$$

This constraint is also satisfied by our proposed solution since $\mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}^{(1)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \leq \mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}^{(1)} \leq \bar{1}$. $\square$

As promised, we can now get a lower bound on the optimal value of the linear program solved by Algorithm 1. Recall that the objective function of this linear program is given (in iteration i of the algorithm) by $\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \rangle$.

Lemma C.4. *For every integer $1 \leq i \leq \varepsilon^{-1}$,*

$$\begin{aligned}
&\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \rangle \\
&\qquad \geq (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{o}) + (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{z}^{(i-1)}) \\
&\qquad\quad + \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{z}^{(i-1)}) \rangle - 2F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace .
\end{aligned}$$

Proof. Since Lemma C.3 guarantees that $\mathbf{a}^{(i)} = \mathbf{o}_{\mathrm{N}}$ and $\mathbf{b}^{(i)} = \mathbf{o}_{\mathrm{D}}^{(1)}$ is one feasible solution for the linear program solved in iteration i of Algorithm 1, we get

$$\begin{aligned}
&\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\rangle \qquad (1)\\
\geq{}& \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}^{(1)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\rangle\\
={}& \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{o} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle\\
&\qquad + \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{o}_{\mathrm{N}} \odot \mathbf{y}^{(i-1)} \odot (1 - \mathbf{z}^{(i-1)})\rangle \enspace .
\end{aligned}$$

The first term on the rightmost side of this inequality can be lower bounded, by Property 1 of Lemma B.1, as follows.

$$\begin{aligned}
\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{o} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle &\geq F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} \oplus \mathbf{o}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\
&\geq (1 - \|\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}\|_\infty) \cdot F(\mathbf{o}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\
&\geq (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{o}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace ,
\end{aligned}$$

where the second inequality holds by Corollary B.3, and the last inequality follows from Lemma C.2.

Next, we need to lower bound the second term on the rightmost side of Inequality (1).

$$\begin{aligned}
\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{o}_{\mathrm{N}} \odot \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle &- \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle\\
&= -\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} - \mathbf{z}^{(i-1)} \oplus (\mathbf{o}_{\mathrm{N}} \odot \mathbf{y}^{(i-1)})\rangle\\
&\geq F(\mathbf{z}^{(i-1)} \oplus (\mathbf{o}_{\mathrm{N}} \odot \mathbf{y}^{(i-1)})) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\
&\geq (1 - \|\mathbf{o}_{\mathrm{N}} \odot \mathbf{y}^{i-1}\|_\infty) \cdot F(\mathbf{z}^{(i-1)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\
&\geq (1 - \|\mathbf{y}^{i-1}\|_\infty) \cdot F(\mathbf{z}^{(i-1)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\
&\geq (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{z}^{(i-1)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace ,
\end{aligned}$$

where the first inequality holds by Property (2) of Lemma B.1 since $\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} - \mathbf{z}^{(i-1)} \oplus (\mathbf{o}_{\mathrm{N}} \odot \mathbf{y}^{(i-1)} \geq \mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} - \mathbf{z}^{(i-1)} \oplus (\bar{1} \odot \mathbf{y}^{(i-1)}) = \bar{0}$, the second inequality follows from Corollary B.3, and the last inequality holds due to Lemma C.2. □

The lower bound given by the last lemma depends on the term $F(\mathbf{z}^{(i-1)})$. Thus, to make this lower bound useful, we need to prove also a lower bound on this term, which is done by the next lemma.

Lemma C.5. *For every integer* $0 \leq i \leq \varepsilon^{-1}$, $F(\mathbf{z}^{(i)}) \geq \varepsilon i \cdot (1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - i \cdot \varepsilon^2 \beta D^2 / [2(1-m)^2]$.

Proof. We prove the lemma by induction on i. For $i = 0$, the lemma trivially holds by the non-negativity of F. Assume now that the lemma holds for $i - 1$, and let us prove it for $i \geq 1$. By the chain rule,

$$\begin{aligned}
F(\mathbf{z}^{(i)}) &= F(\mathbf{z}^{(i-1)} + \varepsilon(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)})\\
&= F(\mathbf{z}^{(i-1)}) + \varepsilon \cdot \langle (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle\\
&\qquad + \int_0^\varepsilon \langle (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}, \nabla F(\mathbf{z}^{(i-1)} + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}) - \nabla F(\mathbf{z}^{(i-1)})\rangle d\tau\\
&\geq F(\mathbf{z}^{(i-1)}) + \varepsilon \cdot [(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)})]\\
&\qquad - \int_0^\varepsilon \|(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}\|_2 \cdot \|\nabla F(\mathbf{z}^{(i-1)} + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}) - \nabla F(\mathbf{z}^{(i-1)})\|_2 d\tau\\
&\geq (1-\varepsilon) \cdot F(\mathbf{z}^{(i-1)}) + \varepsilon(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - \int_0^\varepsilon \tau \cdot \beta \|(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}\|_2^2 d\tau\\
&\geq (1-\varepsilon) \cdot F(\mathbf{z}^{(i-1)}) + \varepsilon(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - \varepsilon^2 \beta D^2 / [2(1-m)^2] \enspace ,
\end{aligned}$$

where the first inequality follows from the Cauchy–Schwarz inequality and fact that $\mathbf{b}^{(i)}$ is part of a feasible solution for the linear program that Algorithm 1 solves at iteration i, the second inequality holds by the β-smoothness of F, and the last inequality uses the observation that since $\|\mathbf{y}^{(0)}\|_\infty = m$ and $\mathcal{K}_{\mathrm{D}}$ is down-closed, both $\mathbf{y}^{(0)}$ and $\mathbf{y}^{(0)} + (1-m) \cdot \mathbf{b}^{(i)}$ are vectors in $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$, and thus,

$$\|(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}\|_2 \leq \|\mathbf{b}^{(i)}\|_2 = \frac{\|(\mathbf{y}^{(0)} + (1-m) \cdot \mathbf{b}^{(i)}) - \mathbf{y}^{(0)}\|_2}{1-m} \leq \frac{D}{1-m} \enspace .$$

Plugging the induction hypothesis into the last inequality yields

$$\begin{aligned}F(\mathbf{z}^{(i)}) &\geq (1-\varepsilon)\cdot[\varepsilon(i-1)\cdot(1-\varepsilon)^{i-2}\cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})] + \varepsilon(1-\varepsilon)^{i-1}\cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - i\cdot\varepsilon^2\beta D^2/[2(1-m)^2]\\ &= \varepsilon i\cdot(1-\varepsilon)^{i-1}\cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - i\cdot\varepsilon^2\beta D^2/[2(1-m)^2]\ . \qquad \square\end{aligned}$$

We now present, in Lemma C.6, another possible solution for the linear program solved by Algorithm 1. Corollary C.7 then states the lower bound implied by this solution for the optimal value of the objective function of this linear program.

Lemma C.6. *For every integer $1 \leq i \leq \varepsilon^{-1}$, the assignment $\mathbf{a}^{(i)} = \mathbf{y}^{(i-1)}$ and $\mathbf{b}^{(i)} = \mathbf{o}_{\mathrm{D}}^{(2)}$ is a feasible solution for the linear program solved by Algorithm 1 in iteration number i.*

Proof. Lemma C.1 shows that $\mathbf{a}^{(i)} = \mathbf{y}^{(i-1)} \in \mathcal{K}_{\mathrm{N}}$, and by definition we have $\mathbf{o}_{\mathrm{D}}^{(2)} \in \mathcal{K}_{\mathrm{D}}$. Thus, the first two constraints of the linear program are satisfied by the solution stated in the lemma. The third constraint of this linear program is

$$\langle \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{z}^{(i-1)})\rangle \geq (1-\varepsilon)^{i-1}\cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)})\ .$$

Repeating the part of the proof of Lemma C.3 related to this constraint, with $\mathbf{o}_{\mathrm{D}}^{(2)}$ taking the role of $\mathbf{o}_{\mathrm{D}}^{(1)}$, we get

$$\begin{aligned}\langle \mathbf{o}_{\mathrm{D}}^{(2)} \odot (\bar{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{z}^{(i-1)})\rangle &\geq (1-\varepsilon)^{i-1}\cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - F(\mathbf{z}^{(i-1)})\\ &\geq (1-\varepsilon)^{i-1}\cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)})\ ,\end{aligned}$$

where the second inequality holds by our assumption that $F(\mathbf{o}_{\mathrm{D}}^{(2)}) \geq F(\mathbf{o}_{\mathrm{D}}^{(1)})$. Hence, the above stated third constraint is satisfied by our solution, and it only remains to prove that this solution also satisfies the last constraint of the linear program, which is

$$\mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \leq \bar{1}\ .$$

This is indeed the case since $\mathbf{y}^{(i-1)} + \mathbf{o}_{\mathrm{D}}^{(2)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \leq \mathbf{y}^{(i-1)} + (\bar{1} - \mathbf{y}^{(i-1)}) = \bar{1}$. $\square$

Corollary C.7. *For every integer $1 \leq i \leq \varepsilon^{-1}$,*

$$\begin{aligned}\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\rangle &\geq (1-\varepsilon)^{i-1}(1-m)\cdot F(\mathbf{o}_{\mathrm{D}}^{(2)})\\ &- F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle\ .\end{aligned}$$

Proof. Recall that the left hand side of the inequality of the lemma is the objective function of the linear program solved by Algorithm 1 in iteration i. Thus, its value is at least the value obtained by plugging in the feasible solution described by Lemma C.6. Hence,

$$\begin{aligned}&\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{a}^{(i)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\rangle\\ &\quad\geq \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{y}^{(i-1)} + \mathbf{o}_{\mathrm{D}}^{(2)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\rangle\\ &\quad= \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{o}_{\mathrm{D}}^{(2)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle\\ &\qquad + \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle\ .\end{aligned}$$

To complete the proof of the corollary, it remains to observe that

$$\begin{aligned}&\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{o}_{\mathrm{D}}^{(2)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\\ &\quad\geq F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} \oplus \mathbf{o}_{\mathrm{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\ &\quad\geq (1 - \|\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}\|_\infty)\cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\ &\quad\geq (1-\varepsilon)^{i-1}(1-m)\cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\ ,\end{aligned}$$

where the first inequality holds by Property 1 of Lemma B.1, the second inequality holds by Corollary B.3, and the last inequality follows from Lemma C.2. $\square$

Using the above results, we can now prove the following lemma about the rate in which the value of $F(y^{(i)})$ increases as a function of i.

Lemma C.8. *For every integer $1 \leq i \leq \varepsilon^{-1}$, the value of $F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})$ can be lower bounded by both expressions*

$$\begin{aligned}&\varepsilon(1-m)\cdot[(1-\varepsilon)^i\cdot F(\mathbf{o}) + \varepsilon(1-\varepsilon)^{2i-2}(i-1)\cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})]\\ &\qquad - 2\varepsilon\cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - 3\varepsilon^2\cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - O(\tfrac{\varepsilon^2\beta D^2}{1-m})\ ,\end{aligned}$$

and

$$\varepsilon(1-m)\cdot(1-\varepsilon)^i\cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - \varepsilon\cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - 3\varepsilon^2\cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - O(\tfrac{\varepsilon^2\beta D^2}{1-m})\ .$$

Proof. By the chain rule,

$$\begin{aligned}
&F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \qquad (2)\\
&\quad = F(((1-\varepsilon)\mathbf{y}^{(i-1)} + \varepsilon\mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \varepsilon(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)})) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\
&\quad = \int_0^\varepsilon \Big\langle \frac{d[((1-\tau)\mathbf{y}^{(i-1)} + \tau\mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)})]}{d\tau},\\
&\qquad\qquad \nabla F(((1-\tau)\mathbf{y}^{(i-1)} + \tau\mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}))\Big\rangle d\tau\\
&\quad = \int_0^\varepsilon \langle(\bar{1} - \mathbf{z}^{(i-1)}) \odot [(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - 2\tau \cdot \mathbf{b}^{(i)})],\\
&\qquad\qquad \nabla F(((1-\tau)\mathbf{y}^{(i-1)} + \tau\mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}))\rangle d\tau \enspace .
\end{aligned}$$

At this point, we need to lower bound the integrand on the rightmost side of the last equality. The first step towards obtaining this lower bound is the following inequality.

$$\begin{aligned}
&\langle(\bar{1} - \mathbf{z}^{(i-1)}) \odot [(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - 2\tau \cdot \mathbf{b}^{(i)})], \qquad (3)\\
&\qquad \nabla F(((1-\tau)\mathbf{y}^{(i-1)} + \tau\mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)})) - \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle\\
&\geq -\|(\bar{1} - \mathbf{z}^{(i-1)}) \odot [(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - 2\tau \cdot \mathbf{b}^{(i)})]\|_2 \cdot\\
&\qquad \|\nabla F(((1-\tau)\mathbf{y}^{(i-1)} + \tau\mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)})) - \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\|_2\\
&\geq -3D \cdot \beta\|(((1-\tau)\mathbf{y}^{(i-1)} + \tau\mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)})) - (\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\|_2\\
&= -3\beta D \cdot \|\tau(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}) + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\\
&\qquad - \tau^2(\bar{1} - \mathbf{z}^{(i-1)}) \odot (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}\|_2 \geq -6\tau\beta D^2 \enspace ,
\end{aligned}$$

where the first inequality follows from the Cauchy-Schwarz inequality, and the second inequality uses the β-smoothness of F and the observation that since $\mathbf{a}^{(i)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}$, $\mathbf{a}^{(i)}$ and $\mathbf{y}^{(i-1)}$ are all vectors in $(\mathcal{K}_{\text{N}} + \mathcal{K}_{\text{D}}) \cap [0,1]^n$, it holds that

$$\begin{aligned}
&\|(\bar{1} - \mathbf{z}^{(i-1)}) \odot [(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - 2\tau \cdot \mathbf{b}^{(i)})]\|_2\\
&\qquad \leq \|(\mathbf{a}^{(i)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}) - \mathbf{y}^{(i-1)}\|_2 + 2\tau \cdot \|(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}\|_2\\
&\qquad \leq D + 2\tau \cdot \|\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}\|_2 \leq 3D \enspace .
\end{aligned}$$

Similarly, the last inequality of Inequality (3) holds since

$$\begin{aligned}
&\|\tau(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}) + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\\
&\qquad - \tau^2(\bar{1} - \mathbf{z}^{(i-1)}) \odot (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}\|_2\\
&\quad = \|\tau(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (1 - \tau\mathbf{b}^{(i)}) + \tau(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2\\
&\quad \leq \|\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}\|_2 + \|(\mathbf{y}^{(i-1)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}) - \mathbf{y}^{(i-1)}\|_2 \leq 2D \enspace .
\end{aligned}$$

We now need another inequality.

$$\begin{aligned}
&\langle(\bar{1} - \mathbf{z}^{(i-1)}) \odot [(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - 2\tau \cdot \mathbf{b}^{(i)})], \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle\\
&\quad = (1 - 2\tau) \cdot \{\langle(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + \mathbf{a}^{(i)}, \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle\\
&\qquad - \langle\mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle\}\\
&\qquad + 2\tau \cdot \langle(\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}, \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle\\
&\qquad + 2\tau \cdot \langle(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{a}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{b}^{(i)}), \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle\\
&\qquad - 2\tau \cdot \langle(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{a}^{(i)}) \odot (\bar{1} - \mathbf{b}^{(i)}), \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle\\
&\quad \geq (1 - 2\tau) \cdot \{\langle(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + \mathbf{a}^{(i)}, \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle\\
&\qquad - \langle\mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle\}\\
&\qquad + 2\tau \cdot \{F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} \oplus \mathbf{b}^{(i)}) + F(\mathbf{y}^{i-1} \oplus \mathbf{z}^{(i-1)} \oplus ((\bar{1} - \mathbf{b}^{(i)}) \odot \mathbf{a}^{(i)}))\\
&\qquad + F(((\mathbf{a}^{(i)} \oplus \mathbf{b}^{(i)}) \odot \mathbf{y}^{(i-1)}) \oplus \mathbf{z}^{(i-1)}) - 3F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\}\\
&\quad \geq (1 - 2\tau) \cdot \max\{(1 - m) \cdot [(1 - \varepsilon)^{i-1} F(\mathbf{o}) + (1 - \varepsilon)^{2i-3} \cdot \varepsilon(i-1) \cdot F(\mathbf{o}_{\text{D}}^{(1)})] - O(\tfrac{\varepsilon\beta D^2}{1-m})
\end{aligned}$$

$$- 2F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{o}_{\text{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\}$$
$$- 6\tau \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace ,$$

where the second inequality holds by Properties 1 and 2 of Lemma B.1, and the last inequality follows from Lemmata C.4 and C.5, Corollary C.7 and the non-negativity of F.

Adding the last inequality to Inequality (3), we get the promised lower bound on the integrand on the rightmost side of Equality (2); and plugging this lower bound into the equality yields

$$\begin{aligned}
&F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \\
&\quad \geq (\varepsilon - \varepsilon^2) \cdot \max\{(1-m) \cdot [(1-\varepsilon)^{i-1} F(\mathbf{o}) + (1-\varepsilon)^{2i-3} \cdot \varepsilon(i-1) \cdot F(\mathbf{o}_{\text{D}}^{(1)})] \\
&\qquad - 2F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{o}_{\text{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\} \\
&\qquad - 3\varepsilon^2 \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - O(\tfrac{\varepsilon^2 \beta D^2}{1-m}) \enspace . \qquad \square
\end{aligned}$$

The last lemma implies recursive lower bounds on the values of $\{F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) \mid i \in \mathbb{Z}, 0 \leq i \leq \varepsilon^{-1}\}$. We need to get closed expression forms for these lower bounds, and the next two lemmata are steps towards this goal. To simplify the statements of these lemmata, it is useful to define $\alpha \triangleq (1-m)(1-\varepsilon)^2$ and $g(t) \triangleq (e^{-t} - e^{-2t}) \cdot F(\mathbf{o}) + \frac{t^2 \cdot e^{-2t}}{2} \cdot F(\mathbf{o}_{\text{D}}^{(1)})$.

Lemma C.9. *For every integer* $0 \leq i \leq \varepsilon^{-1}$, $F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) \geq \alpha \cdot g(\varepsilon i) - 3\varepsilon^2 \cdot \sum_{i'=1}^{i} F(\mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)}) - i \cdot O(\varepsilon^2(\alpha \cdot F(\mathbf{o}) + \alpha \cdot F(\mathbf{o}_{\text{D}}^{(1)}) + \frac{\beta D^2}{1-m}))$.

Proof. We prove the lemma by induction on i. For $i = 0$, the lemma immediately follows from the non-negativity of F since $g(0) = 0$. Assume now that the lemma holds for $i-1$, and let us prove it for $i \geq 1$.

Observe that
$$g'(t) = (2e^{-2t} - e^{-t}) \cdot F(\mathbf{o}) + te^{-2t}(1-t) \cdot F(\mathbf{o}_{\text{D}}^{(1)}) \enspace ;$$
and therefore,

$$\begin{aligned}
g(\varepsilon i) - g(\varepsilon(i-1)) &= \int_{\varepsilon(i-1)}^{\varepsilon i} [(2e^{-2t} - e^{-t}) \cdot F(\mathbf{o}) + te^{-2t}(1-t) \cdot F(\mathbf{o}_{\text{D}}^{(1)})] dt \qquad (4)\\
&\leq \int_{\varepsilon(i-1)}^{\varepsilon i} [(2e^{-2\varepsilon(i-1)} - e^{-\varepsilon(i-1)}) \cdot F(\mathbf{o}) + \varepsilon(i-1)e^{-2\varepsilon(i-1)}(1-\varepsilon(i-1)) \cdot F(\mathbf{o}_{\text{D}}^{(1)}) \\
&\qquad + O(\varepsilon(F(\mathbf{o}) + F(\mathbf{o}_{\text{D}}^{(1)})))] dt \\
&= \varepsilon[(2e^{-2\varepsilon(i-1)} - e^{-\varepsilon(i-1)}) \cdot F(\mathbf{o}) + \varepsilon(i-1)e^{-2\varepsilon(i-1)}(1-\varepsilon(i-1)) \cdot F(\mathbf{o}_{\text{D}}^{(1)})] \\
&\qquad + O(\varepsilon^2(F(\mathbf{o}) + F(\mathbf{o}_{\text{D}}^{(1)}))) \\
&= \varepsilon[e^{-\varepsilon(i-1)} \cdot F(\mathbf{o}) + \varepsilon(i-1)e^{-2\varepsilon(i-1)} \cdot F(\mathbf{o}_{\text{D}}^{(1)}) - 2g(\varepsilon(i-1))] + O(\varepsilon^2(F(\mathbf{o}) + F(\mathbf{o}_{\text{D}}^{(1)}))) \enspace ,
\end{aligned}$$

where the inequality holds since, for every $t \in [\varepsilon(i-1), \varepsilon i]$,

$$\begin{aligned}
&(2e^{-2t} - e^{-t}) \cdot F(\mathbf{o}) + te^{-2t}(1-t) \cdot F(\mathbf{o}_{\text{D}}^{(1)}) \\
&\qquad - (2e^{-2\varepsilon(i-1)} - e^{-\varepsilon(i-1)}) \cdot F(\mathbf{o}) - \varepsilon(i-1)e^{-2\varepsilon(i-1)}(1-\varepsilon(i-1)) \cdot F(\mathbf{o}_{\text{D}}^{(1)}) \\
&\quad = \int_{\varepsilon(i-1)}^{t} [(e^{-\tau} - 4e^{-2\tau}) \cdot F(\mathbf{o}) + e^{-2\tau}((1-\tau) - 2\tau(1-\tau) - \tau) \cdot F(\mathbf{o}_{\text{D}}^{(1)})] d\tau \\
&\quad = \int_{\varepsilon(i-1)}^{t} [(e^{-\tau} - 4e^{-2\tau}) \cdot F(\mathbf{o}) + e^{-2\tau}(1 - 4\tau + 2\tau^2) \cdot F(\mathbf{o}_{\text{D}}^{(1)})] d\tau \\
&\quad \leq \int_{\varepsilon(i-1)}^{t} \left(\frac{F(\mathbf{o})}{16} + F(\mathbf{o}_{\text{D}}^{(1)})\right) d\tau = O(\varepsilon(F(\mathbf{o}) + F(\mathbf{o}_{\text{D}}^{(1)}))) \enspace .
\end{aligned}$$

Rearranging Inequality (4) yields

$$(1-2\varepsilon) \cdot g(\varepsilon(i-1)) + \varepsilon[e^{-\varepsilon(i-1)} \cdot F(\mathbf{o}) + \varepsilon(i-1)e^{-2\varepsilon(i-1)} \cdot F(\mathbf{o}_{\text{D}}^{(1)})] \geq g(\varepsilon i) - O(\varepsilon^2(F(\mathbf{o}) + F(\mathbf{o}_{\text{D}}^{(1)}))) \enspace . \qquad (5)$$

Now, we can use Lemma C.8 (and the non-negativity of F) to get

$$F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) \geq (1 - 2\varepsilon - 3\varepsilon^2) \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})$$

$$
\begin{aligned}
&\quad + \varepsilon\alpha[(1-\varepsilon)^{i-2} \cdot F(\mathbf{o}) + \varepsilon(1-\varepsilon)^{2i-4}(i-1) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})] - O(\tfrac{\varepsilon^2\beta D^2}{1-m}) \\
&\geq (1-2\varepsilon) \cdot \Big[\alpha \cdot g(\varepsilon(i-1)) - 3\varepsilon^2 \cdot \sum_{i'=1}^{i-2} F(\mathbf{y}^{(i')} \oplus \mathbf{z}^{(i')}) \\
&\quad - (i-1) \cdot O(\varepsilon^2(\alpha \cdot F(\mathbf{o}) + \alpha \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) + \tfrac{\beta D^2}{1-m}))\Big] - 3\varepsilon^2 \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \\
&\quad + \varepsilon\alpha[e^{-\varepsilon(i-1)} \cdot F(\mathbf{o}) + \varepsilon e^{-2\varepsilon(i-1)}(i-1) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})] - O(\tfrac{\varepsilon^2\beta D^2}{1-m}) \\
&\geq \alpha \cdot g(\varepsilon i) - 3\varepsilon^2 \cdot \sum_{i'=1}^{i-1} F(\mathbf{y}^{(i')} \oplus \mathbf{z}^{(i')}) - i \cdot O(\varepsilon^2(\alpha \cdot F(\mathbf{o}) + \alpha \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) + \tfrac{\beta D^2}{1-m}) \enspace ,
\end{aligned}
$$

where the second inequality holds by the induction hypothesis, and the last inequality follows from Inequality (5). ☐

Lemma C.10. *For every two integers* $0 \leq i_s \leq i \leq \varepsilon^{-1}$, $F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) \geq \alpha\varepsilon(i-i_s)(1-\varepsilon)^{i-2} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) + (1-\varepsilon)^{i-i_s} \cdot F(\mathbf{y}^{(i_s)} \oplus \mathbf{z}^{(i_s)}) - 3\varepsilon^2 \cdot \sum_{i'=i_s+1}^{i} F(\mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)}) - (i-i_s) \cdot O(\frac{\varepsilon^2\beta D^2}{1-m})$.

Proof. We prove the lemma by induction on i. For $i = i_s$ the lemma is trivially true. Thus, assume that the lemma holds for $i-1$, and let us prove it for $i > i_s$. By Lemma C.8,

$$
\begin{aligned}
F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) &\geq (1-\varepsilon) \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + \varepsilon(1-m) \cdot (1-\varepsilon)^i \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) \\
&\qquad - 3\varepsilon^2 \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - O(\tfrac{\varepsilon^2\beta D^2}{1-m}) \\
&\geq (1-\varepsilon) \cdot \Big[\alpha\varepsilon(i-1-i_s)(1-\varepsilon)^{i-3} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) + (1-\varepsilon)^{i-i_s-1} \cdot F(\mathbf{y}^{(i_s)} \oplus \mathbf{z}^{(i_s)}) \\
&\qquad - \sum_{i'=i_s+1}^{i-1} F(\mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)}) - (i-i_s-1) \cdot O(\tfrac{\varepsilon^2\beta D^2}{1-m})\Big] \\
&\qquad + \alpha\varepsilon \cdot (1-\varepsilon)^{i-2} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - 3\varepsilon^2 \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - O(\tfrac{\varepsilon^2\beta D^2}{1-m}) \\
&\geq \alpha\varepsilon(i-i_s)(1-\varepsilon)^{i-2} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) + (1-\varepsilon)^{i-i_s} \cdot F(\mathbf{y}^{(i_s)} \oplus \mathbf{z}^{(i_s)}) \\
&\qquad - 3\varepsilon \cdot \sum_{i'=i_s+1}^{i} F(\mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)}) - (i-i_s) \cdot O(\tfrac{\varepsilon^2\beta D^2}{1-m}) \enspace ,
\end{aligned}
$$

where the second inequality holds by the induction hypothesis. ☐

We are ready now to prove the next corollary, which completes the proof of Theorem 4.1. Recall that the value of the output vector $\mathbf{w}$ of Algorithm 1 is $\max_{0 \leq i \leq \varepsilon^{-1}} F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)})$.

Corollary C.11. *It holds that*

$$
\begin{aligned}
(1+O(\varepsilon)) \cdot \max_{0 \leq i \leq \varepsilon^{-1}T} F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) &\geq \alpha \cdot \max_{t_s \in [0,1]} \max_{T \in [t_s, 1]} \Big\{((T-t_s)e^{-T} - O(\varepsilon)) \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) \\
&+ \left[\frac{t_s^2 \cdot e^{-t_s-T}}{2} - O(\varepsilon)\right] \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) + (e^{-T} - e^{-t_s-T} - O(\varepsilon)) \cdot F(\mathbf{o})\Big\} - O(\tfrac{\varepsilon\beta D^2}{1-m}) \enspace .
\end{aligned}
$$

Proof. Combining Lemmata C.9 and C.10, we get, for every two integers $0 \leq i_s \leq i_T \leq \varepsilon^{-1}$,

$$
\begin{aligned}
F(\mathbf{y}^{(i_T)} \oplus \mathbf{z}^{(i_T)}) &\geq \alpha\varepsilon(i_T-i_s)(1-\varepsilon)^{i_T-2} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - (i_T-i_s) \cdot O(\tfrac{\varepsilon^2\beta D^2}{1-m}) \\
&\quad + (1-\varepsilon)^{i_T-i_s} \cdot \Big[\alpha \cdot g(\varepsilon i_s) - i_s \cdot O(\varepsilon^2(\alpha \cdot F(\mathbf{o}) + \alpha \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) + \tfrac{\beta D^2}{1-m})) \\
&\quad - 3\varepsilon^2 \cdot \sum_{i'=1}^{i_s} F(\mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)})\Big] - 3\varepsilon^2 \cdot \sum_{i'=i_s+1}^{i_T} F(\mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)}) \\
&\geq \alpha\varepsilon(i_T-i_s)(1-\varepsilon)^{i_T-2} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - O(\varepsilon(\alpha \cdot F(\mathbf{o}) + \alpha \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) + \tfrac{\beta D^2}{1-m})) \\
&\quad + (1-\varepsilon)^{i_T-i_s} \cdot \alpha\Big[(e^{-\varepsilon i_s} - e^{-2\varepsilon i_s}) \cdot F(\mathbf{o}) + \frac{(\varepsilon i_s)^2 \cdot e^{-2\varepsilon i_s}}{2} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})\Big]
\end{aligned}
$$

$$
\begin{aligned}
&- 3\varepsilon^2 \cdot \sum_{i'=1}^{i_T} F(\mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)}) \\
&\geq \alpha\varepsilon(i_T - i_s)e^{-\varepsilon i_T} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) + \alpha\left[\frac{(\varepsilon i_s)^2 \cdot e^{-\varepsilon i_s - \varepsilon i_T}}{2} - O(\varepsilon)\right] \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) \\
&\quad + \alpha(e^{-\varepsilon i_T} - e^{-\varepsilon i_s - \varepsilon i_T} - O(\varepsilon)) \cdot F(\mathbf{o}) - 3\varepsilon \max_{1 \leq i' \leq i_T} F(\mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)}) - O(\tfrac{\varepsilon\beta D^2}{1-m}) \enspace .
\end{aligned}
$$

This implies that, for every $t_s \in [0, 1]$ and $T \in [t_s, 1]$,

$$
\begin{aligned}
&(1 + 3\varepsilon) \cdot \max_{0 \leq i \leq \varepsilon^{-1}} F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) \geq F(\mathbf{y}^{(\lfloor \varepsilon^{-1} T \rfloor)} \oplus \mathbf{z}^{(\lfloor \varepsilon^{-1} T \rfloor)}) + 3\varepsilon \cdot \max_{1 \leq i' \leq \lfloor \varepsilon^{-1} T \rfloor} \mathbf{y}^{(i'-1)} \oplus \mathbf{z}^{(i'-1)} \\
&\geq \alpha\varepsilon(\lfloor \varepsilon^{-1} T \rfloor - \lfloor \varepsilon^{-1} t_s \rfloor)e^{-\varepsilon \lfloor \varepsilon^{-1} T \rfloor} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) + \alpha\left[\frac{(\varepsilon \lfloor \varepsilon^{-1} t_s \rfloor)^2 \cdot e^{-\varepsilon \lfloor \varepsilon^{-1} t_s \rfloor - \varepsilon \lfloor \varepsilon^{-1} T \rfloor}}{2} - O(\varepsilon)\right] \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) \\
&\quad + \alpha(e^{-\varepsilon \lfloor \varepsilon^{-1} T \rfloor} - e^{-\varepsilon \lfloor \varepsilon^{-1} t_s \rfloor - \varepsilon \lfloor \varepsilon^{-1} T \rfloor} - O(\varepsilon)) \cdot F(\mathbf{o}) - O(\tfrac{\varepsilon\beta D^2}{1-m}) \\
&\geq \alpha(T - t_s - \varepsilon)e^{-T} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) + \alpha\left[\frac{t_s^2 \cdot e^{-t_s - T}}{2} - O(\varepsilon)\right] \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) \\
&\quad + \alpha(e^{-T} - e^{-t_s - T} - O(\varepsilon)) \cdot F(\mathbf{o}) - O(\tfrac{\varepsilon\beta D^2}{1-m}) \enspace . \qquad \square
\end{aligned}
$$

C.2 Guess-Free Offline Maximization

In this section, we reprove Theorem 4.1 without assuming knowledge of $F(\mathbf{o}_{\mathrm{D}}^{(1)})$ like in Section C.1. Formally, we prove in this section the following proposition.

Proposition C.12. *Let $\mathcal{K}_{\mathrm{N}} \subseteq [0, 1]^n$ be a general solvable convex set, $\mathcal{K}_{\mathrm{D}} \subseteq [0, 1]^n$ be a down-closed solvable convex set, and $F\colon [0, 1]^n \to \mathbb{R}_{\geq 0}$ be a non-negative β-smooth DR-submodular function. Then, there exists a polynomial time algorithm that, given a time $t_s \in [0, 1]$ and an error parameter $\varepsilon \in (0, 1)$, outputs vector $\mathbf{w} \in (\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0, 1]^n$ such that*

$$
\begin{aligned}
F(\mathbf{w}) \geq (1 - m) \cdot \max_{T \in [t_s, 1]} \Bigg\{ &((T - t_s)e^{-T} - O(\varepsilon)) \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) + \left(\frac{t_s^2 \cdot e^{-t_s - T}}{2} - O(\varepsilon)\right) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) \\
&+ (e^{-T} - e^{-t_s - T} - O(\varepsilon)) \cdot F(\mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}^{(1)}) \Bigg\} - O(\tfrac{\varepsilon\beta D^2}{1-m}) \enspace ,
\end{aligned}
$$

where $m = \min_{\mathbf{x} \in \mathcal{K}_{\mathrm{N}}} \|\mathbf{x}\|_\infty$, D is the diameter of $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0, 1]^n$, $\mathbf{o}_{\mathrm{N}} \in \mathcal{K}_{\mathrm{N}}$ and $\mathbf{o}_{\mathrm{D}}^{(1)} \in \mathcal{K}_{\mathrm{D}}$ are any vectors whose sum belongs to $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0, 1]^n$, and $\mathbf{o}_{\mathrm{D}}^{(2)} \in \mathcal{K}_{\mathrm{D}}$.

To see why Proposition C.12 implies Theorem 4.1, we observe that, by executing the algorithm from this proposition for every $t_s \in \{\varepsilon i \mid i \in \mathbb{Z}, 0 \leq i \leq \varepsilon^{-1}\}$, and then outputting the best solution obtained, one can get (in polynomial time) a vector $\mathbf{w}$ such that

$$
\begin{aligned}
F(\mathbf{w}) \geq (1 - m) \cdot \max_{t_s \in \{\varepsilon i \mid i \in \mathbb{Z}, 0 \leq i \leq \varepsilon^{-1}\}} \max_{T \in [t_s, 1]} \Bigg\{ &((T - t_s)e^{-T} - O(\varepsilon)) \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) \\
+ \left(\frac{t_s^2 \cdot e^{-t_s - T}}{2} - O(\varepsilon)\right) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) + (e^{-T} - e^{-t_s - T} - O(\varepsilon)) \cdot F(\mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}^{(1)}) \Bigg\} - O(\tfrac{\varepsilon\beta D^2}{1-m}) \enspace .
\end{aligned}
$$

Since the coefficients of $F(\mathbf{o}_{\mathrm{D}}^{(1)})$, $F(\mathbf{o}_{\mathrm{D}}^{(2)})$ and $F(\mathbf{o}_{\mathrm{N}} + \mathbf{o}_{\mathrm{D}}^{(1)})$ in the right side of the last inequality all change by at most $O(\varepsilon)$ when t_s and T change by at most ε, this right side is equivalent to the right side of the inequality stated in Theorem 4.1. Thus, the vector $\mathbf{w}$ obeys the properties guaranteed by this theorem.

The proof of Proposition C.12 is based on Algorithm 3. The pseudocode of this algorithm makes, without loss of generality, two assumptions, which we list below.

- The pseudocode assumes that ε^{-1} is integral and $\varepsilon \leq 1/30$. If this is not the case, then we can replace ε with $1/\lceil 30\varepsilon^{-1} \rceil$, which decreases ε by most a constant factor.
- The pseudocode assumes that $\varepsilon^{-1} t_s$ is integral. If this is not the case, we reduce t_s by at most ε to $\varepsilon \lfloor \varepsilon^{-1} t_s \rfloor$. Notice that such a reduction again affects the right hand side of the inequality stated in Proposition C.12 only by modifying the hidden constants within the big O notation.

Algorithm 3 `Frank-Wolfe/Continuous-Greedy Hybrid`

```
1: Let $\mathbf{y}^{(0)} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{K}_{\text{N}}}\|\mathbf{x}\|_\infty$, $\mathbf{z}^{(0)} \leftarrow \bar{0}$ and $m \leftarrow \|\mathbf{y}^{(0)}\|_\infty$.
2: for $i = 1$ to $\varepsilon^{-1}$ do
3:     if $i \leq \varepsilon^{-1}t_s$ then
4:         Solve the following linear program. The variables in this program are the vectors $\mathbf{a}^{(i)}$ and $\mathbf{b}^{(i)}$.
5:
           $$\begin{aligned} \text{maximize} \quad & e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + \mathbf{a}^{(i)} \rangle \\ & \quad + (1-m) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle \nabla F(\mathbf{z}^{(i-1)}) \odot (1 - \mathbf{z}^{(i-1)}), \mathbf{b}^{(i)} \rangle \\ \text{subject to} \quad & \mathbf{a}^{(i)} \in \mathcal{K}_{\text{N}}, \mathbf{b}^{(i)} \in \mathcal{K}_{\text{D}} \\ & \mathbf{a}^{(i)} + \mathbf{b}^{(i)} \in [0,1]^n \end{aligned}$$
6:     else
7:         Solve the following linear program. The variables in this program are the vectors $\mathbf{a}^{(i)}$ and $\mathbf{b}^{(i)}$.
8:
           $$\begin{aligned} \text{maximize} \quad & \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \rangle \\ \text{subject to} \quad & \mathbf{a}^{(i)} = \mathbf{y}^{(i)}, \mathbf{b}^{(i)} \in \mathcal{K}_{\text{D}} \end{aligned}$$
9:     end if
10:    Let $\mathbf{y}^{(i)} \leftarrow (1-\varepsilon) \cdot \mathbf{y}^{(i-1)} + \varepsilon \cdot \mathbf{a}^{(i)}$.
11:    Let $\mathbf{z}^{(i)} \leftarrow \mathbf{z}^{(i-1)} + \varepsilon \cdot (1 - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}$.
12: end for
13: Return a vector maximizing $F$ among all the vectors in $\{\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} \mid i \in \mathbb{Z}, \varepsilon^{-1}t_s \leq i \leq \varepsilon^{-1}\}$.
```

Intuitively, Algorithm 1 strives to guarantee that both $F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)})$ and $F(\mathbf{z}^{(i)})$ increase at appropriate rates. Algorithm 3 relaxes this goal, and its aim is to guarantee that some combination of these expressions increases at an appropriate rate. Specifically, the combination considered is given by the potential function $\phi(i) \triangleq e^{2(\varepsilon i - t_s)} \cdot F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) + (1-m)(1-\varepsilon) \cdot e^{\varepsilon i - 2t_s}(t_s - \varepsilon i) \cdot F(\mathbf{z}^{(i)})$. We note that the idea of using such a dynamic combination of the two functions of interest as a potential function can be traced back to the work of [Feldman, 2021].

It is also interesting to note that, for $i > \varepsilon^{-1}t_s$, the variable $\mathbf{a}^{(i)}$ and $\mathbf{y}^{(i)}$ of Algorithm 1 are always set to be equal to $\mathbf{y}^{(i-1)}$. This means that one could simplify the algorithm by dropping the constraint regarding $\mathbf{a}^{(i)}$ from the second linear program of Algorithm 1 and executing Line 10 only for $i \leq \varepsilon^{-1}t_s$. This observation is essential for getting the online result of Section 5. However, for the sake of the current section, the given description of the algorithm is more convenient as it immediately implies the following observation, which shows that the pair $(\mathbf{a}^{(i)}, \mathbf{b}^{(i)})$ always obey all the constraints of the linear program of Algorithm 1, except for one.

Observation C.13. *For every $1 \leq i \leq \varepsilon^{-1}$, $\mathbf{a}^{(i)} \in \mathcal{K}_{\text{N}}$, $\mathbf{b}^{(i)} \in \mathcal{K}_{\text{D}}$ and $\mathbf{a}^{(i)} + (1 - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} \leq \bar{1}$.*

Given Observation C.13, the proofs of Lemmata C.1 and C.2 go through without a change. Thus, we can use both lemmata in the analysis of Algorithm 3, which in particular, implies that the output vector of this algorithm belongs to $(\mathcal{K}_{\text{N}} + \mathcal{K}_{\text{D}}) \cap [0,1]^n$.

Our objective in the rest of this section is to lower bound the value of this output vector. We begin with a basic lower bound on the rate in which the potential function $\phi(i)$ increases.

Lemma C.14. *For every integer $1 \leq i \leq \varepsilon^{-1}t_s$, the expression $\varepsilon^{-1}e^{2t_s}[\phi(i) - \phi(i-1)]$ can be lower bounded both by $2e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) - \frac{25\varepsilon\beta D^2}{1-m} - 15\varepsilon \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})$ and by the sum of this expression and*

$$\begin{aligned} (1-\varepsilon) \cdot e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{o}_{\text{D}}^{(1)} + \mathbf{o}_{\text{N}} - \mathbf{y}^{(i-1)} \rangle \\ + (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{o}_{\text{D}}^{(1)}, \nabla F(\mathbf{z}^{(i-1)}) \rangle \enspace . \end{aligned}$$

Proof. Observe that

$$\begin{aligned} & e^{2\varepsilon i} \cdot F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \qquad (6) \\ & \quad = [e^{2\varepsilon i} - e^{2\varepsilon(i-1)}] \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + e^{2\varepsilon i} \cdot [F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})] \\ & \quad \geq \int_0^\varepsilon \{2e^{2(\varepsilon(i-1)+\tau)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \end{aligned}$$

$$(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - 2\tau \cdot \mathbf{b}^{(i)})\rangle - 45\tau\beta D^2\}d\tau$$
$$\geq 2\varepsilon e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + \varepsilon e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}),$$
$$(\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \varepsilon \cdot \mathbf{b}^{(i)})\rangle - 23\varepsilon^2\beta D^2 \enspace ,$$

where the first inequality follows from the calculations done in the first half of the proof of Lemma C.8, and the second inequality uses the non-negativity of F. Similarly,

$$\begin{aligned}
&e^{\varepsilon i}(t_s - \varepsilon i) \cdot F(\mathbf{z}^{(i)}) - e^{\varepsilon(i-1)}(t_s - \varepsilon(i-1)) \cdot F(\mathbf{z}^{(i-1)}) \qquad (7)\\
&\quad = [e^{\varepsilon i}(t_s - \varepsilon i) - e^{\varepsilon(i-1)}(t_s - \varepsilon(i-1))] \cdot F(\mathbf{z}^{(i-1)}) + e^{\varepsilon i}(t_s - \varepsilon i) \cdot [F(\mathbf{z}^{(i)}) - F(\mathbf{z}^{(i-1)})]\\
&\quad \geq \int_0^{\varepsilon} \{e^{\varepsilon(i-1)+\tau}(t_s - 1 - \varepsilon(i-1) - \tau) \cdot F(\mathbf{z}^{(i-1)}) + e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle\\
&\qquad\qquad - 4\tau\beta D^2/(1-m)^2\}d\tau\\
&\quad \geq -\varepsilon e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) + \varepsilon e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle\\
&\qquad\qquad - 2\varepsilon^2\beta D^2/(1-m)^2 \enspace ,
\end{aligned}$$

where the first inequality is based on calculations from the proof of Lemma C.5, and the second inequality uses F's non-negativity. Adding up Inequality (6) and $(1-m)(1-\varepsilon)$ times Inequality (7), we get

$$\begin{aligned}
&\varepsilon^{-1}e^{2t_s}[\phi(i) - \phi(i-1)] \qquad (8)\\
&\quad \geq 2e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) - \tfrac{25\varepsilon\beta D^2}{1-m}\\
&\qquad + (1-\varepsilon) \cdot e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + \mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}\rangle\\
&\qquad + (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle\\
&\qquad + \varepsilon e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{b}^{(i)})\rangle \enspace .
\end{aligned}$$

Note now that Properties 1 and 2 of Lemma B.1 imply together

$$\begin{aligned}
&\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{b}^{(i)})\rangle\\
&\quad = \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{a}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{b}^{(i)})\rangle\\
&\qquad - \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{a}^{(i)}) \odot (\bar{1} - \mathbf{b}^{(i)})\rangle\\
&\quad \geq F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} \oplus (\mathbf{a}^{(i)} \odot (\bar{1} - \mathbf{b}^{(i)}))) + F((\mathbf{y}^{(i-1)} \odot (\mathbf{a}^{(i)} \oplus \mathbf{b}^{(i)})) \oplus \mathbf{z}^{(i-1)})\\
&\qquad - 2 \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \geq -2 \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace ,
\end{aligned}$$

where the last inequality holds by the non-negativity of F. Plugging this inequality into Inequality (8) yields

$$\begin{aligned}
&\varepsilon^{-1}e^{2t_s}[\phi(i) - \phi(i-1)]\\
&\quad \geq 2e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) - \tfrac{25\varepsilon\beta D^2}{1-m}\\
&\qquad + (1-\varepsilon) \cdot e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)} + \mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}\rangle\\
&\qquad + (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle - 15\varepsilon \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace .
\end{aligned}$$

The last two terms on the right hand side of the last inequality are related to the objective function of the first linear program of Algorithm 3. Specifically, to get from them to the objective function of this linear program, it is necessary to multiply by $(1-\varepsilon)^{-1}$, and then remove the additive term $e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), -\mathbf{y}^{(i-1)}\rangle$, which does not depend on $\mathbf{a}^{(i)}$ and $\mathbf{b}^{(i)}$. Therefore, we can lower bound the right hand side of the last inequality by plugging in feasible solutions for the first linear program of Algorithm 3. Specifically, we plug in the solutions $(\mathbf{a}^{(i)}, \mathbf{b}^{(i)}) = (\mathbf{o}_{\mathrm{N}}, \mathbf{o}_{\mathrm{D}}^{(1)})$ and $(\mathbf{y}^{(i-1)}, \bar{0})$, which implies the two lower bounds stated in the lemma. Notice that both these solutions are guaranteed to be feasible since $\mathcal{K}_{\mathrm{D}}$ is down-closed and $\mathbf{y}^{(i-1)} + \bar{0} = \mathbf{y}^{(i-1)} \leq \bar{1}$. □

We now need to develop the basic lower bound given by Lemma C.14 in two ways. First, we show in the next corollary that the potential cannot decrease significantly. Then, in Lemma C.16, we show a lower bound on the increase of the potential in terms of $F(\mathbf{o})$ and $F(\mathbf{o}_{\mathrm{D}}^{(1)})$.

Corollary C.15. *For every integer $1 \leq i \leq \varepsilon^{-1}t_s$,*

$$\phi(i) - \phi(i-1) \geq -\tfrac{25\varepsilon^2\beta D^2}{1-m} - 15\varepsilon^2 \cdot \phi(i-1) \enspace .$$

Proof. By Corollary B.3 and Lemma C.2,

$$2e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \geq 2(1-m) \cdot e^{2\varepsilon(i-1)} \cdot (1-\varepsilon)^{i-1} \cdot F(\mathbf{z}^{(i-1)})$$
$$\geq (1-m) \cdot e^{\varepsilon(i-1)} \cdot F(\mathbf{z}^{(i-1)}) \geq (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1-t_s+\varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) \enspace ,$$

where the second inequality follows from our assumption that $\varepsilon \leq 1/30$. By plugging the last inequality into Lemma C.14, we get

$$\varepsilon^{-1} e^{2t_s}[\phi(i) - \phi(i-1)] \geq -\tfrac{25\varepsilon\beta D^2}{1-m} - 15\varepsilon \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace .$$

The corollary follows from this inequality since the non-negativity of F implies that

$$\phi(i-1) \geq e^{2(\varepsilon(i-1)-t_s)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \geq e^{-2t_s} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace . \qquad \square$$

Lemma C.16. *For every integer* $1 \leq i \leq \varepsilon^{-1} t_s$,

$$\phi(i) - \phi(i-1) \geq \varepsilon(1-m)(1-\varepsilon) \cdot [e^{\varepsilon i - 2t_s} \cdot F(\mathbf{o}) + e^{-2t_s}(t_s - \varepsilon i) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})] - \tfrac{25\varepsilon^2\beta D^2}{1-m} - 62\varepsilon^2 \cdot \phi(i-1) \enspace .$$

Proof. Repeating some of the calculations from the proof of Lemma C.4, we get

$$\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{o}_{\mathrm{D}}^{(1)} + (\mathbf{o}_{\mathrm{N}} - \mathbf{y}^{(i-1)})\rangle$$
$$\geq (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{o}) + (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{z}^{(i-1)}) - 2F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace .$$

Additionally, by repeating some of the calculations from the proof of Lemma C.3, we get

$$\langle \mathbf{o}_{\mathrm{D}}^{(1)} \odot (\bar{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{z}^{(i-1)})\rangle \geq (1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)}) \enspace .$$

Plugging both these inequalities into the guarantee of Lemma C.14, we get that $\varepsilon^{-1} e^{2t_s}[\phi(i) - \phi(i-1)]$ is at least

$$2e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1-t_s+\varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) - \tfrac{25\varepsilon\beta D^2}{1-m}$$
$$+ (1-\varepsilon) \cdot e^{2\varepsilon i} \cdot [(1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{o}) + (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{z}^{(i-1)}) - 2F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})]$$
$$+ (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot [(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) - F(\mathbf{z}^{(i-1)})] - 15\varepsilon \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})$$
$$\geq (1-m)(1-\varepsilon) \cdot [e^{\varepsilon i} \cdot F(\mathbf{o}) + (t_s - \varepsilon i) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})] - \tfrac{25\varepsilon\beta D^2}{1-m} - 62\varepsilon \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})$$
$$\geq (1-m)(1-\varepsilon) \cdot [e^{\varepsilon i} \cdot F(\mathbf{o}) + (t_s - \varepsilon i) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})] - \tfrac{25\varepsilon\beta D^2}{1-m} - 62\varepsilon \cdot e^{2t_s} \cdot \phi(i-1) \enspace . \qquad \square$$

The last two claims provide lower bounds on the change in ϕ as a function of i. We now use these bounds to get a lower bound on $F(\mathbf{y}^{(\varepsilon^{-1}t_s)} \oplus \mathbf{z}^{(\varepsilon^{-1}t_s)}) = \phi(\varepsilon^{-1}t_s)$.

Lemma C.17. *It holds that*

$$F(\mathbf{y}^{(\varepsilon^{-1}t_s)} \oplus \mathbf{z}^{(\varepsilon^{-1}t_s)}) = \phi(\varepsilon^{-1}t_s)$$
$$\geq \frac{(1-m)(1-\varepsilon) \cdot \{(e^{-t_s} - e^{-2t_s}) \cdot F(\mathbf{o}) + e^{-2t_s} \cdot \frac{t_s^2 - \varepsilon}{2} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})\} - t_s \cdot O(\frac{\varepsilon\beta D^2}{1-m})}{1 + O(\varepsilon)} \enspace .$$

Proof. Summing up the guarantee of Lemma C.16 for every integer $1 \leq i \leq \varepsilon^{-1}t_s$ yields

$$\phi(\varepsilon^{-1}t_s) \geq \phi(0) + \varepsilon(1-m)(1-\varepsilon) \cdot \sum_{i=1}^{\varepsilon^{-1}t_s} [e^{\varepsilon i - 2t_s} \cdot F(\mathbf{o}) + e^{-2t_s}(t_s - \varepsilon i) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)})] \tag{9}$$
$$- \tfrac{25\varepsilon t_s \beta D^2}{1-m} - 62\varepsilon^2 \cdot \sum_{i=1}^{\varepsilon^{-1}t_s} \phi(i-1) \enspace .$$

Let us now bound some of the terms in the last inequality. First,

$$\varepsilon \cdot \sum_{i=1}^{\varepsilon^{-1}t_s} e^{\varepsilon i - 2t_s} \cdot F(\mathbf{o}) \geq \int_0^{t_s} e^{\tau - 2t_s} d\tau \cdot F(\mathbf{o}) = e^{\tau - 2t_s}|_0^{t_s} \cdot F(\mathbf{o}) = (e^{-t_s} - e^{-2t_s}) \cdot F(\mathbf{o}) \enspace .$$

Second,

$$\varepsilon \cdot \sum_{i=1}^{\varepsilon^{-1}t_s} e^{-2t_s}(t_s - \varepsilon i) \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) = e^{-2t_s}\left[t_s^2 - \varepsilon^2 \cdot \frac{(\varepsilon^{-1}t_s)(\varepsilon^{-1}t_s + 1)}{2}\right] \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) \geq e^{-2t_s} \cdot \frac{t_s^2 - \varepsilon}{2} \cdot F(\mathbf{o}_{\mathrm{D}}^{(1)}) \enspace .$$

Finally, if we denote by i^* the value of i for which the maximum of $\max_{0 \leq i \leq \varepsilon^{-1} t_s} \phi(i)$ is obtained, then, by Corollary C.15,

$$\phi(i^*) \leq \phi(\varepsilon^{-1} t_s) + \tfrac{25 t_s \varepsilon \beta D^2}{1-m} + 15\varepsilon^2 \cdot \sum_{i=i^*+1}^{\varepsilon^{-1} t_s} \phi(i-1) \leq \phi(\varepsilon^{-1} t_s) + \tfrac{25 t_s \varepsilon \beta D^2}{1-m} + 15\varepsilon \cdot \phi(i^*) \enspace ,$$

and thus,

$$\begin{aligned}\varepsilon \cdot \sum_{i=1}^{\varepsilon^{-1} t_s} \phi(i-1) - \tfrac{50 t_s \varepsilon \beta D^2}{1-m} &\leq \phi(i^*) - \tfrac{50 t_s \varepsilon \beta D^2}{1-m} \\ &\leq \frac{(1-15\varepsilon) \cdot \phi(i^*) - \frac{25 t_s \varepsilon \beta D^2}{1-m}}{1-15\varepsilon} \leq \frac{\phi(\varepsilon^{-1} t_s)}{1-15\varepsilon} \leq 2 \cdot \phi(\varepsilon^{-1} t_s) \enspace ,\end{aligned}$$

where the second and last inequalities follows from our assumption that $\varepsilon \leq 1/30$. The lemma now follows by plugging all the above bounds into Inequality (9), and observing that $\phi(0) \geq 0$ due to the non-negativity of F. $\square$

Up to this point we have only considered the first $\varepsilon^{-1} t_s$ iterations of Algorithm 3. We now need to lower bound the rate in which $F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)})$ increases in the remaining iterations of the algorithm.

Lemma C.18. *For every integer $\varepsilon^{-1} t_s < i \leq \varepsilon^{-1}$,*

$$F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \geq \varepsilon \cdot [(1-m)(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})] - \varepsilon^2 \beta D^2/2 \enspace .$$

Proof. By the chain rule,

$$\begin{aligned}F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) &= \int_0^\varepsilon \frac{dF(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \tau \cdot \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)})))}{d\tau} d\tau \qquad (10) \\ &= \int_0^\varepsilon \langle \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}), \nabla F(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \tau \cdot \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)})))\rangle d\tau \enspace .\end{aligned}$$

We would like to lower bound the integrand on the rightmost side of the last equality. We do that by lower bounding two expressions whose sum is equal to this integrand. The first expression is

$$\begin{aligned}&\langle \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}), \\ &\qquad \nabla F(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \tau \cdot \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}))) - \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle \\ \geq{}& -\|\mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2 \\ &\qquad \cdot \|\nabla F(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \tau \cdot \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}))) - \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\|_2 \\ \geq{}& -\tau\beta \|\mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2^2 \geq -\tau\beta D^2 \enspace ,\end{aligned}$$

where the first inequality holds by the Cauchy–Schwarz inequality, the second inequality holds due to the β-smoothness of F, and the last inequality holds since

$$\|\mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2 \leq \|\mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2 = \|(\mathbf{y}^{(i-1)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})) - \mathbf{y}^{(i-1)}\|_2 \leq D$$

because both $\mathbf{y}^{(i-1)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})$ and $\mathbf{y}^{(i-1)}$ are vectors of $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$. The second expression is

$$\begin{aligned}\langle \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}), &\nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\rangle \\ &= \langle \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}), \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle \\ &\geq \langle \mathbf{o}_{\mathrm{D}}^{(2)} \odot (\bar{1} - \mathbf{y}^{(i-1)}), \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\rangle \\ &\geq F(\mathbf{o}_{\mathrm{D}}^{(2)} \oplus \mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \\ &\geq (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace ,\end{aligned}$$

where the first inequality holds since $\mathbf{b}^{(i)} = \mathbf{o}_{\mathrm{D}}^{(2)}$ is a feasible solution for the second linear program of Algorithm 3, the second inequality follows from Property (1) of Lemma B.1, and the last inequality follows from Corollary B.3 and Lemma C.2. The lemma now follows by plugging the lower bounds we have proved into the rightmost side of Equality (10), and then solving the integral obtained. $\square$

Using the lower bound proved in the last lemma on the rate in which $F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)})$ increases as a function of i for $i \geq \varepsilon^{-1}t_s$, we derive in the next lemma a lower bound on the value of this expression for particular values of i. The proof of this lower bound is very similar to the proof of Lemma C.10.

Lemma C.19. *For every integer $\varepsilon^{-1}t_s \leq i \leq \varepsilon^{-1}$, $F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) \geq (1-m)(\varepsilon i - t_s)(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\text{D}}^{(2)}) + (1-\varepsilon)^{i-\varepsilon^{-1}t_s} \cdot F(\mathbf{y}^{(\varepsilon^{-1}t_s)} \oplus \mathbf{z}^{(\varepsilon^{-1}t_s)}) - (i - \varepsilon^{-1}t_s) \cdot O(\varepsilon^2\beta D^2)$.*

Proof. We prove the lemma by induction on i. For $i = \varepsilon^{-1}t_s$ the lemma trivially holds. Thus, assume that the lemma holds for $i-1$, and let us prove it for $i > \varepsilon t_s$. By Lemma C.18,

$$\begin{aligned}
F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) &\geq (1-\varepsilon) \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + \varepsilon(1-m)(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\text{D}}^{(2)}) - O(\varepsilon^2\beta D^2)\\
&\geq (1-\varepsilon) \cdot [(1-m)(\varepsilon i - \varepsilon - t_s)(1-\varepsilon)^{i-2} \cdot F(\mathbf{o}_{\text{D}}^{(2)})\\
&\qquad\quad + (1-\varepsilon)^{i-1-\varepsilon^{-1}t_s} \cdot F(\mathbf{y}^{(\varepsilon^{-1}t_s)} \oplus \mathbf{z}^{(\varepsilon^{-1}t_s)}) - (i - 1 - \varepsilon^{-1}t_s) \cdot O(\varepsilon^2\beta D^2)]\\
&\qquad\quad + \varepsilon(1-m)(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\text{D}}^{(2)}) - O(\varepsilon^2\beta D^2)\\
&\geq (1-m)(\varepsilon i - t_s)(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\text{D}}^{(2)}) + (1-\varepsilon)^{i-\varepsilon^{-1}t_s} \cdot F(\mathbf{y}^{(\varepsilon^{-1}t_s)} \oplus \mathbf{z}^{(\varepsilon^{-1}t_s)})\\
&\qquad\quad - (i - \varepsilon^{-1}t_s) \cdot O(\varepsilon^2\beta D^2) \enspace,
\end{aligned}$$

where the second inequality holds by the induction hypothesis. □

We are now ready to complete the proof of the approximation guarantee of Algorithm 3, which completes the proof of Proposition C.12 (recall that the output of Algorithm 3 has a value of $\max_{\varepsilon^{-1}t_s \leq i \leq \varepsilon^{-1}} F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)})$).

Lemma C.20. *For every value $T \in [t_s, 1]$, it holds that*

$$\begin{aligned}
\max_{\varepsilon^{-1}t_s \leq i \leq \varepsilon^{-1}} F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) \geq F(\mathbf{y}^{(\lfloor \varepsilon^{-1}T \rfloor)} \oplus \mathbf{z}^{(\lfloor \varepsilon^{-1}T \rfloor)}) \geq \frac{(1-m)}{1+42\varepsilon} \cdot \Big\{ &(e^{-T} - e^{-t_s-T} - 2\varepsilon) \cdot F(\mathbf{o})\\
&+ (T - t_s - \varepsilon) \cdot e^{-T} \cdot F(\mathbf{o}_{\text{D}}^{(2)}) + \left[\frac{t_s^2 \cdot e^{-t_s-T}}{2} - \varepsilon\right] \cdot F(\mathbf{o}_{\text{D}}^{(1)}) \Big\} - O(\tfrac{\varepsilon\beta D^2}{1-m}) \enspace.
\end{aligned}$$

Proof. Plugging the guarantee of Lemma C.17 into the guarantee of Lemma C.19 yields

$$\begin{aligned}
&F(\mathbf{y}^{(\lfloor \varepsilon^{-1}T \rfloor)} \oplus \mathbf{z}^{(\lfloor \varepsilon^{-1}T \rfloor)}) \geq (1-m)(\varepsilon\lfloor \varepsilon^{-1}T \rfloor - t_s)(1-\varepsilon)^{\varepsilon\lfloor \varepsilon^{-1}T \rfloor - 1} \cdot F(\mathbf{o}_{\text{D}}^{(2)})\\
&\quad - (\varepsilon\lfloor \varepsilon^{-1}T \rfloor - t_s) \cdot O(\varepsilon\beta D^2) + (1-\varepsilon)^{\varepsilon^{-1}\lfloor \varepsilon^{-1}T \rfloor - \varepsilon^{-1}t_s} \cdot F(\mathbf{y}^{(\varepsilon^{-1}t_s)} \oplus \mathbf{z}^{(\varepsilon^{-1}t_s)})\\
&\geq (1-m)(T - t_s - \varepsilon) \cdot e^{-T} \cdot F(\mathbf{o}_{\text{D}}^{(2)}) - (T - t_s) \cdot O(\varepsilon\beta D^2)\\
&\quad + e^{t_s - T}(1-\varepsilon)\left[\frac{(1-m)(1-\varepsilon)\{(e^{-t_s} - e^{-2t_s}) \cdot F(\mathbf{o}) + e^{-2t_s} \cdot \frac{t_s^2 - \varepsilon}{2} \cdot F(\mathbf{o}_{\text{D}}^{(1)})\} - t_s \cdot O(\frac{\varepsilon\beta D^2}{1-m})}{1 + O(\varepsilon)}\right]\\
&\geq \frac{(1-m)(1-2\varepsilon)}{1+O(\varepsilon)} \cdot \Big\{ (e^{-T} - e^{-t_s-T}) \cdot F(\mathbf{o})\\
&\qquad\qquad + (T - t_s - \varepsilon) \cdot e^{-T} \cdot F(\mathbf{o}_{\text{D}}^{(2)}) + \left[\frac{t_s^2 \cdot e^{-t_s-T}}{2} - \varepsilon\right] \cdot F(\mathbf{o}_{\text{D}}^{(1)}) \Big\} - O(\tfrac{\varepsilon\beta D^2}{1-m}) \enspace. \qquad \square
\end{aligned}$$

C.3 Empirical Improvements of Algorithm 3

In this section, we present a version of Algorithm 3 (appears as Algorithm 4) that includes modifications designed to improve the empirical performance of the algorithm. These modifications were not included in the original Algorithm 3 because they do not affect the algorithm's theoretical guarantee, and furthermore, they cannot be applied to the online counterpart of Algorithm 3 (i.e., Algorithm 2).

To be more concrete, the modifications done in Algorithm 4 compared to Algorithm 3 are the following.

- Algorithm 3 was designed with the view that $\mathbf{o}_{\text{D}}^{(1)}$ and $\mathbf{o}_{\text{D}}^{(2)}$ should be a feasible assignments to $\mathbf{b}^{(i)}$, and then the vector $\mathbf{z}^{(i)}$ is increased proportionally to $(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{b}^{(i)}$. The multiplication by $(1 - \mathbf{z}^{(i-1)})$ was done to reduce the speed in which $\|\mathbf{z}^{(i)}\|_\infty$ increases as a function of i. Following the work of Bian et al. [2017a], Algorithm 4 was modified to make $(\bar{1} - \mathbf{z}^{(i-1)}) \cdot \mathbf{o}_{\text{D}}^{(1)}$ and $(\bar{1} - \mathbf{z}^{(i-1)}) \cdot \mathbf{o}_{\text{D}}^{(2)}$ natural assignments for $\mathbf{b}^{(i)}$. This was done by allowing $\mathbf{b}^{(i)}$ to be any vector in $\mathcal{K}_{\text{D}}$ which is upper bounded by $(\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{a}^{(i)})$, and making the vector $\mathbf{z}^{(i)}$ increase proportionally to the vector $\mathbf{b}^{(i)}$ itself. Note that these modifications preserve the bound on the rate in which $\|\mathbf{z}^{(i)}\|_\infty$ increases, but give the algorithm more flexability as the increase in $\mathbf{z}^{(i)}$ no longer has to be equal to $\bar{1} - \mathbf{z}^{(i-1)}$ times some vector in $\mathcal{K}_{\text{D}}$.

Algorithm 4 `Frank-Wolfe/Continuous-Greedy Hybrid with Empirical Improvements`

```
1: Let $\mathbf{y}^{(0)} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{K}_{\mathrm{N}}} \|\mathbf{x}\|_\infty$, $\mathbf{z}^{(0)} \leftarrow \bar{0}$ and $m \leftarrow \|\mathbf{y}^{(0)}\|_\infty$.
2: for $i = 1$ to $\varepsilon^{-1}$ do
3:   if $i \leq \varepsilon^{-1} t_s$ then
4:     Solve the following linear program. The variables in this program are the vectors $\mathbf{a}^{(i)}$, $\mathbf{b}^{(i)}$ and $\mathbf{c}^{(i)}$.
```

$$\begin{aligned}\max \quad & e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \\ & \quad + (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{a}^{(i)} \rangle + (1 - m) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle \nabla F(\mathbf{z}^{(i-1)}), \mathbf{b}^{(i)} - \mathbf{c}^{(i)} \rangle \\ \text{s.t.} \quad & \mathbf{a}^{(i)} \in \mathcal{K}_{\mathrm{N}}, \mathbf{b}^{(i)} \in \mathcal{K}_{\mathrm{D}} \\ & \mathbf{b}^{(i)} \leq (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{a}^{(i)}) \\ & \bar{0} \leq \mathbf{c}^{(i)} \leq \mathbf{z}^{(i-1)}\end{aligned}$$

```
5:   else
6:     Solve the following linear program. The variables in this program are the vectors $\mathbf{a}^{(i)}$, $\mathbf{b}^{(i)}$ and $\mathbf{c}^{(i)}$.
```

$$\begin{aligned}\max \quad & \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}), \mathbf{b}^{(i)} - \mathbf{c}^{(i)} \rangle \\ \text{s.t.} \quad & \mathbf{a}^{(i)} = \mathbf{y}^{(i)}, \mathbf{b}^{(i)} \in \mathcal{K}_{\mathrm{D}} \\ & \mathbf{b}^{(i)} \leq \bar{1} - \mathbf{z}^{(i-1)} \\ & \bar{0} \leq \mathbf{c}^{(i)} \leq \mathbf{z}^{(i-1)}\end{aligned}$$

```
7:     Let $\mathbf{y}^{(i)} \leftarrow (1 - \varepsilon) \cdot \mathbf{y}^{(i-1)} + \varepsilon \cdot \mathbf{a}^{(i)}$.
8:     Let $\mathbf{z}^{(i)} \leftarrow \mathbf{z}^{(i-1)} + \varepsilon \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})$.
9:   end if
10: end for
11: return a vector maximizing $F$ among all the vectors in $\{\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} \mid i \in \mathbb{Z}, 0 \leq i \leq \varepsilon^{-1}\}$.
```

- Algorithm 3 can both increase and decrease $\mathbf{y}^{(i)}$. However, a similar flexibility does not exist for the vector $\mathbf{z}^{(i)}$, which is maintained in a continuous-greedy like fashion. Algorithm 4 introduces a new vector $\mathbf{c}^{(i)}$ that is used to decrease coordinates of $\mathbf{z}^{(i)}$ when this helps the objective (or the potential function when $i \leq \varepsilon^{-1} t_s$).
- Algorithm 3 returns the best solution for any $\varepsilon^{-1} \cdot t_s \leq i \leq \varepsilon^{-1}$. Algorithm 4 returns the best solution obtained after any number of iterations because that is always at least as good.

The rest of this section is devoted to proving that, like Algorithm 3, Algorithm 4 has the properties guaranteed by Theorem 4.1. We begin with the following lemma, which is a counterpart of Lemma C.1, and proves that Algorithm 4 returns a feasible solution.

Lemma C.21. *For every integer* $0 \leq i \leq \varepsilon^{-1}$, $\mathbf{y}^{(i)} \in \mathcal{K}_{\mathrm{N}}$ *and* $\mathbf{z}^{(i)} \in \varepsilon i \cdot \mathcal{K}_{\mathrm{D}}$. *Hence,* $\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)} \in (\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0, 1]^n$.

Proof. Given the proof of Lemma C.1, to prove the current lemma we only need to show that $\mathbf{z}^{(i)} \in \varepsilon i \cdot \mathcal{K}_{\mathrm{D}}$, which we do by induction on i. Clearly, $\mathbf{z}^{(0)} = \bar{0} \in 0 \cdot \mathcal{K}_{\mathrm{D}}$. Assume now that $\mathbf{z}^{(i-1)} \in \varepsilon(i-1) \cdot \mathcal{K}_{\mathrm{D}}$ for some $i \in [\varepsilon^{-1}]$, and let us prove that $\mathbf{z}^{(i)} \in \varepsilon i \cdot \mathcal{K}_{\mathrm{D}}$.

Since $\mathbf{z}^{(i-1)} \in \varepsilon(i-1) \cdot \mathcal{K}_{\mathrm{D}}$, there must exist a vector $\mathbf{x} \in \mathcal{K}_{\mathrm{D}}$ such that $\mathbf{z}^{(i-1)} = \varepsilon(i-1) \cdot \mathbf{x}$. Therefore,

$$\mathbf{z}^{(i)} = \mathbf{z}^{(i-1)} + \varepsilon(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) = \varepsilon i \cdot \left(\frac{i-1}{i} \cdot \mathbf{x} + \frac{1}{i} \cdot \mathbf{b}^{(i)} - \frac{1}{i} \cdot \mathbf{c}^{(i)} \right) \quad . \tag{11}$$

The convexity of $\mathcal{K}_{\mathrm{D}}$ implies that $\frac{i-1}{i} \cdot \mathbf{x} + \frac{1}{i} \cdot \mathbf{b}^{(i)}$ is a vector in $\mathcal{K}_{\mathrm{D}}$. This vector upper bounds $\frac{1}{i} \cdot \mathbf{c}^{(i)}$ because

$$\frac{1}{i} \cdot \mathbf{c}^{(i)} \leq \frac{1}{i} \cdot \mathbf{z}^{(i-1)} = \frac{\varepsilon(i-1)}{i} \cdot \mathbf{x} \leq \frac{i-1}{i} \cdot \mathbf{x} \quad .$$

Therefore, Equation (11) and the down-closeness of $\mathcal{K}_{\mathrm{D}}$ imply together that $\mathbf{z}^{(i)} \in \varepsilon i \cdot \mathcal{K}_{\mathrm{D}}$. □

We now divert our attention to proving the approximation guarantee of Algorithm 4. It turns out that most of the analysis of the approximation guarantee of Algorithm 3 from Section C.2 can be reused to prove the same guarantee for Algorithm 4, except for the proofs of three lemmata: Lemma C.2, Lemma C.14 and Lemma C.18, which we reprove below in the context of this algorithm. We begin by proving that Lemma C.2 still holds in the context of Algorithm 4.

Lemma C.2. *For every integer $0 \leq i \leq \varepsilon^{-1}$, $\|\mathbf{z}^{(i)}\|_\infty \leq 1 - (1-\varepsilon)^i$ and $\|\mathbf{y}^{(i)}\|_\infty \leq \|\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}\|_\infty \leq 1 - (1-\varepsilon)^i(1-m)$.*

Proof. We prove the lemma by induction on i. For $i = 0$, we have $\|\mathbf{z}^{(0)}\|_\infty = \|\bar{0}\|_\infty = 0 = 1-(1-\varepsilon)^0$ and $\|\mathbf{y}^{(0)} \oplus \mathbf{z}^{(0)}\|_\infty = \|\mathbf{y}^{(0)}\|_\infty = m = 1 - (1-\varepsilon)^0(1-m)$. Assume now that $\|\mathbf{z}^{(i-1)}\|_\infty \leq 1-(1-\varepsilon)^{i-1}$ and $\|\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}\|_\infty \leq 1-(1-\varepsilon)^{i-1}(1-m)$ for some $i \in [\varepsilon^{-1}]$, and let us prove the corresponding claim for i.

Observe that

$$\begin{aligned}\|\mathbf{z}^{(i)}\|_\infty &= \|\mathbf{z}^{(i-1)} + \varepsilon \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})\|_\infty \leq \|\mathbf{z}^{(i-1)} + \varepsilon \cdot \mathbf{b}^{(i)}\|_\infty \\ &\leq \|\mathbf{z}^{(i-1)} + \varepsilon \cdot (\bar{1} - \mathbf{z}^{(i-1)})\|_\infty \leq \|(1-\varepsilon) \cdot \mathbf{z}^{(i-1)}\|_\infty + \varepsilon \cdot \|\bar{1}\|_\infty = (1-\varepsilon) \cdot \|\mathbf{z}^{(i-1)}\|_\infty + \varepsilon \\ &\leq (1-\varepsilon) \cdot [1-(1-\varepsilon)^{i-1}] + \varepsilon = 1-(1-\varepsilon)^i \enspace ,\end{aligned}$$

where the first inequality holds since $\mathbf{c}^{(i)}$ is non-negative, the second inequality holds since both linear programs of Algorithm 4 require $\mathbf{b}^{(i)}$ to be coordinate-wise upper bounded by $(\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{a}^{(i)}) \leq \bar{1} - \mathbf{z}^{(i-1)}$, and the last inequality follows from the induction hypothesis.

Similarly,

$$\begin{aligned}\|\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}\|_\infty &= \|((1-\varepsilon) \cdot \mathbf{y}^{(i-1)} + \varepsilon \cdot \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \varepsilon \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))\|_\infty \\ &\leq \|((1-\varepsilon) \cdot \mathbf{y}^{(i-1)} + \varepsilon \cdot \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \varepsilon \cdot \mathbf{b}^{(i)})\|_\infty \\ &\leq \|((1-\varepsilon) \cdot \mathbf{y}^{(i-1)} + \varepsilon \cdot \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \varepsilon \cdot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{a}^{(i)}))\|_\infty \\ &\leq \|(\mathbf{y}^{(i-1)} \oplus (\varepsilon \cdot \mathbf{a}^{(i)})) \oplus (\mathbf{z}^{(i-1)} \oplus (\varepsilon(\bar{1} - \mathbf{a}^{(i)})))\|_\infty \leq \|\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} \oplus (\varepsilon \cdot \bar{1})\|_\infty \\ &= \|\varepsilon \cdot \bar{1}\|_\infty + (1-\varepsilon) \cdot \|\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}\|_\infty \\ &\leq \varepsilon + (1-\varepsilon) \cdot [1-(1-\varepsilon)^{i-1}(1-m)] = 1-(1-\varepsilon)^i(1-m) \enspace . \qquad \square\end{aligned}$$

As the proof that Lemma C.14 applies to Algorithm 4 is somewhat long, we defer it a bit, and prove first that Lemma C.18 applies in the context of this algorithm. To be completely honest, we prove here a modified version of Lemma C.18 that has a slightly larger error term ($\varepsilon^2\beta D^2$ instead of $\varepsilon^2\beta D^2/2$). However, this affects the proof of Theorem 4.1 only by changing the constants hidden by the big O notation.

Lemma C.18. *For every integer $\varepsilon^{-1}t_s < i \leq \varepsilon^{-1}$,*

$$F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \geq \varepsilon \cdot [(1-m)(1-\varepsilon)^{i-1} \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})] - \varepsilon^2\beta D^2 \enspace .$$

Proof. By the chain rule,

$$\begin{aligned}F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) &= \int_0^\varepsilon \frac{dF(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})))}{d\tau} d\tau \qquad (12) \\ &= \int_0^\varepsilon \langle (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}), \nabla F(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))) \rangle d\tau \enspace .\end{aligned}$$

We would like to lower bound the integrand on the rightmost side of the last equality. We do that by lower bounding two expressions whose sum is equal to this integrand. The first expression is

$$\begin{aligned}&\langle (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}), \\ &\qquad \nabla F(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))) - \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \rangle \\ \geq{}& -\|(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2 \\ &\qquad \cdot \|\nabla F(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))) - \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\|_2 \\ \geq{}& -\tau\beta\|(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2^2 \geq -2\tau\beta D^2 \enspace ,\end{aligned}$$

where the first inequality holds by the Cauchy–Schwarz inequality, the second inequality holds due to the β-smoothness of F, and the last inequality holds since

$$\begin{aligned}&\|(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2 \leq \|\mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2 + \|\mathbf{c}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2 \\ &\quad = \|(\mathbf{y}^{(i-1)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})) - \mathbf{y}^{(i-1)}\|_2 + \|(\mathbf{y}^{(i-1)} + \mathbf{c}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})) - \mathbf{y}^{(i-1)}\|_2 \leq 2D \enspace .\end{aligned}$$

Note that the last inequality holds because the fact that $\mathbf{c}^{(i)} \leq \mathbf{z}^{(i-1)} \in \mathcal{K}_{\mathrm{D}}$ implies that $\mathbf{y}^{(i-1)} + \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})$, $\mathbf{y}^{(i-1)} + \mathbf{c}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)})$ and $\mathbf{y}^{(i-1)}$ are all vectors of $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$. The second expression (of the two mentioned above) is

$$\begin{aligned}\langle (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}), \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \rangle &= \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}) \rangle \\ &\geq \langle \mathbf{o}_{\mathrm{D}}^{(2)} \odot (\bar{1} - \mathbf{z}^{(i-1)}), \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}) \rangle \\ &\geq F(\mathbf{o}_{\mathrm{D}}^{(2)} \oplus \mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \\ &\geq (1-\varepsilon)^{i-1}(1-m) \cdot F(\mathbf{o}_{\mathrm{D}}^{(2)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \ ,\end{aligned}$$

where the first inequality holds since the second side of the above inequality is identical to the objective function of the second linear program of Algorithm 4, and one feasible solution for this linear program is $\mathbf{b}^{(i)} = \mathbf{o}_{\mathrm{D}}^{(2)} \odot (\bar{1} - \mathbf{z}^{(i-1)})$ and $\mathbf{c}^{(i)} = \bar{0}$. The second inequality follows from Property (1) of Lemma B.1, and the last inequality follows from Corollary B.3 and Lemma C.2. The lemma now follows by plugging the lower bounds we have proved into the rightmost side of Equality (12), and then solving the integral obtained. ☐

It remains to show that Lemma C.14 applies to Algorithm 4. The next two lemmata are steps toward this goal.

Lemma C.22. *For every $i \in [\varepsilon^{-1} t_s]$,*

$$\begin{aligned}&e^{2\varepsilon i} \cdot F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \\ &\quad \geq 2\varepsilon e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + \varepsilon e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \\ &\qquad\qquad + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - \varepsilon \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})) \rangle - 56\varepsilon^2 \beta D^2 \ .\end{aligned}$$

Proof. By the chain rule,

$$\begin{aligned}&F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \qquad (13)\\ &\quad = F(((1-\varepsilon)\mathbf{y}^{(i-1)} + \varepsilon \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \varepsilon(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \\ &\quad = \int_0^{\varepsilon} \Big\langle \frac{d[((1-\tau)\mathbf{y}^{(i-1)} + \tau \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))]}{d\tau}, \\ &\qquad\qquad \nabla F(((1-\tau)\mathbf{y}^{(i-1)} + \tau \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))) \Big\rangle d\tau \\ &\quad = \int_0^{\varepsilon} \langle (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - 2\tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})), \\ &\qquad\qquad \nabla F(((1-\tau)\mathbf{y}^{(i-1)} + \tau \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))) \rangle d\tau \ .\end{aligned}$$

The following inequality is used later to lower bound the integrand on the rightmost side of the last equality.

$$\begin{aligned}&\langle (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - 2\tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})), \qquad (14)\\ &\qquad \nabla F(((1-\tau)\mathbf{y}^{(i-1)} + \tau \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))) - \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \rangle \\ \geq\ & -\|(\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - 2\tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))\|_2 \cdot \\ &\qquad \|\nabla F(((1-\tau)\mathbf{y}^{(i-1)} + \tau \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))) - \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\|_2 \\ \geq\ & -5D \cdot \beta \|((1-\tau)\mathbf{y}^{(i-1)} + \tau \mathbf{a}^{(i)}) \oplus (\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)})) - \mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}\|_2 \\ =\ & -5\beta D \cdot \|\tau(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}) + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}) \\ &\qquad - \tau^2(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})\|_2 \geq -15\tau\beta D^2 \ ,\end{aligned}$$

where the first inequality follows from the Cauchy-Schwarz inequality, and the second inequality uses the β-smoothness of F and the observation that since $\mathbf{y}^{(i)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}$, $\mathbf{y}^{(i)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{c}^{(i)}$, $\mathbf{a}^{(i)}$ and $\mathbf{y}^{(i-1)}$ are all vectors in $(\mathcal{K}_{\mathrm{N}} + \mathcal{K}_{\mathrm{D}}) \cap [0,1]^n$ (because $\mathbf{c}^{(i)} \leq \mathbf{z}^{(i-1)}$ and $\mathcal{K}_{\mathrm{D}}$ is down-closed), it holds that

$$\begin{aligned}&\|(\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - 2\tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))\|_2 \\ &\quad \leq \|(\mathbf{y}^{(i-1)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}) - \mathbf{y}^{(i-1)}\|_2 + \|(\mathbf{y}^{(i-1)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{c}^{(i)}) - \mathbf{y}^{(i-1)}\|_2 \\ &\qquad - \|\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}\|_2 \odot \|\bar{1} - \mathbf{z}^{(i-1)} - 2\tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})\|_\infty \leq 5D \ .\end{aligned}$$

Similarly, the last inequality of Inequality (14) holds since

$$\begin{aligned}&\|\tau(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}) + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)}) - \tau^2(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})\|_2\\ &\quad= \|\tau(\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}) \odot (1 - \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)})) + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}) \odot (\bar{1} - \mathbf{y}^{(i-1)})\|_2\\ &\quad\leq \|\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}\|_2 + \|(\mathbf{y}^{(i-1)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{b}^{(i)}) - \mathbf{y}^{(i-1)}\|_2\\ &\qquad\qquad + \|(\mathbf{y}^{(i-1)} + (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{c}^{(i)}) - \mathbf{y}^{(i-1)}\|_2 \leq 3D \enspace,\end{aligned}$$

where the penultimate inequality uses the fact that $\|(\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} + \mathbf{c}^{(i)})\|_\infty \leq \|(\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} + \mathbf{z}^{(i-1)})\|_\infty \leq 1$.

Combining Inequalities (13) and (14) now yields

$$\begin{aligned}&e^{2\varepsilon i} \cdot F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})\\ &\quad= [e^{2\varepsilon i} - e^{2\varepsilon(i-1)}] \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + e^{2\varepsilon i} \cdot [F(\mathbf{y}^{(i)} \oplus \mathbf{z}^{(i)}) - F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})]\\ &\quad\geq \int_0^\varepsilon \{2e^{2(\varepsilon(i-1)+\tau)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})\\ &\qquad\qquad + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - 2\tau \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))\rangle - 111\tau\beta D^2\} d\tau\\ &\quad\geq 2\varepsilon e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) + \varepsilon e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})\\ &\qquad\qquad + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - \varepsilon \cdot (\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))\rangle - 56\varepsilon^2\beta D^2 \enspace,\end{aligned}$$

where the second inequality uses the non-negativity of F. □

Lemma C.23. *For every $i \in [\varepsilon^{-1}t_s]$,*

$$\begin{aligned}&e^{\varepsilon i}(t_s - \varepsilon i) \cdot F(\mathbf{z}^{(i)}) - e^{\varepsilon(i-1)}(t_s - \varepsilon(i-1)) \cdot F(\mathbf{z}^{(i-1)})\\ &\quad\geq -\varepsilon e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) + \varepsilon e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle - 6\varepsilon^2\beta D^2/(1-m)^2 \enspace.\end{aligned}$$

Proof. By the chain rule,

$$\begin{aligned}&F(\mathbf{z}^{(i)}) - F(\mathbf{z}^{(i-1)}) = \int_0^\varepsilon \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)}))\rangle d\tau\\ &= \varepsilon \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle + \int_0^\varepsilon \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)})) - \nabla F(\mathbf{z}^{(i-1)})\rangle d\tau\\ &\geq \varepsilon \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle - \int_0^\varepsilon \|\mathbf{b}^{(i)} - \mathbf{c}^{(i)}\|_2 \cdot \|\nabla F(\mathbf{z}^{(i-1)} + \tau(\mathbf{b}^{(i)} - \mathbf{c}^{(i)})) - \nabla F(\mathbf{z}^{(i-1)})\|_2 d\tau\\ &\geq \varepsilon \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle - \beta\|\mathbf{b}^{(i)} - \mathbf{c}^{(i)}\|_2^2 \cdot \int_0^\varepsilon \tau d\tau \geq \varepsilon \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle - \frac{2\varepsilon^2\beta D^2}{(1-m)^2} \enspace,\end{aligned}$$

where the first inequality follows from the Cauchy–Schwarz inequality, the second inequality holds by the β-smoothness of F, and the last inequality uses the fact that both $\mathbf{b}^{(i)}$ and $\mathbf{c}^{(i)}$ are vectors in $\mathcal{K}_{\text{D}}$ (in the case of $\mathbf{c}^{(i)}$ this holds by the down-monotonicity of $\mathcal{K}_{\text{D}}$ since $\mathbf{c}^{(i)} \leq \mathbf{z}^{(i-1)}$), and thus, the ℓ_2 norm of both these vectors is at most $D/(1-m)$ according to the proof of Lemma C.5.

Using the last inequality, we now get

$$\begin{aligned}&e^{\varepsilon i}(t_s - \varepsilon i) \cdot F(\mathbf{z}^{(i)}) - e^{\varepsilon(i-1)}(t_s - \varepsilon(i-1)) \cdot F(\mathbf{z}^{(i-1)})\\ &\quad= [e^{\varepsilon i}(t_s - \varepsilon i) - e^{\varepsilon(i-1)}(t_s - \varepsilon(i-1))] \cdot F(\mathbf{z}^{(i-1)}) + e^{\varepsilon i}(t_s - \varepsilon i) \cdot [F(\mathbf{z}^{(i)}) - F(\mathbf{z}^{(i-1)})]\\ &\quad\geq \int_0^\varepsilon e^{\varepsilon(i-1)+\tau}(t_s - 1 - \varepsilon(i-1) - \tau) \cdot F(\mathbf{z}^{(i-1)}) d\tau + \varepsilon e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle\\ &\qquad\qquad - 6\varepsilon^2\beta D^2/(1-m)^2\\ &\quad\geq -\varepsilon e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) + \varepsilon e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)})\rangle - 6\varepsilon^2\beta D^2/(1-m)^2 \enspace,\end{aligned}$$

where the second inequality uses F's non-negativity. □

We are now ready to prove Lemma C.14 in the context of Algorithm 4. Like in the case of Lemma C.18, the version of Lemma C.14 that we prove here has slightly worse error terms compared to the version from Section C.2, but the difference again affects the proof of Theorem 4.1 only by changing the constants hidden by the big O notation as long as $\varepsilon \leq 1/60$.

Lemma C.14. *For every integer $1 \leq i \leq \varepsilon^{-1}t_s$, the expression $\varepsilon^{-1}e^{2t_s}[\phi(i) - \phi(i-1)]$ can be lower bounded both by $2e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) - \frac{62\varepsilon\beta D^2}{1-m} - 30\varepsilon \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)})$ and by the sum of this expression and*

$$(1-\varepsilon) \cdot e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{o}_{\text{D}}^{(1)} + \mathbf{o}_{\text{N}} - \mathbf{y}^{(i-1)} \rangle$$
$$+ (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{o}_{\text{D}}^{(1)}, \nabla F(\mathbf{z}^{(i-1)}) \rangle \enspace .$$

Proof. Adding $(1-m)(1-\varepsilon)$ times the guarantee of Lemma C.23 to the guarantee of Lemma C.22, we get

$$\begin{aligned}
&\varepsilon^{-1}e^{2t_s}[\phi(i) - \phi(i-1)] \qquad (15)\\
&\geq 2e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) - \tfrac{62\varepsilon\beta D^2}{1-m}\\
&\quad + (1-\varepsilon)e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i-1)}) + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}) \rangle\\
&\quad + (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)}) \rangle\\
&\quad + \varepsilon e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})) \rangle \enspace .
\end{aligned}$$

To bound the last term on the rightmost side of the last inequality, we need to make a few observations. First, $\bar{1} - \mathbf{z}^{(i-1)} - \mathbf{b}^{(i)} + \mathbf{c}^{(i)} \in [0,1]^n$ because $\mathbf{c}^{(i)} \leq \mathbf{z}^{(i-1)}$ and $\mathbf{b}^{(i)} \leq \bar{1} - \mathbf{z}^{(i-1)}$; second, $\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} \geq \mathbf{y}^{(i-1)} \geq \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{a}^{(i)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - \mathbf{b}^{(i)} + \mathbf{c}^{(i)})$; and finally, by Lemma C.2, $\|\mathbf{z}^{i-1} + \mathbf{a}^{(i)} \odot \mathbf{c}^{(i)}/2\|_\infty \leq \|\mathbf{z}^{i-1} + \mathbf{c}^{(i)}/2\|_\infty \leq (3/2) \cdot \|\mathbf{z}^{(i-1)}\|_\infty \leq (3/2) \cdot (1 - 1/e) < 1$. Given all these observations, we can use Properties 1 and 2 of Lemma B.1 to get

$$\begin{aligned}
&\langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - (\mathbf{b}^{(i)} - \mathbf{c}^{(i)})) \rangle\\
&= \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{a}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - \mathbf{b}^{(i)}) \rangle\\
&\quad + 2 \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{a}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1)}) \odot \mathbf{c}^{(i)}/2 \rangle\\
&\quad - \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{a}^{(i)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - \mathbf{b}^{(i)} + \mathbf{c}^{(i)}) \rangle\\
&\geq F(\mathbf{y}^{(i-1)} \oplus (\mathbf{a}^{(i)} \odot (\bar{1} - \mathbf{b}^{(i)}) + (\bar{1} - \mathbf{a}^{(i)}) \odot \mathbf{z}^{(i-1)})) + 2F(\mathbf{y}^{(i-1)} \oplus (\mathbf{z}^{(i-1)} + \mathbf{a}^{(i)} \odot \mathbf{c}^{(i-1)}/2))\\
&\quad + F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)} - \mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{a}^{(i)}) \odot (\bar{1} - \mathbf{z}^{(i-1)} - \mathbf{b}^{(i)} + \mathbf{c}^{(i)}))\\
&\quad - 4 \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \geq -4 \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace ,
\end{aligned}$$

where the last inequality holds by the non-negativity of F. Plugging this inequality into Inequality (15) yields

$$\begin{aligned}
&\varepsilon^{-1}e^{2t_s}[\phi(i) - \phi(i-1)]\\
&\geq 2e^{2\varepsilon(i-1)} \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) - (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F(\mathbf{z}^{(i-1)}) - \tfrac{62\varepsilon\beta D^2}{1-m}\\
&\quad + (1-\varepsilon)e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1)}) \odot (\mathbf{b}^{(i)} - \mathbf{c}^{(i-1)}) + (\mathbf{a}^{(i)} - \mathbf{y}^{(i-1)}) \odot (\bar{1} - \mathbf{z}^{(i-1)})\\
&\quad + (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle \mathbf{b}^{(i)} - \mathbf{c}^{(i)}, \nabla F(\mathbf{z}^{(i-1)}) \rangle - 30\varepsilon \cdot F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}) \enspace .
\end{aligned}$$

The last two terms on the right hand side of the last inequality are related to the objective function of the first linear program of Algorithm 4. Specifically, to get from them to the objective function of this linear program, it is necessary to multiply by $(1-\varepsilon)^{-1}$, and then remove the additive term $e^{2\varepsilon i} \cdot \langle \nabla F(\mathbf{y}^{(i-1)} \oplus \mathbf{z}^{(i-1)}), -\mathbf{y}^{(i-1)} \odot (\bar{1} - \mathbf{z}^{(i-1)}) \rangle$, which does not depend on the variables $\mathbf{a}^{(i)}$, $\mathbf{b}^{(i)}$ and $\mathbf{c}^{(i)}$. Therefore, we can lower bound the right hand side of the last inequality by plugging in feasible solutions for the first linear program of Algorithm 4. Specifically, we plug in the solutions $(\mathbf{a}^{(i)}, \mathbf{b}^{(i)}, \mathbf{c}^{(i)}) = (\mathbf{o}_{\text{N}}, (\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{o}_{\text{D}}^{(1)}, \bar{0})$ and $(\mathbf{a}^{(i)}, \mathbf{b}^{(i)}, \mathbf{c}^{(i)}) = (\mathbf{y}^{(i-1)}, \bar{0}, \bar{0})$, which implies the two lower bounds stated in the lemma. Notice that both these solutions are guaranteed to be feasible since $\mathcal{K}_{\text{D}}$ is down-closed and $(\bar{1} - \mathbf{z}^{(i-1)}) \odot \mathbf{o}_{\text{D}}^{(1)} \leq (\bar{1} - \mathbf{z}^{(i-1)}) \odot (\bar{1} - \mathbf{o}_{\text{N}})$. □

D Analysis of Our Online Algorithm

In this section, we prove that our online algorithm (i.e., Algorithm 2) obeys the guarantee of Proposition 5.2. Similarly to Appendix C, we use $\mathbf{o}$ below as a shorthand for the sum $\mathbf{o}_{\text{N}} + \mathbf{o}_{\text{D}}$.

One can observe that, for every fixed time step $\ell \in [L]$, Algorithm 2 sets the vectors $\{\mathbf{y}^{(i,\ell)}, \mathbf{z}^{(i,\ell)} \mid i \in \mathbb{Z}, 0 \leq i \leq \varepsilon^{-1}\}$ in the same way in which Algorithm 3 sets the vectors $\{\mathbf{y}^{(i)}, \mathbf{z}^{(i)} \mid i \in \mathbb{Z}, 0 \leq i \leq \varepsilon^{-1}\}$, with the sole difference that the vectors $\{\mathbf{a}^{(i,\ell)}, \mathbf{b}^{(i,\ell)} \mid i \in [\varepsilon^{-1}]\}$ used for this purpose may not optimize the linear programs of Algorithm 3 (but they are feasible solutions for these linear programs). Since the property that these vectors optimize the linear programs of Algorithm 3 is not used in the proof that Lemmata C.1 and C.2 apply to this algorithm, we immediately get that both these lemmata apply also to Algorithm 2 for any fixed ℓ. In other words, we get the following lemma.

Lemma D.1. *For every two integers $0 \leq i \leq \varepsilon^{-1}$ and $\ell \in [L]$,*

- $\mathbf{y}^{(i,\ell)} \in \mathcal{K}_\text{N}$ *and* $\mathbf{z}^{(i,\ell)} \in \varepsilon i \cdot \mathcal{K}_\text{D}$*, and thus,* $\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)} \in (\mathcal{K}_\text{N} + \mathcal{K}_\text{D}) \cap [0,1]^n$.
- $\|\mathbf{z}^{(i,\ell)}\|_\infty \leq 1 - (1-\varepsilon)^i$ *and* $\|\mathbf{y}^{(i,\ell)}\|_\infty \leq \|\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)}\|_\infty \leq 1 - (1-\varepsilon)^i(1-m)$.

The last lemma implies, in particular, that Algorithm 2 outputs feasible vectors. The rest of this section is devoted to showing that the total value of these vectors is at least as large as is guaranteed by Proposition 5.2. For that purpose, we need to define a new potential function $\phi(i) \triangleq \sum_{\ell=1}^{L} \phi(i,\ell)$, where

$$\phi(i,\ell) \triangleq e^{2(\varepsilon i - t_s)} \cdot F_\ell(\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)}) + (1-m)(1-\varepsilon) \cdot e^{\varepsilon i - 2t_s}(t_s - \varepsilon i) \cdot F_\ell(\mathbf{z}^{(i,\ell)}) \enspace .$$

Notice that the definition $\phi(i,\ell)$ is identical to the definition of the potential function from Section C.2 up to the addition of the index ℓ to the various vectors and the function F.

The next lemma corresponds to Lemma C.14 from the analysis of Algorithm 3. Notice that this lemma has only one guarantee, in contrast to Lemma C.14 that has two guarantees (because the proof of the other guarantee does not generalize to the online setting). Due to the lack of this addition guarantee, there is no result in this section corresponding to Corollary C.15 from the analysis of Algorithm 3, and the proof of Lemma D.4 is more involved than the proof of the corresponding Lemma C.17.

Lemma D.2. *For every integer $1 \leq i \leq \varepsilon^{-1} t_s$,*

$$\begin{aligned}
\varepsilon^{-1} e^{2t_s}[\phi(i) - \phi(i-1)] \geq{}& 2e^{2\varepsilon(i-1)} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \\
&- (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{z}^{(i-1,\ell)}) - 25\varepsilon\beta L D^2 - 15\varepsilon \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \\
&+ (1-\varepsilon) \cdot e^{2\varepsilon i} \cdot \sum_{\ell=1}^{L} \langle \nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1)}), (\bar{1} - \mathbf{y}^{(i-1,\ell)}) \odot \mathbf{o}_\text{D} + \mathbf{o}_\text{N} - \mathbf{y}^{(i-1,\ell)} \rangle \\
&+ (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \sum_{\ell=1}^{L} \langle (\bar{1} - \mathbf{z}^{(i-1,\ell)}) \odot \mathbf{o}_\text{D}, \nabla F_\ell(\mathbf{z}^{(i-1,\ell)}) \rangle - 24DG\sqrt{L} \enspace .
\end{aligned}$$

Proof. The definition of the convex-body assigned to $\mathcal{E}_i$ guarantees that, for every $\ell \in [L]$, there exists a vector $\mathbf{x}^{(\ell)} \in \mathcal{K}_\text{N}$ such that $\mathbf{x}^{(\ell)} + \mathbf{b}^{(i,\ell)} \in (\mathcal{K}_\text{N} + \mathcal{K}_\text{D}) \cap [0,1]^n$. Thus, $\|\mathbf{b}^{(i,\ell)}\|_2 = \|(\mathbf{x}^{(\ell)} + \mathbf{b}^{(i,\ell)}) - \mathbf{x}^{(\ell)}\|_2 \leq D$ because the down-closeness of $\mathcal{K}_\text{D}$ implies that $\mathbf{x}^{(\ell)}$ also belongs to $(\mathcal{K}_\text{N} + \mathcal{K}_\text{D}) \cap [0,1]^n$. Plugging this observation into the first part of the proof of Lemma C.14, we get, for every $\ell \in [L]$,

$$\begin{aligned}
\varepsilon^{-1} e^{2t_s}[\phi(i,\ell) - \phi(i-1,\ell)] \geq{}& 2e^{2\varepsilon(i-1)} \cdot F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \\
&- (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot F_\ell(\mathbf{z}^{(i-1,\ell)}) - 25\varepsilon\beta D^2 - 15\varepsilon \cdot F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \\
&+ (1-\varepsilon) \cdot e^{2\varepsilon i} \cdot \langle \nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)}), (\bar{1} - \mathbf{y}^{(i-1,\ell)}) \odot \mathbf{b}^{(i,\ell)} + \mathbf{a}^{(i,\ell)} - \mathbf{y}^{(i-1,\ell)} \rangle \\
&+ (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \langle (\bar{1} - \mathbf{z}^{(i-1,\ell)}) \odot \mathbf{b}^{(i,\ell)}, \nabla F_\ell(\mathbf{z}^{(i-1,\ell)}) \rangle \enspace .
\end{aligned}$$

Adding up this inequality for all $\ell \in [L]$ yields

$$\begin{aligned}
\varepsilon^{-1} e^{2t_s}[\phi(i) - \phi(i-1)] \geq{}& 2e^{2\varepsilon(i-1)} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \qquad (16) \\
&- (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(1 - t_s + \varepsilon i) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{z}^{(i-1,\ell)}) - 25\varepsilon\beta L D^2 - 15\varepsilon \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \\
&+ (1-\varepsilon)e^{2\varepsilon i} \cdot \sum_{\ell=1}^{L} \langle \nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)}), (\bar{1} - \mathbf{y}^{(i-1,\ell)}) \odot \mathbf{b}^{(i,\ell)} + \mathbf{a}^{(i,\ell)} - \mathbf{y}^{(i-1,\ell)} \rangle \\
&+ (1-m)(1-\varepsilon) \cdot e^{\varepsilon i}(t_s - \varepsilon i) \cdot \sum_{\ell=1}^{L} \langle (\bar{1} - \mathbf{z}^{(i-1,\ell)}) \odot \mathbf{b}^{(i,\ell)}, \nabla F_\ell(\mathbf{z}^{(i-1,\ell)}) \rangle \enspace .
\end{aligned}$$

One can observe that the last two terms of on the right hand side of the last inequality are equal to the expression $(1-\varepsilon) \cdot \sum_{\ell=1}^{L} \langle \mathbf{g}^{(i,\ell)}, (\mathbf{a}^{(i,\ell)}, \mathbf{b}^{(i,\ell)}) \rangle$ up to an additive term of $(1-\varepsilon)e^{2\varepsilon i} \cdot \sum_{\ell=1}^{L} \langle \nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)}), -\mathbf{y}^{(i-1,\ell)} \rangle$.

Therefore, since $(\mathbf{o}_\mathrm{N}, \mathbf{o}_\mathrm{D})$ is one possible solution that $\mathcal{E}_i$ can return, the guarantee of $\mathcal{E}_i$ implies

$$\begin{aligned}(1-\varepsilon)e^{2\varepsilon i}\cdot&\sum_{\ell=1}^{L}\langle\nabla F_\ell(\mathbf{y}^{(i-1,\ell)}\oplus\mathbf{z}^{(i-1,\ell)})\odot(\bar{1}-\mathbf{z}^{(i-1,\ell)}),(\bar{1}-\mathbf{y}^{(i-1,\ell)})\odot\mathbf{b}^{(i,\ell)}+\mathbf{a}^{(i,\ell)}-\mathbf{y}^{(i-1,\ell)}\rangle \qquad (17)\\&+(1-m)(1-\varepsilon)\cdot e^{\varepsilon i}(t_s-\varepsilon i)\cdot\sum_{\ell=1}^{L}\langle(\bar{1}-\mathbf{z}^{(i-1,\ell)})\odot\mathbf{b}^{(i,\ell)},\nabla F_\ell(\mathbf{z}^{(i-1,\ell)})\rangle\\&\geq(1-\varepsilon)e^{2\varepsilon i}\cdot\sum_{\ell=1}^{L}\langle\nabla F_\ell(\mathbf{y}^{(i-1,\ell)}\oplus\mathbf{z}^{(i-1,\ell)})\odot(\bar{1}-\mathbf{z}^{(i-1,\ell)}),(\bar{1}-\mathbf{y}^{(i-1,\ell)})\odot\mathbf{o}_\mathrm{D}+\mathbf{o}_\mathrm{N}-\mathbf{y}^{(i-1,\ell)}\rangle\\&+(1-m)(1-\varepsilon)\cdot e^{\varepsilon i}(t_s-\varepsilon i)\cdot\sum_{\ell=1}^{L}\langle(\bar{1}-\mathbf{z}^{(i-1,\ell)})\odot\mathbf{o}_\mathrm{N},\nabla F_\ell(\mathbf{z}^{(i-1,\ell)})\rangle-D'\sqrt{2L}\cdot\max\nolimits_{1\leq i\leq L}\|\mathbf{g}^{(i,\ell)}\|_2\ ,\end{aligned}$$

where D' is the diameter of the convex body assigned to $\mathcal{E}_i$.

We need to upper bound the terms D' and $\max_{1\leq i\leq L}\|\mathbf{g}^{(i,\ell)}\|_2$ appearing in the last inequality. First,

$$\begin{aligned}D'={}&\max_{\substack{\mathbf{x}^{(1)},\mathbf{x}^{(2)}\in\mathcal{K}_\mathrm{N},\mathbf{w}^{(1)},\mathbf{w}^{(2)}\in\mathcal{K}_\mathrm{D}\\\mathbf{x}^{(1)}+\mathbf{w}^{(1)},\mathbf{x}^{(2)}+\mathbf{w}^{(2)}\in[0,1]^n}}\|(\mathbf{x}^{(1)},\mathbf{w}^{(1)})-(\mathbf{x}^{(2)},\mathbf{w}^{(2)})\|_2\\\leq{}&\max_{\substack{\mathbf{x}^{(1)},\mathbf{x}^{(2)}\in\mathcal{K}_\mathrm{N},\mathbf{w}^{(1)},\mathbf{w}^{(2)}\in\mathcal{K}_\mathrm{D}\\\mathbf{x}^{(1)}+\mathbf{w}^{(1)},\mathbf{x}^{(2)}+\mathbf{w}^{(2)}\in[0,1]^n}}\|(\mathbf{x}^{(1)}+\mathbf{w}^{(1)})-(\mathbf{x}^{(2)}+\mathbf{w}^{(2)})\|_2\leq D\ .\end{aligned}$$

Second, for every $\ell\in[L]$, the ℓ_2-norm of the first n coordinates of $\mathbf{g}^{(i,\ell)}$ is upper bounded by $e^{2\varepsilon i}\cdot\|\nabla F_\ell(\mathbf{y}^{(i-1,\ell)}\oplus\mathbf{z}^{(i-1,\ell)})\|\leq e^2G$, and the ℓ_2-norm of the other n coordinates of $\mathbf{g}^{(i,\ell)}$ is upper bounded by $\|e^{2\varepsilon i}\cdot\nabla F_\ell(\mathbf{y}^{(i-1,\ell)}\oplus\mathbf{z}^{(i-1,\ell)})+e^{\varepsilon i}\cdot\nabla F_\ell(\mathbf{z}^{(i-1,\ell)})\|_2\leq 2e^2G$. Thus,

$$\|\mathbf{g}^{(i,\ell)}\|_2\leq\sqrt{(e^2G)^2+(2e^2G)^2}=\sqrt{5}\cdot e^2G\leq\frac{24}{\sqrt{2}}\cdot G\ .$$

The lemma now follows by plugging the above upper bounds into Inequality (17), and then combining the obtained inequality with Inequality (16). ☐

To get the following corollary, we repeat the proof of Lemma C.16 from the analysis of Algorithm 3 with Lemma D.2 taking the role of Lemma C.14 and $\mathbf{o}_\mathrm{D}$ taking the role of $\mathbf{o}_\mathrm{D}^{(1)}$.

Corollary D.3. *For every integer* $1\leq i\leq\varepsilon^{-1}t_s$,

$$\begin{aligned}\phi(i)-\phi(i-1)\geq\varepsilon(1-m)(1-\varepsilon)\cdot\left[e^{\varepsilon i-2t_s}\cdot\sum_{\ell=1}^{L}F_\ell(\mathbf{o})+e^{-2t_s}(t_s-\varepsilon i)\cdot\sum_{\ell=1}^{L}F_\ell(\mathbf{o}_\mathrm{D})\right]\\-25\varepsilon^2\beta LD^2-62\varepsilon^2\cdot\phi(i-1)-24\varepsilon DG\sqrt{L}\ .\end{aligned}$$

We now use the last corollary to get a lower bound on $\sum_{\ell=1}^{L}F_\ell(\mathbf{y}^{(\varepsilon^{-1}t_s,\ell)}\oplus\mathbf{z}^{(\varepsilon^{-1}t_s,\ell)})=\phi(\varepsilon^{-1}t_s)$. As mentioned above, Lemma D.4 corresponds to Lemma C.17 from the analysis of Algorithm 3, but its analysis is somewhat more involved.

Lemma D.4. *It holds that*

$$\begin{aligned}\sum_{\ell=1}^{L}F_\ell(\mathbf{y}^{(\varepsilon^{-1}t_s,\ell)}\oplus\mathbf{z}^{(\varepsilon^{-1}t_s,\ell)})=\phi(\varepsilon^{-1}t_s)\geq-63t_s\varepsilon\beta LD^2-60t_sDG\sqrt{L}\\+(1-m)\cdot\left[(e^{-t_s}-e^{-2t_s}-O(\varepsilon))\cdot\sum_{\ell=1}^{L}F_\ell(\mathbf{o})+\left(\frac{e^{-2t_s}\cdot t_s^2}{2}-O(\varepsilon)\right)\cdot\sum_{\ell=1}^{L}F_\ell(\mathbf{o}_\mathrm{D})\right]\ .\end{aligned}$$

Proof. We prove by induction on i that

$$\begin{aligned}(1-62\varepsilon^2)^{-i}\cdot\phi(i)\geq\varepsilon(1-m)(1-\varepsilon)\cdot\sum_{i'=1}^{i}\left[e^{\varepsilon i'-2t_s}\cdot\sum_{\ell=1}^{L}F_\ell(\mathbf{o})+e^{-2t_s}(t_s-\varepsilon i')\cdot\sum_{\ell=1}^{L}F_\ell(\mathbf{o}_\mathrm{D})\right] \qquad (18)\\-63i\varepsilon^2\beta LD^2-60i\varepsilon DG\sqrt{L}\ .\end{aligned}$$

Inequality (18) holds for $i = 0$ since the non-negativity of F guarantees that $\phi(0) \geq 0$. Assume now that Inequality (18) holds for $i - 1$, and let us prove it for i. By Corollary D.3 and the induction hypothesis,

$$\begin{aligned}
(1-62\varepsilon^2)^{-i} \cdot \phi(i) &= (1-62\varepsilon^2)^{-(i-1)} \cdot \phi(i-1) + (1-62\varepsilon^2)^{-i} \cdot [\phi(i) - (1-62\varepsilon^2)\phi(i-1)] \\
&\geq \varepsilon(1-m)(1-\varepsilon) \cdot \sum_{i'=1}^{i-1} \left[e^{\varepsilon i' - 2t_s} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}) + e^{-2t_s}(t_s - \varepsilon i') \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}})\right] \\
&\quad - 63(i-1)\varepsilon^2\beta L D^2 - 60(i-1)\varepsilon D G\sqrt{L} \\
&\quad + (1-62\varepsilon^2)^{-i} \cdot \left\{\varepsilon(1-m)(1-\varepsilon) \cdot \left[e^{\varepsilon i - 2t_s} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}) + e^{-2t_s}(t_s - \varepsilon i) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}})\right]\right. \\
&\quad \left. - 25\varepsilon^2\beta L D^2 - 24\varepsilon D G\sqrt{L}\right\} \\
&\geq \varepsilon(1-m)(1-\varepsilon) \cdot \sum_{i'=1}^{i} \left[e^{\varepsilon i' - 2t_s} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}) + e^{-2t_s}(t_s - \varepsilon i') \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}})\right] \\
&\quad - 63i\varepsilon^2\beta L D^2 - 60i\varepsilon D G\sqrt{L} \enspace ,
\end{aligned}$$

where the second inequality uses the non-negativity of F and the fact that our assumption that $\varepsilon \leq 1/70$ implies that $(1 - 62\varepsilon^2)^{-i} \leq (1-62\varepsilon^2)^{-1/\varepsilon} \leq e^{62\varepsilon} \leq 2.5$. This completes the proof of Inequality (18). Plugging $i = \varepsilon^{-1}t_s$ into this inequality, we get

$$\begin{aligned}
\phi(\varepsilon^{-1}t_s) &\geq (1-62\varepsilon^2)^{\varepsilon^{-1}t_s} \cdot \left\{\varepsilon(1-m)(1-\varepsilon) \cdot \sum_{i'=1}^{\varepsilon^{-1}t_s} \left[e^{\varepsilon i' - 2t_s} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}) + e^{-2t_s}(t_s - \varepsilon i') \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}})\right]\right. \\
&\quad \left. - 63t_s\varepsilon\beta L D^2 - 60t_s D G\sqrt{L}\right\} \\
&\geq (1-m) \cdot \left[(e^{-t_s} - e^{-2t_s} - O(\varepsilon)) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}) + \left(\frac{e^{-2t_s} \cdot t_s^2}{2} - O(\varepsilon)\right) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}})\right] \\
&\quad - 63t_s\varepsilon\beta L D^2 - 60t_s D G\sqrt{L} \enspace ,
\end{aligned}$$

where the second inequality uses two lower bounds on sums that are justified in the proof of Lemma C.17. □

Up to this point we have only studied vectors in $\{\mathbf{y}^{(i,\ell)}, \mathbf{z}^{(i,\ell)} \mid i \in \mathbb{Z}, 0 \leq i \leq \varepsilon^{-1}t_s, \ell \in [L]\}$. We now need to consider vectors corresponding to larger values of i. We begin with the following lemma, which corresponds to Lemma C.18 from the analysis of Algorithm 3.

Lemma D.5. *For every integer $\varepsilon^{-1}t_s < i \leq \varepsilon^{-1}$,*

$$\begin{aligned}
&\sum_{\ell=1}^{\ell} [F_\ell(\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)}) - F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)})] \\
&\qquad \geq \max\left\{\varepsilon \cdot \left[(1-m)(1-\varepsilon)^{i-1} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) - \sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)})\right], 0\right\} \\
&\qquad\quad - \varepsilon^2\beta L D^2/2 - \varepsilon D G\sqrt{2L} \enspace .
\end{aligned}$$

Proof. Repeating the first part of the proof of Lemma C.18 implies that, for every $\ell \in [L]$,

$$\begin{aligned}
&F_\ell(\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)}) - F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \\
&\qquad \geq \varepsilon \cdot \langle \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1,\ell)}), \nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)})\rangle - \varepsilon^2\beta D^2/2 \enspace .
\end{aligned}$$

Summing up this inequality for all $\ell \in [L]$ yields

$$\begin{aligned}
&\sum_{\ell=1}^{\ell}[F_\ell(\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)}) - F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)})] \\
&\qquad\qquad \geq \varepsilon \cdot \sum_{\ell=1}^{L} \langle \mathbf{b}^{(i)} \odot (\bar{1} - \mathbf{y}^{(i-1,\ell)}), \nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)})\rangle - \varepsilon^2\beta L D^2/2 \\
&\qquad\qquad \geq \max\left\{\varepsilon \cdot \sum_{\ell=1}^{L} \langle \mathbf{o}_{\mathrm{D}} \odot (\bar{1} - \mathbf{y}^{(i-1,\ell)}), \nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)})\rangle, 0\right\} \\
&\qquad\qquad\quad - \varepsilon^2\beta L D^2/2 - \varepsilon D G\sqrt{2L} \enspace ,
\end{aligned}$$

where the second inequality holds by the properties of $\mathcal{E}_i$ since (i) the convex body assigned to $\mathcal{E}_i$ is of diameter at most D (see the proof of Lemma D.2), (ii) both $(\mathbf{o}_{\mathrm{N}}, \mathbf{o}_{\mathrm{D}})$ and $(\mathbf{o}_{\mathrm{N}}, \bar{0})$ are vectors in this convex body, and (iii) for every $\ell \in [L]$, $\|g^{(i,\ell)}\|_2 \leq \|\nabla F(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)})\|_2 \leq G$.

To complete the proof of the lemma, it remains to observe that, for every $\ell \in [L]$,

$$\begin{aligned}
&\langle \mathbf{o}_{\mathrm{D}} \odot (\bar{1} - \mathbf{y}^{(i-1,\ell)}), \nabla F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \odot (\bar{1} - \mathbf{z}^{(i-1,\ell)})\rangle \\
&\qquad \geq F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)} \oplus \mathbf{o}_{\mathrm{D}}) - F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \\
&\qquad \geq (1-m)(1-\varepsilon)^{i-1} \cdot F_\ell(\mathbf{o}_{\mathrm{D}}) - F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)}) \enspace ,
\end{aligned}$$

where the first inequality holds by Property 1 of Lemma B.1, and the second inequality follows from Corollary B.3 and Lemma D.1. □

Corollary D.6. *For every integer $\varepsilon^{-1}t_s \leq i \leq \varepsilon^{-1}$, $\sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(\varepsilon^{-1},\ell)} \oplus \mathbf{z}^{(\varepsilon^{-1},\ell)}) \geq (1-m)(\varepsilon i - t_s)(1-\varepsilon)^{i-1} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) + (1-\varepsilon)^{i-\varepsilon^{-1}t_s} \cdot \sum_{i=1}^{L} F_\ell(\mathbf{y}^{(\varepsilon^{-1}t_s,\ell)} \oplus \mathbf{z}^{(\varepsilon^{-1}t_s,\ell)}) - (1-t_s) \cdot O(\varepsilon\beta L D^2) - (1-t_s) D G\sqrt{2L}$.*

Proof. By repeating the proof of Lemma C.19 with Lemma D.5 taking the role of Lemma C.18, one can prove that

$$\begin{aligned}
&\sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)}) \\
&\qquad \geq (1-m)(\varepsilon i - t_s)(1-\varepsilon)^{i-1} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) + (1-\varepsilon)^{i-\varepsilon^{-1}t_s} \cdot \sum_{i=1}^{L} F_\ell(\mathbf{y}^{(\varepsilon^{-1}t_s,\ell)} \oplus \mathbf{z}^{(\varepsilon^{-1}t_s,\ell)}) \\
&\qquad\quad - (\varepsilon i - t_s) \cdot O(\varepsilon\beta L D^2) - (\varepsilon i - t_s) D G\sqrt{2L} \enspace .
\end{aligned}$$

By adding up the inequalities guaranteed by Lemma D.5 for all $i < i' \leq \varepsilon^{-1}$, we can also get

$$\begin{aligned}
&\sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(\varepsilon^{-1},\ell)} \oplus \mathbf{z}^{(\varepsilon^{-1},\ell)}) - \sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)}) \\
&\qquad = \sum_{i'=i+1}^{\varepsilon^{-1}} \sum_{\ell=1}^{\ell}[F_\ell(\mathbf{y}^{(i,\ell)} \oplus \mathbf{z}^{(i,\ell)}) - F_\ell(\mathbf{y}^{(i-1,\ell)} \oplus \mathbf{z}^{(i-1,\ell)})] \\
&\qquad \geq \sum_{i'=i+1}^{\varepsilon^{-1}} \{-\varepsilon^2\beta L D^2/2 - \varepsilon D G\sqrt{2L}\} = -(1-\varepsilon i)\varepsilon\beta L D^2/2 - (1-\varepsilon i) \cdot D G\sqrt{2L} \enspace .
\end{aligned}$$

The corollary now follows by adding up the two above inequalities. □

We are now ready to complete the proof of the approximation guarantee of Algorithm 2, which completes the proof of Proposition 5.2.

Lemma D.7. *It holds that*

$$\begin{aligned}
\sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(\varepsilon^{-1},\ell)} \oplus \mathbf{z}^{(\varepsilon^{-1},\ell)}) \geq (1-m) \cdot \max_{T \in [t_s,1]} \Bigg\{ \left[(T-t_s)e^{-T} + \frac{e^{-t_s-T} \cdot t_s^2}{2} - O(\varepsilon)\right] \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) \\
+ (e^{-T} - e^{-t_s-T} - O(\varepsilon)) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o})\Bigg\} - O(\varepsilon\beta L D^2) - O(D G\sqrt{L}) \enspace .
\end{aligned}$$

Proof. We prove below that the inequality stated in the lemma holds for any fixed $T \in [t_s, 1]$. Plugging the guarantee of Lemma D.4 into the guarantee of Corollary D.6 for $i = \lfloor \varepsilon^{-1} T \rfloor$ yields

$$\begin{aligned}
&\sum_{\ell=1}^{L} F_\ell(\mathbf{y}^{(\varepsilon^{-1},\ell)} \oplus \mathbf{z}^{(\varepsilon^{-1},\ell)}) \\
&\geq (1-m)(\varepsilon\lfloor \varepsilon^{-1}T \rfloor - t_s)(1-\varepsilon)^{\lfloor \varepsilon^{-1}T \rfloor - 1} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) - (\varepsilon\lfloor \varepsilon^{-1}T \rfloor - t_s) \cdot O(\varepsilon\beta L D^2) \\
&\quad - (1-t_s)DG\sqrt{2L} + (1-\varepsilon)^{\lfloor \varepsilon^{-1}T \rfloor - \varepsilon^{-1}t_s} \cdot \Bigg\{(1-m) \cdot \Bigg[(e^{-t_s} - e^{-2t_s} - O(\varepsilon)) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}) \\
&\quad + \left(\frac{e^{-2t_s} \cdot t_s^2}{2} - O(\varepsilon)\right) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}})\Bigg] - 63 t_s \varepsilon \beta L D^2 - 60 t_s DG\sqrt{L}\Bigg\} \\
&\geq (1-m)(T - t_s - \varepsilon)e^{-T} \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) - O(\varepsilon\beta L D^2) - O(DG\sqrt{L}) \\
&\quad + e^{t_s - T}(1-m)(1-\varepsilon)\Bigg[(e^{-t_s} - e^{-2t_s} - O(\varepsilon)) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}) + \left(\frac{e^{-2t_s} \cdot t_s^2}{2} - O(\varepsilon)\right) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}})\Bigg] \\
&= (1-m) \cdot \Bigg\{\Bigg[(T - t_s)e^{-T} + \frac{e^{-t_s - T} \cdot t_s^2}{2} - O(\varepsilon)\Bigg] \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o}_{\mathrm{D}}) \\
&\quad + (e^{-T} - e^{-t_s - T} - O(\varepsilon)) \cdot \sum_{\ell=1}^{L} F_\ell(\mathbf{o})\Bigg\} - O(\varepsilon\beta L D^2) - O(DG\sqrt{L}) \ . \qquad \square
\end{aligned}$$

E Additional Applications

In this section, we present two applications (in addition to the one given in Section 6).

E.1 Location Summarization

In this experimental setting, our objective is to create a summary of the locations around the current user location based on the Yelp dataset, a subset of Yelp's businesses, reviews, and user data covering information about local businesses in 11 metropolitan areas [Yelp, 2019]. We employ the formalization of this task used by Mualem & Feldman [2023]. Specifically, symmetry scores between the various locations have been generated using the methodology introduced by Kazemi et al. [2021] based on features extracted from location descriptions and associated user reviews, encompassing aspects like parking availability, WiFi access, vegan menu offerings, delivery options, suitability for outdoor seating, and being conducive to group gatherings. Then, assuming the set of locations is denoted by $[n]$, $M_{i,j}$ represents the similarity score between locations i and j, and d_i is the distance of location i from the user (measured in units of 200KM), the objective function for every set $S \subseteq [n]$ is given by

$$f(S) = \tfrac{1}{n}\sum_{i=1}^{n} \max_{j \in S} M_{i,j} - \sum_{i \in S} d_i \ .$$

Intuitively, this objective function favors sets S of locations that effectively summarizes the existing locations while remaining close to the user's current location.

Since the methods developed in this work are tailored for continuous functions, they cannot be used to directly optimize f. Thus, we need to invoke the multilinear extension[9] F of f defined as follows. For integers $i, j, j' \in [n]$, we write $M_{i,j} \prec M_{i,j'}$ if $M_{i,j} < M_{i,j'}$ or $M_{i,j} = M_{i,j'}$ and $j < j'$. Then, for every vector $\mathbf{x} \in [0,1]^n$,

$$F(\mathbf{x}) = \tfrac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\Bigg[x_j M_{i,j} \cdot \prod_{j' \mid M_{i,j} \prec M_{i,j'}} (1 - x_{j'})\Bigg] - \sum_{i=1}^{n} x_i d_i \ .$$

The multilinear extension F is amenable to optimization using our methods since the submodularity of f implies that its multi-linear extension F is DR-submodular [Bian *et al.*, 2017b]. Furthermore, the solution obtained in this way for F can be

[9] See, for example, [Buchbinder and Feldman, 2018] for the definition of the multi-linear extension.

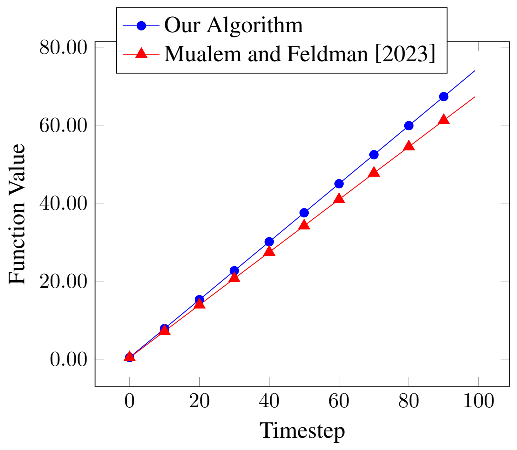


Figure 3: Results of the Location Summarization Experiment

rounded into a discrete solution with the same approximation for f using either pipage or swap rounding [Calinescu *et al.*, 2011; Chekuri *et al.*, 2010].

Our experiment closely follows the one of [Mualem and Feldman, 2023] by focusing on a single metropolitan area (Charlotte) and considering a time horizon of 100 steps. Each time step is associated with a different user u whose location is chosen uniformly at random within the rectangle encompassing the metropolitan area. If we denote by F_u the function F computed based on the location of user u, then in the time step associated with u, our objective is to select a vector $\mathbf{x}^{(u)}$ that maximizes F_u among all vectors satisfying $\|\mathbf{x}\|_1 \in [1, 2]$ (we seek solutions incorporating either 1 or 2 locations). As this optimization should be done before learning the location of u for prompt responses and privacy reasons, online optimization algorithms are used for the task. Akin to Section 6.1, we compare the performance in this context of our algorithm from Section 5 with the algorithm proposed by Mualem & Feldman [2023] when the number of online linear optimizers is set to $L = 100$ for both algorithms. As our algorithm requires a decomposition of the feasible polytope into a down-closed polytope $\mathcal{K}_{\mathrm{D}}$ and a general polytope $\mathcal{K}_{\mathrm{N}}$, we chose the decomposition $\mathcal{K}_{\mathrm{D}} = \{\mathbf{x} \in [0, 1]^n \mid \|\mathbf{x}\|_1 \leq 1\}$ and $\mathcal{K}_{\mathrm{N}} = \{\mathbf{x} \in [0, 1]^n \mid \|\mathbf{x}\|_1 = 1\}$ (note that $(\mathcal{K}_{\mathrm{D}} + \mathcal{K}_{\mathrm{N}}) \cap [0, 1]^n$ is indeed the original feasible polytope). The results of our experiments are illustrated in Figure 3, and demonstrate that our algorithm consistently outperforms the state-of-the-art algorithm of [Mualem and Feldman, 2023].

E.2 Quadratic Programming

In this section, we showcase the ability of our algorithm to interpolate between down-closed and general convex bodies. To demonstrate this, we consider a setting with a down-closed polytope constraint, and compare the performance of our method with to two other algorithms: the non-monotone Franke-Wolfe algorithm proposed by Bian et al. [2017a], which was designed for down-closed convex bodies and achieves e^{-1}-approximation for maximizing DR-submodular functions over such constraints (but is not well-defined when the constraint is not down-closed), and the non-monotone Franke-Wolfe algorithm suggested by Mualem & Feldman [2023] for general convex-body constraints. In a nutshell, the experiments we discuss below show that, despite the ability of our algorithm to optimize over general convex-body constraint, it is able to outperform the algorithm of [Mualem and Feldman, 2023] and recover the performance of the algorithm of [Bian *et al.*, 2017a] (and sometimes even slightly outperform it).

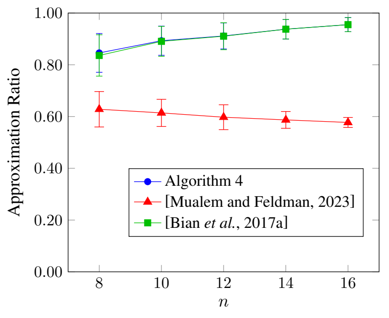
Every instance of the setting we consider is defined by two matrices $\mathbf{A} \in \mathbb{R}^{m \times n}_{\geq 0}$ and $\mathbf{H} \in \mathbb{R}^{m \times n}_{\leq 0}$ (the details of the setting closely follow settings considered by [Bian *et al.*, 2017a; Mualem and Feldman, 2023]). These matrices are chosen at random according to distributions described in Sections E.2 and E.2. However, before getting to the description of these distributions, let us explain how the instance is constructed based on the matrices $\mathbf{A}$ and $\mathbf{H}$ obtained. First, the down-closed polytope constraint is given by

$$\mathcal{K} = \{\mathbf{x} \in \mathbb{R}^n_{\geq 0} \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \leq \mathbf{u}\} \enspace ,$$

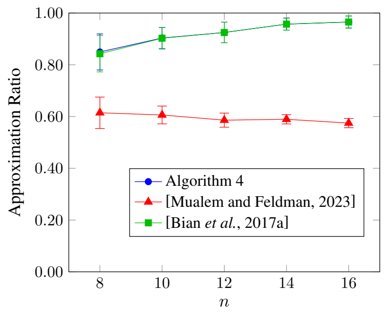
where $\mathbf{b}$ is the all-ones vector, and the upper bound vector $\mathbf{u}$ is given by $u_j = \min_{i \in [m]} b_i / A_{i,j}$ for each $j \in [n]$. The function F to be maximized subject to $\mathcal{K}$ is given, for every vector $\bar{0} \leq \mathbf{x} \leq \mathbf{u}$, by

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x} + c \enspace ,$$

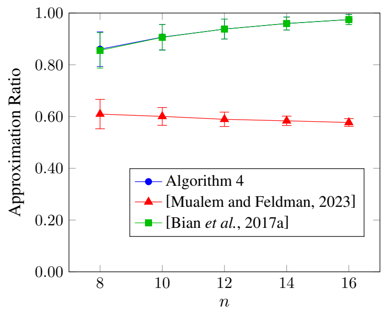
where $\mathbf{h}$ is a vector and c is a scalar. The non-positivity of the matrix $\mathbf{H}$ ensures that F is DR-submodular. Additionally, we set $\mathbf{h} = -0.1 \cdot \mathbf{H}^T\mathbf{u}$. Finally, to make F non-negative, it is necessary to set the value of c to be at least



(a) $m = \lfloor 0.5n \rfloor$



(b) $m = n$



(c) $m = \lfloor 1.5n \rfloor$

Figure 4: Quadratic Programming with Uniform Distribution

$M = -\min_{\bar{0} \leq \mathbf{x} \leq \mathbf{u}} \left(\frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{h}^T\mathbf{x}\right)$. The value of M can be approximately calculated via QUADPROGIP[10] [Xia *et al.*, 2020], and we set c to be $M + 0.1|M|$, which is a bit larger than the necessary minimum.

Uniform Distribution

In this section, we we use a method for choosing the matrices $\mathbf{H}$ and $\mathbf{A}$ that utilizes uniform distributions. In this method, the matrix $\mathbf{H} \in \mathbb{R}^{n \times n}$ is a symmetric matrix randomly generated by drawing each entry independently and uniformly from the interval $[-1, 0]$. Similarly, the matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is generated with entries randomly drawn from the interval $[v, v + 1]$, where $v = 0.01$. Note that, by using a positive value for v, we ensure that the entries of $\mathbf{A}$ are all strictly positive.

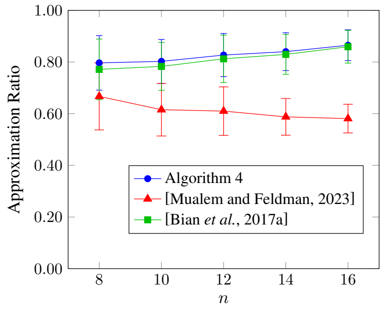
Our experiments based on the above distributions vary in the values chosen for the dimensions n and m. For each choice, we generated 100 instances and executed on them our offline algorithm (the version given as Algorithm 4 in Appendix C.3) as well as the algorithms of Mualem & Feldman [2023] and Bian et al. [2017a]. The number of iterations was set to 100 in all algorithms, which forces the the error control parameter ε to be set to 0.01 in our algorithm and the algorithm of Bian et al. [2017a], and to $^{\ln 2}/_{100}$ in the remaining algorithm. Additionally, since $\mathcal{K}$ is down-closed, the decomposition used by our algorithm is simply $\mathcal{K}_{\text{D}} = \mathcal{K}$ and $\mathcal{K}_{\text{N}} = \{0\}$. Figure 4 depicts the results obtained by the three algorithms, averaged over the 100 instances generated. The x-axis in each plot represents the value of n, and the value of m was derived based on n as specified by each plot caption. The y-axis illustrates the approximation ratios of the various algorithms in comparison to the optimum value calculated using a quadratic programming solver. Notably, our proposed algorithm demonstrates near-identical performance to the Frank-Wolfe algorithm suggested by Bian et al. [2017a], and clearly surpasses the algorithm suggested by Mualem & Feldman [2023].

Exponential Distribution

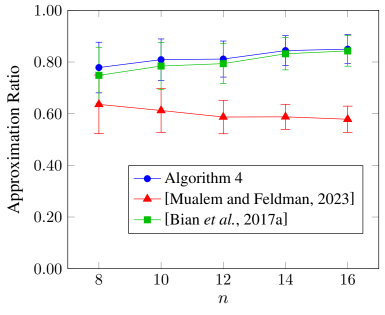
In this section, we use a different method for choosing the matrices $\mathbf{H}$ and $\mathbf{A}$ that utilizes exponential distributions. For every value $\lambda > 0$, the exponential distribution $\exp(\lambda)$ is defined by a density function assigning a density of $\lambda e^{-\lambda y}$ for $y \geq 0$, and a density of 0 for $y < 0$. Given this definition, $\mathbf{H} \in \mathbb{R}^{n \times n}_{\leq 0}$ is a symmetric matrix randomly generated with entries drawn independently from $-\exp(1)$, while $\mathbf{A} \in \mathbb{R}^{m \times n}_{\geq 0}$ is a randomly generated matrix with entries drawn independently from $\exp(0.25) + 0.01$.

For this method of generating $\mathbf{H}$ and $\mathbf{A}$, we conducted the same set of experiments as for the previous approach. The results of these experiments, again averaged over 100 independently chosen instances, are illustrated in Figure 5. Our proposed algorithm demonstrates slightly better performance than the non-monotone Frank-Wolfe algorithm suggested by Bian et al. [2017a], and surpasses the algorithm suggested by Mualem & Feldman [2023].

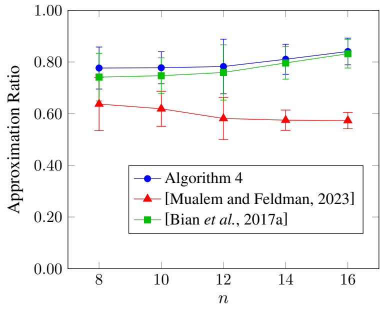
[10]We used IBM CPLEX optimization studio https://www.ibm.com/products/ilog-cplex-optimization-studio.



(a) $m = \lfloor 0.5n \rfloor$



(b) $m = n$



(c) $m = \lfloor 1.5n \rfloor$

Figure 5: Quadratic Programming with Exponential Distribution

Statement of Authorship

Principal Author

Title of Paper	Submodular Minimax Optimization: Finding Effective Sets		
Publication Status	[✓ Published] ☐ Accepted for Publication ☐ Submitted for Publication		
Publication Details	Proceedings of The 27th International Conference on Artificial Intelligence and Statistics, PMLR 238:1081-1089, 2024.		
Name of Principal Author (Candidate)	Loay Mualem		
Contribution to the Paper	Contributed to the theory, implementation and writing of the paper.		
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Name and Signature	Loay Mualem [illegible]	Date	24/6/2025

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the dissertation

Name of Co-Author	Ethan R. Elenberg		
Contribution to the Paper	Contributed to the writing and implementation of the paper.		
Name and Signature		Date	

Name of Co-Author	Moran Feldman		
Contribution to the Paper	Contributed to the theory and writing of the paper.		
Name and Signature	Moran Feldman	Date	25.6.2025

Name of Co-Author	Amin Karbasi		
Contribution to the Paper	Contributed to the theory and writing of the paper.		
Name and Signature		Date	

Please cut and paste additional co-author panels here as required.

Chapter 6

Sumbodular Minimax Optimization: Finding Effective Sets

In this chapter, we introduce and investigate the problem of submodular minimax optimization, a generalization of submodular maximization. Among its applications is modeling competition between multiple submodular functions, and robust maximization of a submodular function under uncertainty. We propose several algorithms with provable theoretical guarantees, tailored to different settings. Our framework has broad applicability, including prompt engineering, robust model training, and adversarial machine learning.

The following paper [MEFK24] was published at the *International Conference on Artificial Intelligence and Statistics (AISTATS 2024)*.

Submodular Minimax Optimization: Finding Effective Sets

Loay Mualem University of Haifa
Ethan R. Elenberg ASAPP
Moran Feldman University of Haifa
Amin Karbasi Yale University

Abstract

Despite the rich existing literature about minimax optimization in continuous settings, only very partial results of this kind have been obtained for combinatorial settings. In this paper, we fill this gap by providing a characterization of submodular minimax optimization, the problem of finding a set (for either the min or the max player) that is effective against every possible response. We show when and under what conditions we can find such sets. We also demonstrate how minimax submodular optimization provides robust solutions for downstream machine learning applications such as (i) prompt engineering in large language models, (ii) identifying robust waiting locations for ride-sharing, (iii) kernelization of the difficulty of instances of the last setting, and (iv) finding adversarial images. Our experiments show that our proposed algorithms consistently outperform other baselines.

1 INTRODUCTION

Many machine learning tasks, ranging from data selection to decision making, are inherently combinatorial and thus, require combinatorial optimization techniques that work at scale. Even though, in general, solving such problems is notoriously hard, practical problems are very often endowed with extra structures that lend them to optimization techniques. One common structure is submodularity, a condition that holds either exactly or approximately in a wide range of machine learning applications, including: dictionary selection (Krause and Cevher, 2010), sparse recovery, feature selection (Das and Kempe, 2011), neural network interpretability (Elenberg et al., 2017), crowd teaching (Singla et al., 2014), human brain mapping (Salehi et al., 2017), data summarizarion (Lin and Bilmes, 2011; Mualem and Feldman, 2022a), among many others. Submodular functions are often considered to be discrete analogs of concave functions, and like concave functions they can be (approximately) maximized. At the same time, submodular functions can also be exactly minimized as they can be extended into an efficiently computable continuous convex function (known as the Lovász extension). These optimization properties of submodular functions has been often exploited in scalable machine learning algorithms.

While scalable optimization methods are desirable, they are not the only requirements for ML algorithm deployment. Very often, it is also important to get solutions that are robust with respect to noise, outliers, adversarial examples, etc. Problems looking for solutions that are robust with respect to worst-case scenarios have usually been expressed as minimax optimization. Accordingly, recent years have witnessed a large body of work addressing minimax optimization in the continuous settings (see, e.g., Diakonikolas et al. (2021); Ibrahim et al. (2020); Lin et al. (2020); Mokhtari et al. (2020)). This line of research has given rise to a myriad of algorithms, and an ever increasing list of applications such as adversarial attack generation (Wang et al., 2021), robust statistics (Agarwal and Zhang, 2022) and multi-agent systems (Li et al., 2019), to name a few. To ensure feasibility of finding a saddle point, one has to make some structural assumptions. For instance, many of the above-mentioned works assume that the minimization is taken over a convex function, and the maximization over a concave function.

Despite the rich existing literature about minimax optimization in continuous settings, very few works have managed to obtain similar results for combinatorial settings. Staib et al. (2019) and Adibi et al. (2022) considered hybrid settings in which the maximization is done with respect to a (discrete) submodular function, but the minimization is still done over a continuous domain. Krause et al. (2008), Torrico et al. (2021) and

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Copyright 2024 by the author(s).

Iyer (2019, 2020) considered settings in which both the maximization and the minimization are discrete, but one of them is done over a small domain that can be efficiently enumerated. In this paper we provide the first systematic study of the natural case of fully discrete minimax optimization with maximization and minimization domains that can both be large. To the best of our knowledge, the only previous works relevant to this case are works of Bogunovic et al. (2017) and Orlin et al. (2018), who studied the maximization of a monotone submodular function subject to a cardinality constraint in the presence of a worst case (represented by a minimization) removal of a small number of elements from the chosen solution.

As submodular functions cannot be maximized exactly, there is no hope to get a saddle point in our setting. Instead, like Adibi et al. (2022), we take a game theoretic perspective on the setting. From this point of view, there are two players. Each player selects a set, and the objective function value is determined by the sets selected by both players. One of the players aims to minimize the objective function, while the other player wishes to maximize it. Our task is to select for one of the players (either the minimization or the maximization player) a set that is *effective* in the sense that it guarantees a good objective value regardless of the set chosen by the other player.

We map the tractability and approximability of the above minimax submodular optimization task as function of various properties, such as: the player considered (minimization or maximization), the constraints (if any) on the sets that can be chosen by the players, and whether the objective function is submodular as a whole, or for each player separately. We refer the reader to Section 2 for our exact results. However, in a nutshell, we have fully mapped the approximability for the minimization player, and we also have non-trivial results for the maximization player.

Our proposed algorithms for minimax submodular optimization can lead to finding of robust solutions for down-stream machine learning applications, including efficient prompt engineering, ride-share difficulty kernalization, adversarial attacks on image summarization and robust ride-share optimization. Empirical evaluation of our algorithms in the context of all the above applications can be found in Section 3.

1.1 Related Work

Submodular Minimization The first polynomial time algorithm for (unconstrained) submodular minimization was obtained by Grötschel et al. (1981) using the ellipsoids method. Almost twenty years later, Schrijver (2000) and Iwata et al. (2001) obtained, independently, the first strongly polynomial time (and combinatorial) algorithms for the problem. Further works have improved over the time complexities of the last algorithms, and the current state-of-the-art algorithm was described by Lee et al. (2015) (see also Axelrod et al. (2020) for a faster approximation algorithm for the problem).

All the above results apply to unconstrained submodular minimization. Unfortunately, constrained submodular minimization often (provably) admits only very poor approximation guarantees even when the constraint is as simple as a cardinality constraint (see, for example, Goel et al. (2010); Svitkina and Fleischer (2011)). Nevertheless, there are rare examples of constraints that allow for efficient submodular minimization, such as the constraint requiring the output set to be of even size (Goemans and Ramakrishnan, 1995).

Submodular Maximization A simple greedy algorithm obtains the optimal approximation ratio of $1 - 1/e$ for maximization of a monotone submodular function subject to a cardinality constraint (Nemhauser and Wolsey, 1978; Nemhauser et al., 1978). The same approximation ratio was later obtained for general matroid constraints via the continuous greedy algorithm (Călinescu et al., 2011). The best possible approximation ratio for unconstrained maximization of a non-monotone submodular function is $1/2$ (Feige et al., 2011; Buchbinder et al., 2015), even for deterministic algorithms (Buchbinder and Feldman, 2018). However, the approximability of constrained maximization of such functions is not as well understood. Following a long line of works (Buchbinder et al., 2014; Ene and Nguyen, 2016; Feldman et al., 2011; Lee et al., 2009; Oveis Gharan and Vondrák, 2011; Vondrák, 2013), the state-of-the-art algorithm for maximizing a non-monotone submodular function subject to a cardinality or matroid constraint guarantees 0.385-approximation (Buchbinder and Feldman, 2019), while the best inapproximability result for these constraints only shows that one cannot obtain 0.478-approximation for them (Gharan and Vondrák, 2011; Qi, 2022).

It is also worth mentioning a line of work (Mirzasoleiman et al., 2017; Mitrovic et al., 2017) aiming to find a small core set such that even if some elements are adversarially chosen for deletion, it is still possible to produce a good solution based on the core set. Note that this line of work differs from the maximization player point of view in our setting, in which the aim is to find a single solution for the maximization player that is good against every choice of the minimization player.

Additional related work relevant to some of our applications can be found in Appendix A.

2 NOTATION AND OUR THEORETICAL CONTRIBUTION

Let us describe the formal model for our setting. There are two (disjoint) ground sets $\mathcal{N}_1$ and $\mathcal{N}_2$, one ground set for each one of the players. For each ground set $\mathcal{N}_i$, we also have a constraint $\mathcal{F}_i \subseteq 2^{\mathcal{N}_i}$ specifying the sets that can be chosen from this ground set. Finally, there is a non-negative objective set function $f\colon 2^{\mathcal{N}_1 \uplus \mathcal{N}_2} \to \mathbb{R}_{\geq 0}$.[1] The minimization player gets to pick a set X from $\mathcal{F}_1$, and wishes to minimize the value of f, while the maximization players picks a set Y from $\mathcal{F}_2$, and aims to maximize the value of f. Our task is to find for each player a set S that yields the best value for f assuming the other player chooses the best response against S. In other words, for the minimization player we want to find a set $X \in \mathcal{F}_1$ that (approximately) minimizes $\max_{Y \in \mathcal{F}_2} f(X \uplus Y)$, and for the maximization player we should find a set $Y \in \mathcal{F}_2$ that (approximately) maximizes $\min_{X \in \mathcal{F}_1} f(X \uplus Y)$. As optimization of general set functions cannot be done efficiently, we must assume that the objective function f obeys some properties. Two common properties that are often considered in the literature are submodularity and monotonicity. However, to assume these properties, we first need to discuss what they mean in our setting.

Let us begin with the property of submodularity. In the following, given an element u and a set S we use $f(u \mid S) \triangleq f(S \cup \{u\}) - f(S)$ to denote the marginal contribution of u to the set S.[2] According to the standard definition of submodularity,[3] f is submodular if the inequality $f(u \mid S) \geq f(u \mid T)$ holds for every two sets $S \subseteq T \subseteq \mathcal{N}_1 \uplus \mathcal{N}_2$ and element $u \in (\mathcal{N}_1 \uplus \mathcal{N}_2) \setminus T$. Since this definition of submodularity treats $\mathcal{N}_1$ and $\mathcal{N}_2$ as two parts of one ground set, in the rest of this paper we call a function that obeys it *jointly-submodular*. However, since the ground sets $\mathcal{N}_1$ and $\mathcal{N}_2$ play very different roles in our problems, it makes sense to consider also functions that are submodular when restricted to one ground set. We say that f is submodular when restricted to $\mathcal{N}_1$ if it becomes a submodular function when we fix the set of elements of $\mathcal{N}_2$ chosen. More formally, f is submodular when restricted to $\mathcal{N}_1$ if the inequality $f(u \mid S \cup A_2) \geq f(u \mid T \cup A_2)$ holds for every $S \subseteq T \subseteq \mathcal{N}_1$ and $u \in \mathcal{N}_1 \setminus T, A_2 \subseteq \mathcal{N}_2$. The definition of being submodular when restricted to $\mathcal{N}_2$ is analogous, and we say that f is *disjointly-submodular* if it is submodular when restricted to either $\mathcal{N}_1$ or $\mathcal{N}_2$.

[1]We use $\uplus$ to denote the union of disjoint sets.

[2]Similarly, given sets S and T, $f(T \mid S) \triangleq f(S \cup T) - f(S)$ denotes the marginal contribution of T to S.

[3]A set function $g\colon \mathcal{N} \to \mathbb{R}$ is submodular if $g(u \mid S) \geq g(u \mid T)$ for every $S \subseteq T \subseteq \mathcal{N}$ and $u \in \mathcal{N} \setminus T$.

Unfortunately, submodular minimization admits very poor approximation guarantees even subject to simple constraints such as cardinality (see Section 1.1 for more details). Therefore, we restrict attention to the case of $\mathcal{F}_1 = 2^{\mathcal{N}_1}$. Given this restriction, we cannot assume that f is monotone[4] since this will guarantee that the best choice for the set X is always $\varnothing$. However, some of our results assume that f is monotone with respect to the elements of $\mathcal{N}_2$. In other words, we say that f is $\mathcal{N}_2$*-monotone* if the inequality $f(u \mid S) \geq 0$ holds for every $S \subseteq \mathcal{N}_1 \uplus \mathcal{N}_2$ and $u \in \mathcal{N}_2 \setminus S$.

Table 1 summarizes the theoretical results proved in this paper. When the table states that we have an inapproximability result of c for a problem, it means that no polynomial time algorithm can produce a value that with probability at least 2/3 approximates the exact value of this problem up to a factor of c. For example, if we look at the optimization problem $\min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y)$, then an inapproximability result of c means that no polynomial time algorithm can produce a value v that obeys

$$v \leq c \cdot \min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y) \leq c \cdot v \tag{1}$$

with probability at least 2/3. In contrast, we hold our algorithms to a higher standard. Specifically, when Table 1 states that we have a c-approximation algorithm for a problem, it means that the algorithm is able to produce with probability at least 2/3 two things: a value v of the above kind, and a solution set S for the external min or max operation that leads to c-approximation when the internal min or max is optimality solved. For example, given the above optimization problem, a c-approximation algorithm produces with probability at least 2/3 a value v obeying Equation (1), and a set $S \subseteq \mathcal{N}_1$ such that $\min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y) \leq \max_{Y \in \mathcal{F}_2} f(S \uplus Y) \leq c \cdot \min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y)$.

The success probability of 2/3 in the above definitions can always be increased via repetitions. However, such repetitions can usually be avoided since our algorithms are typically either deterministic or naturally have a high success probability.

As is standard in the literature, we assume that access to the objective function f is done via a value oracle that given a set $S \subseteq \mathcal{N}_1 \uplus \mathcal{N}_2$ returns $f(S)$. Furthermore, given a set S and element u, we use $S + u$ and $S - u$ to denote $S \cup \{u\}$ and $S \setminus \{u\}$, respectively.

2.1 Results for $\max\min$ Optimization

For $\max\min$ expressions (the problem of the maximization player) we have a good understanding of the

[4]A set function $g\colon 2^{\mathcal{N}} \to \mathbb{R}$ is monotone if $g(S) \leq g(T)$ for every two sets $S \subseteq T \subseteq \mathcal{N}$.

Table 1: Our theoretical results. We denote by α the approximation ratio that can be obtained for maximizing a non-negative submodular function subject to $\mathcal{F}_2$. If f happens to be $\mathcal{N}_2$-monotone, then α can be improved to be the approximation ratio that can be obtained for maximizing a non-negative *monotone* submodular function subject to $\mathcal{F}_2$.

Expression to approximate	**Assumptions**	**Result proved**
$\max_{Y\in\mathcal{F}_2}\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} Y)$	jointly-submodular	$(\alpha+\varepsilon)$-approx. alg. (Thm 2.1)
$\max_{Y\subseteq\mathcal{N}_2}\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} Y)$	disjointly-submodular	No finite approximation ratio possible unless $BPP = NP$ (Thms 2.2 and 2.3)
$\max_{Y\subseteq\mathcal{N}_2, \lvert Y\rvert\leq k}\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} Y)$	disjointly-submodular $\mathcal{N}_2$-monotone	
$\min_{X\subseteq\mathcal{N}_1}\max_{Y\subseteq\mathcal{N}_2} f(X \mathbin{\dot\cup} Y)$	disjointly-submodular	$(4+\varepsilon)$-approx. alg. (Thm 2.5)
$\min_{X\subseteq\mathcal{N}_1}\max_{Y\in\mathcal{F}_2} f(X \mathbin{\dot\cup} Y)$	jointly-submodular, $\varnothing\in\mathcal{F}_2$	$O(\alpha\sqrt{\lvert\mathcal{N}_1\rvert})$-approx. alg. (Thm. 2.6)
$\min_{X\subseteq\mathcal{N}_1}\max_{Y\in\mathcal{F}_2} f(X \mathbin{\dot\cup} Y)$	disjointly-submodular $\{u\}\in\mathcal{F}_2\ \forall u\in\mathcal{N}_2$	$O(\lvert\mathcal{N}_2\rvert)$-approx. alg. (Thm 2.4)

approximability, and it turns out that this approximability strongly depends on the kind of submodularity guaranteed for f. If f is jointly-submodular, then the problem admits roughly the same approximation ratio as the maximization problem obtained by omitting the min operation.

Theorem 2.1. *Assume that there exists an α-approximation algorithm ALG for the problem of maximizing a non-negative submodular function g subject to $\mathcal{F}_2$. Then, for every polynomially small $\varepsilon \in (0,\alpha]$, there exists a polynomial time algorithm that (i) outputs a set $\hat{Y}\in\mathcal{F}_2$ and the value $\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} \hat{Y})$; and (ii) guarantees that, with probability at least $2/3$, $\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} \hat{Y})$ falls within the range $[\tau/(\alpha+\varepsilon),\tau]$, where $\tau = \max_{Y\in\mathcal{F}_2}\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} Y)$. Furthermore, if f is $\mathcal{N}_2$-monotone, then it suffices for ALG to obtain α-approximation when g is guaranteed to be monotone (in addition to being non-negative and submodular).*

We note that by assuming in Theorem 2.1 that *ALG* is an α-approximation algorithm, we only mean that the expected value of the solution of *ALG* is smaller than the value of the optimal solution by at most a factor of α. In other words, we do not make any high probability assumption on *ALG*. The proof of Theorem 2.1 is based on the observation that the joint-submodularity of f implies that $\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} Y)$ is a submodular function of Y. See Section B.1 for details.

Unfortunately, it turns out that when f is only disjointly submodular, there is little an algorithm can guarantee. The following theorems show this for two basic special cases: unconstrained maximization, and maximization subject to a cardinality constraint of an $\mathcal{N}_2$-monotone function (the special case of unconstrained maximization of an $\mathcal{N}_2$-monotone function is trivial since it is always optimal to set $Y = \mathcal{N}_2$ in this case). Both theorems are proved using a reduction showing that the minimization over X can be replaced with a minimization over multiple functions, which allows us to capture well-known NP-hard problems with max min expressions. See Section B.2 for details.

Theorem 2.2. *When f is only guaranteed to be non-negative and disjointly submodular, no polynomial time algorithm for calculating $\max_{Y\subseteq\mathcal{N}_2}\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} Y)$ has a finite approximation ratio unless $BPP = NP$.*

Theorem 2.3. *When f is only guaranteed to be non-negative, $\mathcal{N}_2$-monotone and disjointly submodular, no polynomial time algorithm for calculating $\max_{Y\subseteq\mathcal{N}_2,\lvert Y\rvert\leq\rho}\min_{X\subseteq\mathcal{N}_1} f(X \mathbin{\dot\cup} Y)$, where ρ is a parameter of the problem, has a finite approximation ratio unless $BPP = NP$.*

2.2 Results for min max Optimization

We also have some results for min max expressions (the problem of the minimization player), although our understanding of the approximability of such expressions is worse than for max min expressions. We begin with the following theorem, which shows that when all the singleton subsets of $\mathcal{N}_1$ are feasible choices for the min operation (which is the case, for example, when the constraint is down-closed), it is possible to get a finite approximation (specifically, $O(\lvert\mathcal{N}_2\rvert)$-approximation) for min max. The proof of Theorem 2.4 can be found in Section C.1. In a nutshell, it is based on the observation for a set $X\subseteq\mathcal{N}_1$ the sum $f(X)+\sum_{u\in\mathcal{N}_2} f(X \mathbin{\dot\cup} \{u\})$ is an easy to calculate submodular function of X that gives $O(\lvert\mathcal{N}_2\rvert)$-approximation for $\max_{Y\subseteq\mathcal{N}_2} f(X \mathbin{\dot\cup} Y)$.

Theorem 2.4. *Assuming $\{u\}\in\mathcal{F}_2$ for every $u\in\mathcal{N}_2$, there is a polynomial time algorithm that, given a non-negative disjointly submodular function $f\colon 2^{\mathcal{N}}\to\mathbb{R}_{\geq 0}$, returns a set $\hat{X}\subseteq\mathcal{N}_1$ and a value v such that both $\max_{Y\in\mathcal{F}_2} f(\hat{X} \mathbin{\dot\cup} Y)$ and v fall within the range $[\tau,(\lvert\mathcal{N}_2\rvert+1)\cdot\tau]$, where $\tau \triangleq \min_{X\subseteq\mathcal{N}_1}\max_{Y\in\mathcal{F}_2} f(X \mathbin{\dot\cup} Y)$.*

The approximation ratio of the last theorem can be improved to a constant when the max operation is unconstrained (like the min operation). The following theorem states this formally, and its proof can be found in Section C.2. The proof is based on using samples of $\mathcal{N}_2$ to construct a random easy to calculate submodular function of X approximating $\max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y)$ up to a factor of roughly 4.

Theorem 2.5. *For every constant $\varepsilon \in (0,1)$, there exists a polynomial time algorithm that given a non-negative disjointly submodular function $f \colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ returns a set $\hat{X} \subseteq \mathcal{N}_1$ and a value v such that the expectations of both $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$ and v fall within the range $[\tau, (4+\varepsilon)\tau]$, where $\tau \triangleq \min_{X \subseteq \mathcal{N}_1} \max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y)$. Furthermore, the probability that both $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$ and v fall within this range is at least $1 - O(|\mathcal{N}_2|^{-1})$.*

The factor of $4+\varepsilon$ in the last theorem improves to $2+\varepsilon$ when f is symmetric with respect to $\mathcal{N}_2$, i.e., when $f(X \uplus Y) = f(X \uplus (\mathcal{N}_2 \setminus Y))$ for every two sets $X \subseteq \mathcal{N}_1$ and $Y \subseteq \mathcal{N}_2$.

It is interesting to note that the last two results show a separation between min max and max min optimization as no finite approximation guarantee can be obtained for disjointly submodular functions in the later case (Theorems 2.2 and 2.3). Our last result obtains a sub-linear approximation guarantee for an (almost) general constraint $\mathcal{F}_2$; however, this comes at the cost of requiring f to be jointly-submodular.

Theorem 2.6. *Assuming $\varnothing \in \mathcal{F}_2$, there exists a polynomial time algorithm that gets as input (i) a non-negative jointly submodular function $f \colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$, and (ii) an oracle that given a set $X \subseteq \mathcal{N}_1$ returns a set $Y \in \mathcal{F}_2$ that maximizes $f(X \uplus Y)$ up to a factor of $\alpha \geq 1$ among such sets,[5] and given this input returns a set $\hat{X} \subseteq \mathcal{N}_1$ and a value v such that both $\max_{Y \in \mathcal{F}_2} f(\hat{X} \uplus Y)$ and v are lower bounded by τ and upper bounded by $O(\alpha\sqrt{|\mathcal{N}_1|}) \cdot \tau$, where $\tau \triangleq \min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y)$.*

Below, we prove Theorem 2.6. In this proof, we denote by X^* an arbitrary set in $\arg\min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y)$. Notice that the definitions of X^* and τ imply together that $\tau = \max_{Y \in \mathcal{F}_2} f(X^* \uplus Y)$. Thus, $f(X^* \uplus Y) \leq \tau$ for every set $Y \in \mathcal{F}_2$, and in particular, since $\varnothing \in \mathcal{F}_2$ by assumption, $f(X^*) \leq \tau$.

[5]Theorem 2.6 assumes an oracle that never fails. Such an oracle can be implemented by a deterministic α-approximation algorithm, or a randomized algorithm that maximizes $f(X \uplus Y)$ up to a factor of α with high probability (in the later case, the algorithm guaranteed by the theorem also succeeds only with high probability). If one only has a randomized algorithm guaranteeing α-approximation in expectation, then repetitions should be used to get an oracle that maximizes $f(X \uplus Y)$ up to a factor of $\alpha + \varepsilon$ with high probability. Note that when $\varepsilon > 0$ is only polynomially small, this requires only a polynomial number of repetitions since we may assume that $\alpha \leq |\mathcal{N}_2|$ (otherwise, Theorem 2.4 already provides a better approximation).

Algorithm 1: Iterative X Growing

```
1 Use an algorithm for submodular minimization to
    find a set $X_0 \in \arg\min_{X \subseteq \mathcal{N}_1} f(X)$.
2 for $i = 1$ to $n_1 + 1$ do
3     Use the given oracle to find a set $Y_i \in \mathcal{F}_2$
        maximizing $f(X_{i-1} \uplus Y_i)$ up to a factor of $\alpha$
        among all sets in $\mathcal{F}_2$.
4     Use an algorithm for submodular minimization
        to find a set $X'_i \in$
        $\arg\min_{X \subseteq \mathcal{N}_1}[\sqrt{n_1} \cdot f(X \cup X_{i-1}) + f(X \uplus Y_i)]$.
5     if $X'_i \subseteq X_{i-1}$ then return the set $X_{i-1}$ and
        the value $\alpha \cdot f(X_{i-1} \uplus Y_i)$.
6     else Let $X_i \leftarrow X_{i-1} \cup X'_i$.
```

The algorithm that we use to prove Theorem 2.6 is Algorithm 1. Below, we use n_1 as a shorthand for $|\mathcal{N}_1|$, and use I to denote the number of iterations completed by the loop of this algorithm. Since the size of X_i increases following every completed iteration, $I \leq n_1$. Note that iteration $I+1$ started, but stopped before completion since X'_{I+1} was a subset of X_I. Hence, X_I is the output set of Algorithm 1. We begin the analysis of Algorithm 1 with the following lemma.

Lemma 2.7. *For every integer $1 \leq i \leq I$,*

$$f(X_i) = f(X_{i-1} \cup X'_i) \leq f((X^* \cap X'_i) \cup X_{i-1}) + \tau/\sqrt{n_1} \enspace .$$

Proof. By the choice of X'_i, we have

$$\begin{aligned} &\sqrt{n_1} \cdot f(X'_i \cup X_{i-1}) + f(X'_i \uplus Y_i) \\ \leq{}& \sqrt{n_1} \cdot f((X^* \cap X'_i) \cup X_{i-1}) + f((X^* \cap X'_i) \uplus Y_i) \\ \leq{}& \sqrt{n_1} \cdot f((X^* \cap X'_i) \cup X_{i-1}) + f(X^* \uplus Y_i) + f(X'_i \uplus Y_i) \\ \leq{}& \sqrt{n_1} \cdot f((X^* \cap X'_i) \cup X_{i-1}) + \tau + f(X'_i \uplus Y_i) \enspace , \end{aligned}$$

where the second inequality follows from the submodularity and non-negativity of f, and the last inequality follows from the definition of X^*. The lemma now follows by rearranging the last inequality. □

Corollary 2.8. *The output set X_I of Algorithm 1 obeys $f(X_I) \leq O(\sqrt{n_1}) \cdot \tau$.*

Proof. If $I = 0$, then $X_I = X_0$, and by definition we have $f(X_I) \leq f(X^*) \leq \tau$. Therefore, we may assume from now on $I \geq 1$. Using Lemma 2.7, we now get

$$\begin{aligned} f(X_I) - f(X_0) &\leq \sum_{i=1}^{I} [f(X_i) - f(X_{i-1})] \\ &\leq \sum_{i=1}^{I} [f(X^* \cap X'_i \mid X_{i-1}) + \tau/\sqrt{n_1}] \end{aligned}$$

$$\begin{aligned}
&\leq \sum_{i=1}^{I} f(X^* \cap X'_i \mid X^* \cap X_{i-1}) + \sqrt{n_1} \cdot \tau \\
&= f(X^* \cap X_I \mid X^* \cap X_0) + \sqrt{n_1} \cdot \tau \\
&\leq f(X^* \cap X_I) - f(X_0) + \sqrt{n_1} \cdot \tau \ ,
\end{aligned}$$

where the penultimate inequality holds by the submodularity of f and the observation that $I \leq n_1$, and the last inequality holds by the definition of X_0. Adding $f(X_0)$ to be both sides of the last inequality yields

$$\begin{aligned}
&f(X_I) - \sqrt{n_1} \cdot \tau \leq f(X^* \cap X_I) \leq f(X^*) + f(X_I) \\
&\quad - f(X^* \cup X_I) \leq \tau + f(X_I) - f(X^* \cup X_I) \ ,
\end{aligned}$$

where the second inequality follows from the submodularity of f, and last inequality follows from the definition of X^*. To lower bound the term $f(X^* \cup X_I)$, we observe that since $I \geq 1$, the definition of X'_I implies

$$\begin{aligned}
f(X^* \cup X_I) &= f((X^* \cup X'_I) \cup X_{I-1}) \\
&\geq f(X'_I \cup X_{I-1}) - \frac{f(X^* \mid X'_I \uplus Y_I)}{\sqrt{n_1}} \\
&\geq f(X_I) - \frac{f(X^*)}{\sqrt{n_1}} \geq f(X_I) - \tau/\sqrt{n_1} \ ,
\end{aligned}$$

where the second inequality uses the submodularity and non-negativity of f, and the last inequality holds by the definition of X^*. The corollary now follows by combining this inequality with the previous one. $\square$

We are now ready to prove Theorem 2.6.

Proof of Theorem 2.6. Since Algorithm 1 outputted the set X_I, we must have $X'_{I+1} \subseteq X_I$. Furthermore, by the choices of Y_{I+1} and X'_{I+1},

$$\begin{aligned}
&\alpha^{-1} \cdot \max_{Y \in \mathcal{F}_2} f(X_I \uplus Y) \\
&\leq f(X_I \uplus Y_{I+1}) = f(X_I) + f(Y_{I+1} \mid X_I) \\
&\leq f(X_I) + f(Y_{I+1} \mid X'_{I+1}) \\
&\leq f(X_I) + [\sqrt{n_1} \cdot f(X^* \cup X_I) + f(X^* \uplus Y_{I+1}) \\
&\quad - \sqrt{n_1} \cdot f(X'_{I+1} \cup X_I) - f(X'_{I+1})] \\
&\leq f(X_I) + \sqrt{n_1} \cdot f(X^* \mid X_I) + f(X^* \uplus Y_{I+1}) \ ,
\end{aligned}$$

where the second inequality follows from the submodularity of f, and the last inequality holds by the non-negativity of f. Observe now that Corollary 2.8 guarantees $f(X_I) \leq O(\sqrt{n}) \cdot \tau$, and the definition of X^* guarantees $f(X^* \cup Y_{I+1}) \leq \tau$. Furthermore, using the submodularity and non-negativity of f, we also get $f(X^* \mid X_I) \leq f(X^*) \leq \tau$. Plugging all these observations into the previous inequality yields

$$\begin{aligned}
&\alpha^{-1} \cdot \max_{Y \in \mathcal{F}_2} f(X_I \uplus Y) \leq f(X_I \uplus Y_{I+1}) \\
&\leq O(\sqrt{n_1}) \cdot \tau + \sqrt{n_1} \cdot \tau + \tau = O(\sqrt{n_1}) \cdot \tau \ .
\end{aligned}$$

Multiplying the last inequality by α, we get the upper bound on $\max_{Y \in \mathcal{F}_2} f(X_I \uplus Y)$ and $\alpha \cdot f(X_I \uplus Y_{I+1})$ (the value outputted by Algorithm 1) promised in the theorem. The promised lower bound on these expressions also holds since the definition of Y_I implies

$$\begin{aligned}
&\alpha \cdot f(X_I \uplus Y_{I+1}) \geq \max_{Y \in \mathcal{F}_2} f(X_I \uplus Y) \\
&\geq \min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y) = \tau \ .
\end{aligned}$$

$\square$

3 APPLICATIONS

In this section and Appendix D we discuss five machine-learning applications: efficient prompt engineering, ride-share difficulty kernelization, adversarial attack on image summarization, robust ride-share optimization, and prompt engineering for dialog state tracking. Each one of these applications necessitates either max-min or min-max optimization on a jointly submodular function[6] (with a cardinality constraint on the maximization part). To demonstrate the robustness of our suggested methods in this work, we empirically compare them against a few benchmarks.

In the max-min optimization applications, we compare the algorithm from Theorem 2.1 (named below Min-as-Oracle) against 4 benchmarks: (i) "Random" choosing a random set of k elements from $\mathcal{N}_2$ as the set Y; (ii) "Max-Only" using a maximization algorithm to find the a set Y that is (approximately) optimal against $X = \varnothing$; (iii) "Top-k" selecting a set Y consisting of the top k singletons $y \in \mathcal{N}_2$, where each singleton is evaluated based on the corresponding worst case set X; and (iv) "Best-Response" simulating a best response dynamic between the minimization and maximization players, and outputting the set used by the maximization player after a given number of iterations. The Best-Response method is a widely used concept in game theory and optimization, first introduced in the seminal work by Von Neumann and Morgenstern (1947).

In the min-max optimization applications, we study the algorithm from Theorem 2.4 (named below Min-by-Singletons) and a slightly modified version of the algorithm from Theorem 2.6 (named below Iterative-X-Growing). Out of the above 4 benchmarks, the Random and Best-Response benchmarks still make sense in min-max settings with the natural adaptations. It was also natural to try to replace the Max-Only benchmark with a "Min-Only" benchmark, but such a benchmark would always output the empty set in our applications. Thus, we use instead a benchmark called "Max-and-then-Min" that returns a set X that is optimal against

[6]We consider only jointly submodular functions in our experiments since our theoretical results for disjointly submodular functions are, unfortunately, mostly negative.

the set Y returned by Max-Only. See Appendix E for further detail about the various benchmarks, and the implementations of our algorithms.

3.1 Efficient Prompt Engineering

Consider the problem of designing efficient prompts for zero-shot in-context learning. Following Si et al. (2023), we consider an open-domain question answering task: the goal is to answer questions from the SQuAD dataset (Rajpurkar et al., 2016) by prompting a large language model with k relevant passages of text taken from a large corpus of Wikipedia articles. To get for each question an initial set of relevant candidate passages, 21 million Wikipedia passages were embedded using a pretrained Contriever model (Izacard et al., 2022) and indexed using FAISS.[7] Then, for each question, the top 100 passages were kept as candidates.

Large language models such as OpenAI's ChatGPT offer very impressive performance on natural language tasks via a public API. As the cost of making a prediction depends on the length of the input prompt, we propose to reduce the cost by *jointly answering* similar questions with a common prompt, and thus, a single query to the GPT-3.5-turbo language model. To use this approach, we need to select a subset of passages that are effective on the set of *answerable* questions, which we formulate as a combinatorial optimization problem. Specifically, let $\mathcal{N}_1$ be a batch of questions and let $\mathcal{N}_2$ be the union of all candidate passages. (In general, $100 \leq |\mathcal{N}_2| \leq 100 \cdot |\mathcal{N}_1|$ since there may be significant overlap among candidates for questions on the same topic.) Let $0 \leq s_{u,v} \leq 1$ be the cosine similarity between passage embedding u and question embedding v. Then, we define

$$\begin{aligned} f(X \uplus Y) = {} & \sum_{v \in \mathcal{N}_1 \setminus X} \max_{u \in Y} s_{u,v} \\ & + \beta \cdot \sum_{u \in \mathcal{N}_1 \setminus X} \sum_{v \in Y} s_{u,v} + \lambda \cdot |X| \enspace . \end{aligned} \tag{2}$$

Here $\lambda \geq 0$ and $\beta \geq 0$ are regularization parameters. The first term represents how well the passages of Y cover the questions in $\mathcal{N}_1 \setminus X$. For small values of β, the second term ensures f increases in $|Y|$, and the last term controls the size of X. The following lemma is proved in Section F.1.

Lemma 3.1. *The objective function* (2) *is a non-negative jointly-submodular function.*

By solving the max-min optimization $\max_{Y \subseteq \mathcal{N}_2, |Y| \leq k} \min_{X \in \mathcal{N}_1} f(X \uplus Y)$, we get the set X of answerable questions, and a small set Y of effective passages. In our experiments we set $\beta = 10^{-3}$, $\lambda = 0.8$, and $k = 10$, and we grouped the SQuAD test set into batches of 25 questions. In addition to the heterogeneity introduced by crude batching, we removed $\delta = 25\%$ of the candidates from $\mathcal{N}_2$, leading to some questions having no relevant passages.

Table 2 shows the performance of the prompts for GPT-3.5-turbo obtained by Min-as-Oracle and various benchmarks. Each method is evaluated in terms of exact match accuracy and F1 score between predicted and ground truth answers. As a baseline, we also consider using the common prompt returned by the retrieval algorithm, but making a *separate prediction* for each question in the cluster. We see our proposed joint prediction with a single prompt increases accuracy while on average requiring only 5.3% of the tokens per question compared to separate prediction. Moreover, Min-as-Oracle has the highest accuracy among all retrieval algorithms used for joint prediction. Figure 1 shows a qualitative example of joint prediction for a batch of questions.

3.2 Ride-Share Difficulty Kernelization

Consider a regulator overseeing the taxi companies licensed to operate within a given city. The regulator wants to make sure that the taxi companies give a fair level of service to all parts of the city, rather than concentrating on the most profitable neighborhoods. However, checking that this is indeed the case is not trivial since often the limited number of taxis available implies that some locations must remain poorly served. Our objective in this section is to give the regulator a small set (kernel) of locations that that capture the difficulty of the problem faced by the taxi company in the sense that the locations in the set cannot be served well (on average) regardless of how the taxi companies choose the waiting locations for their taxis.

Formally, given a set $\mathcal{N}_1$ of (client) pickup locations, and a set $\mathcal{N}_2$ of potential waiting locations for taxies, we define the following score function to capture the convenience of serving all the locations of $\mathcal{N}_1 \setminus X$ by locating taxis at locations Y.[8]

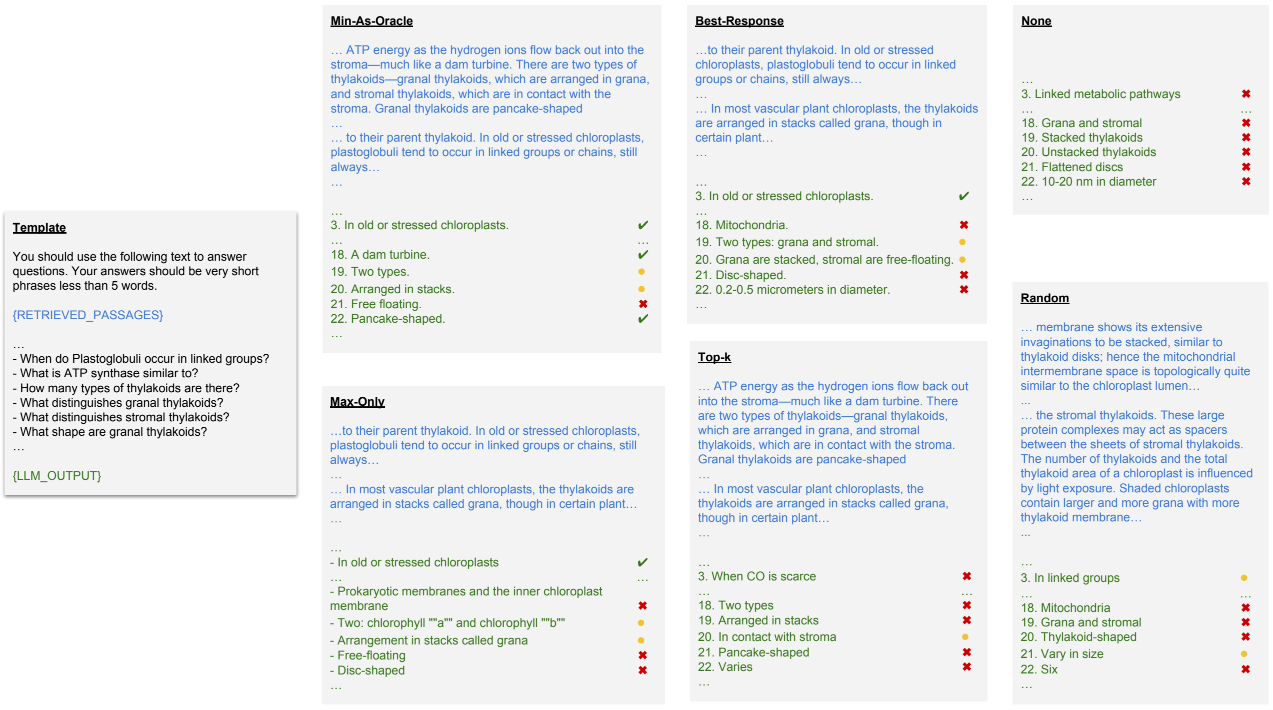
$$\begin{aligned} f(X \uplus Y) = \sum_{v \in \mathcal{N} \setminus X} \max_{u \in Y} s_{u,v} - \frac{1}{|\mathcal{N}_2|} \sum_{u \in Y} \sum_{v \in Y} s_{u,v} \\ + \lambda \cdot |X| \enspace . \end{aligned} \tag{3}$$

Here, $s_{u,v}$ is a "convenience score" which, given a customer location $u = (x_u, y_u)$ and a waiting driver location $v = (x_v, y_v)$,[9] represents the ease of accessing u

[7] https://github.com/facebookresearch/faiss

[8] It would have been more natural to define X as the set of locations to service. However, this would have resulted in an objective function that is only disjointly submodular.

[9] Each location is specified by a (latitude, longitude) coordinate pair.



Template

You should use the following text to answer questions. Your answers should be very short phrases less than 5 words.

{RETRIEVED_PASSAGES}

…
- When do Plastoglobuli occur in linked groups?
- What is ATP synthase similar to?
- How many types of thylakoids are there?
- What distinguishes granal thylakoids?
- What distinguishes stromal thylakoids?
- What shape are granal thylakoids?
…

{LLM_OUTPUT}

Min-As-Oracle

… ATP energy as the hydrogen ions flow back out into the stroma—much like a dam turbine. There are two types of thylakoids—granal thylakoids, which are arranged in grana, and stromal thylakoids, which are in contact with the stroma. Granal thylakoids are pancake-shaped
…
… to their parent thylakoid. In old or stressed chloroplasts, plastoglobuli tend to occur in linked groups or chains, still always…
…

…
3. In old or stressed chloroplasts. ✔
… …
18. A dam turbine. ✔
19. Two types. ●
20. Arranged in stacks. ●
21. Free floating. ✖
22. Pancake-shaped. ✔
…

Max-Only

…to their parent thylakoid. In old or stressed chloroplasts, plastoglobuli tend to occur in linked groups or chains, still always…
…
… In most vascular plant chloroplasts, the thylakoids are arranged in stacks called grana, though in certain plant…
…

…
- In old or stressed chloroplasts ✔
… …
- Prokaryotic membranes and the inner chloroplast membrane ✖
- Two: chlorophyll ""a"" and chlorophyll ""b"" ●
- Arrangement in stacks called grana ●
- Free-floating ✖
- Disc-shaped ✖
…

Best-Response

…to their parent thylakoid. In old or stressed chloroplasts, plastoglobuli tend to occur in linked groups or chains, still always…
…
… In most vascular plant chloroplasts, the thylakoids are arranged in stacks called grana, though in certain plant…
…

…
3. In old or stressed chloroplasts. ✔
…
18. Mitochondria. ✖
19. Two types: grana and stromal. ●
20. Grana are stacked, stromal are free-floating. ●
21. Disc-shaped. ✖
22. 0.2-0.5 micrometers in diameter. ✖
…

Top-k

… ATP energy as the hydrogen ions flow back out into the stroma—much like a dam turbine. There are two types of thylakoids—granal thylakoids, which are arranged in grana, and stromal thylakoids, which are in contact with the stroma. Granal thylakoids are pancake-shaped
…
… In most vascular plant chloroplasts, the thylakoids are arranged in stacks called grana, though in certain plant…
…

…
3. When CO is scarce ✖
… …
18. Two types ✖
19. Arranged in stacks ✖
20. In contact with stroma ●
21. Pancake-shaped ✖
22. Varies ✖
…

None

…
3. Linked metabolic pathways ✖
… …
18. Grana and stromal ✖
19. Stacked thylakoids ✖
20. Unstacked thylakoids ✖
21. Flattened discs ✖
22. 10-20 nm in diameter ✖
…

Random

… membrane shows its extensive invaginations to be stacked, similar to thylakoid disks; hence the mitochondrial intermembrane space is topologically quite similar to the chloroplast lumen…
…
… the stromal thylakoids. These large protein complexes may act as spacers between the sheets of stromal thylakoids. The number of thylakoids and the total thylakoid area of a chloroplast is influenced by light exposure. Shaded chloroplasts contain larger and more grana with more thylakoid membrane…
…

…
3. In linked groups ●
… …
18. Mitochondria ✖
19. Grana and stromal ✖
20. Thylakoid-shaped ✖
21. Vary in size ●
22. Six ✖
…

Figure 1: Example of our proposed max-min formulation for jointly answering a batch of questions from the SQuAD dataset using GPT-3.5-turbo. Template prompt (left), followed by excerpts from the retrieved passages (blue) and generated answers (green). Exact match, partial match, and incorrect answer are denoted ✔, ●, and ✖, respectively. Min-as-Oracle retrieved two passages that are relevant to Questions 3, 18, 19, 20, 21, and 22, while the other selection algorithms retrieved only one or neither of them. Consequently, **Min-As-Oracle is best aligned with the ground truth answers, having the highest number of exact matches and the fewest number of hallucinations.**

from v. Following Mitrovic et al. (2018), we set $s_{u,v} \triangleq 2 - \frac{2}{1+e^{-200d(u,v)}}$, where $d(u,v) = |x_u - x_v| + |y_u - y_v|$ is the Manhattan distance between the two points. The value $\lambda \in [0,1]$ is a regularization parameter whose use is discussed below. Some properties of this objective function are given by the next lemma, proved in Appendix F.2.

Lemma 3.2. *The objective function* (3) *is a non-negative jointly-submodular function.*

Recall that we are looking for a kernel set $\mathcal{N}_1 \setminus X$ of pickup locations that cannot be served well (on average) by any choice of k locations for taxis (k is determined by the number of taxis available). To do that, we need to solve the max-min optimization problem given by $\min_{X \subseteq \mathcal{N}_1} \max_{Y \subseteq \mathcal{N}_2, |Y| \leq k} f(X \cup Y)$. The regularization parameter λ can now be used to control the size the kernel set returned.

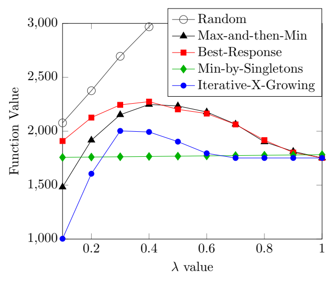
In our experiments for this application, we have used the Uber data set (Uber), which includes real-life Uber pickups in New York City during the month of April in the year 2014. To ensure computational tractability, in each execution of our experiments, we randomly selected from this data set a subset of $|\mathcal{N}_1| = 6{,}000$ pickup locations within the region of Manhattan. Then, we randomly selected a subset of 400 pickup locations from the set $\mathcal{N}_1$ to constitute the set $\mathcal{N}_2$ (we treat locations in $\mathcal{N}_1$ and $\mathcal{N}_2$ as distinct even if they are identical, to guarantee that $\mathcal{N}_1$ and $\mathcal{N}_2$ are disjoint).

In the first experiment, we fixed the maximum number of waiting locations to be 8, and varied λ. Figure 2a depicts the outputs for this experiment for Min-by-Singletons, Iterative-X-Growing (with $\beta = 0.5$) and three benchmarks (averaged over 10 executions of the experiment). One can observe that both Iterative-X-Growing and Min-by-Singletons surpasses the performance of all benchmarks for almost all values of λ. In both this experiment and the next one the standard error of the mean is less than 10 for all data points.

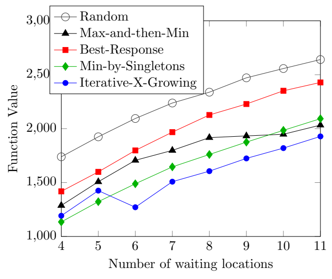
In the second experiment, we fixed λ to 0.2 and varied the number of allowed waiting locations. The results of this experiment are depicted by Figure 2b (averaged over 10 executions of the experiment). Once again, Iterative-X-Growing and Min-by-Singletons demonstrate superior performance compared to the bench-

Table 2: Open-domain question answering on SQuAD using GPT-3.5-turbo. Best values are in **bold**.

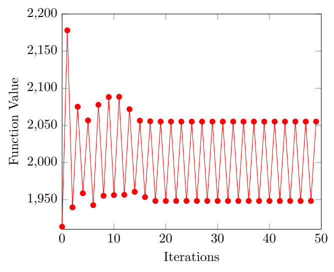
Prompting Method	Retrieval Algorithm	Exact Match % ↑	F1% ↑	Avg Tokens/Question ↓
Joint Prediction	Random	18.6	29.7	73.2
	None	22.5	33.9	20.0
	Top-k	25.0	37.0	**72.8**
	Max-Only	25.9	37.5	73.2
	Best-Response	25.6	37.0	73.2
	Min-as-Oracle	**26.1**	**37.8**	73.2
Separate Prediction	Random	9.7	17.8	1356.3
	None	15.9	29.1	40.1
	Top-k	21.3	31.6	1338.8
	Max-Only	25.3	36.4	1348.7
	Best-Response	25.2	36.2	1348.7
	Min-as-Oracle	25.2	36.3	1348.7



(a) Results for 8 waiting locations



(b) Results for $\lambda = 0.2$



(c) Behavior of Best-Response for $\lambda = 0.2$ and 8 waiting locations

Figure 2: Empirical results for ride-share difficulty kernelization. Figures (a) and (b) compare the performance of our algorithms Min-by-Singletons and Iterative-X-Growing with 3 benchmarks for different value of λ and bounds on the number of weighting locations. Figure (c) depicts the value of the output of the Best-Response method as a function of the number of iterations performed.

marks for almost all values of k. Please refer to Figure 6 in Appendix D.4 for a visual depiction of the results.

As the third experiment for this application, we conducted a more in depth analysis of Best-Response. Figure 2c graphically presents the objective function value obtained by a typical execution of Best-Response after a varying number of iterations (for $\lambda = 0.5$ and an upper bound of 20 on the number of waiting locations). It is apparent that Best-Response does not converge for this execution. Furthermore, both our suggested algorithms demonstrate better performance even with respect to the best performance of Best-Response for any number of iterations between 1 and 50.

4 CONCLUSION

In this paper we have initiated the systematical study of minimax optimization for combinatorial (discrete) settings with large domains. We have fully mapped the theoretical approximability of max-min submodular optimization, and also obtained some understanding of the approximability of min-max submodular optimization. The above theoretical work has been complemented with empirical experiments demonstrating the value of our technique for the machine-learning tasks of efficient prompt engineering, ride-share difficulty kernelization, adversarial attacks on image summarization, and robust ride-share optimization.

We hope future work will lead to a fuller understanding of minimax submodular optimization, and will also consider classes of discrete functions beyond submodularity. A natural class to consider in that regard is the class of weakly-submodular functions (Das and Kempe, 2011), which extends the class of submodular functions and has many machine learning applications (Khanna et al., 2017; Qian and Singer, 2019; Chen et al., 2018; El Halabi et al., 2022). However, minimax optimization of this class seems to be difficult because no algorithm is currently known even for plain minimization of weakly-submodular functions. Another open problem is to prove a performance guarantee for Best-Response.

5 ACKNOWLEDGMENTS

The work of Loay Mualem and Moran Feldman has been partially supported by Israel Science Foundation (ISF) grant number 459/20. Amin Karbasi acknowledges funding in direct support of this work from NSF (IIS-1845032) and the AI Institute for Learning-Enabled Optimization at Scale (TILOS).

References

Arman Adibi, Aryan Mokhtari, and Hamed Hassani. Minimax optimization: The case of convex-submodular. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 151 of *Proceedings of Machine Learning Research*, pages 3556–3580. PMLR, 2022. URL `https://proceedings.mlr.press/v151/adibi22a.html`.

Alekh Agarwal and Tong Zhang. Minimax regret optimization for robust machine learning under distribution shift. In Po-Ling Loh and Maxim Raginsky, editors, *Conference on Learning Theory (COLT)*, volume 178 of *Proceedings of Machine Learning Research*, pages 2704–2729. PMLR, 2022. URL `https://proceedings.mlr.press/v178/agarwal22b.html`.

Brian Axelrod, Yang P. Liu, and Aaron Sidford. Near-optimal approximate discrete and continuous submodular function minimization. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 837–853. SIAM, 2020. doi: 10.1137/1.9781611975994.51. URL `https://doi.org/10.1137/1.9781611975994.51`.

Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *SODA*, pages 1497–1514, 2014. doi: 10.1137/1.9781611973402.110. URL `https://doi.org/10.1137/1.9781611973402.110`.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021. URL `https://doi.org/10.5281/zenodo.5297715`.

Ilija Bogunovic, Slobodan Mitrovic, Jonathan Scarlett, and Volkan Cevher. Robust submodular maximization: A non-uniform partitioning approach. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 508–516. PMLR, 2017. URL `http://proceedings.mlr.press/v70/bogunovic17a.html`.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

Niv Buchbinder and Moran Feldman. Deterministic algorithms for submodular maximization problems. *ACM Trans. Algorithms*, 14(3):32:1–32:20, 2018. doi: 10.1145/3184990. URL `https://doi.org/10.1145/3184990`.

Niv Buchbinder and Moran Feldman. Constrained submodular maximization via a nonsymmetric technique. *Math. Oper. Res.*, 44(3):988–1005, 2019. doi: 10.1287/moor.2018.0955. URL `https://doi.org/10.1287/moor.2018.0955`.

Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In Chandra Chekuri, editor, *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1433–1452. SIAM, 2014. doi: 10.1137/1.9781611973402.106. URL `https://doi.org/10.1137/1.9781611973402.106`.

Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM J. Comput.*, 44(5):1384–1402, 2015. doi: 10.1137/130929205. URL `https://doi.org/10.1137/130929205`.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *EMNLP*, pages 5016–5026, 2018. doi: 10.18653/v1/D18-1547. URL `https://aclanthology.org/D18-1547`.

Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40 (6):1740–1766, 2011. doi: 10.1137/080733991. URL `https://doi.org/10.1137/080733991`.

Lin Chen, Moran Feldman, and Amin Karbasi. Weakly submodular maximization beyond cardinality constraints: Does randomization help greedy? In Jennifer Dy and Andreas Krause, editors, *International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 804–813. PMLR, Jul 2018.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. URL `https://arxiv.org/abs/2107.03374`.

Abhimanyu Das and David Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In Lise Getoor and Tobias Scheffer, editors, *International Conference on Machine Learning (ICML)*, pages 1057–1064. Omnipress, 2011. URL `https://icml.cc/2011/papers/542_icmlpaper.pdf`.

Jelena Diakonikolas, Constantinos Daskalakis, and Michael I. Jordan. Efficient methods for structured nonconvex-nonconcave min-max optimization. In Arindam Banerjee and Kenji Fukumizu, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 130 of *Proceedings of Machine Learning Research*, pages 2746–2754. PMLR, 2021. URL `http://proceedings.mlr.press/v130/diakonikolas21a.html`.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. A survey for in-context learning. *arXiv preprint* *arXiv:2301.00234*, 2022. URL https://arxiv.org/abs/2301.00234.

Marwa El Halabi, Suraj Srinivas, and Simon Lacoste-Julien. Data-efficient structured pruning via submodular optimization. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 36613–36626. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/ed5854c456e136afa3faa5e41b1f3509-Paper-Conference.pdf.

Ethan R. Elenberg, Alexandros G. Dimakis, Moran Feldman, and Amin Karbasi. Streaming weak submodularity: Interpreting neural networks on the fly. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4044–4054, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/c182f930a06317057d31c73bb2fedd4f-Abstract.html.

Alina Ene and Huy L. Nguyen. Constrained submodular maximization: Beyond $1/e$. In Irit Dinur, editor, *FOCS*, pages 248–257. IEEE Computer Society, 2016. doi: 10.1109/FOCS.2016.34. URL https://doi.org/10.1109/FOCS.2016.34.

Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing non-monotone submodular functions. *SIAM J. Comput.*, 40(4):1133–1153, 2011. doi: 10.1137/090779346. URL https://doi.org/10.1137/090779346.

Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In Rafail Ostrovsky, editor, *FOCS*, pages 570–579. IEEE Computer Society, 2011. doi: 10.1109/FOCS.2011.46. URL https://doi.org/10.1109/FOCS.2011.46.

Shayan Oveis Gharan and Jan Vondrák. Submodular maximization by simulated annealing. In Dana Randall, editor, *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1098–1116. SIAM, 2011. doi: 10.1137/1.9781611973082.83. URL https://doi.org/10.1137/1.9781611973082.83.

Gagan Goel, Chinmay Karande, Pushkar Tripathi, and Lei Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. *SIGecom Exch.*, 9(1):8, 2010. doi: 10.1145/1980534.1980542. URL https://doi.org/10.1145/1980534.1980542.

Michel X. Goemans and V. S. Ramakrishnan. Minimizing submodular functions over families of sets. *Comb.*, 15(4): 499–513, 1995. doi: 10.1007/BF01192523. URL https://doi.org/10.1007/BF01192523.

M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, June 1981. doi: 0.1007/BF02579273.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A. Smith, and Mari Ostendorf. In-context learning for few-shot dialogue state tracking. In *Findings of EMNLP*, 2022.

Adam Ibrahim, Waïss Azizian, Gauthier Gidel, and Ioannis Mitliagkas. Linear lower bounds and conditioning of differentiable games. In *International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 4583–4593. PMLR, 2020. URL http://proceedings.mlr.press/v119/ibrahim20a.html.

Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, 2001. doi: 10.1145/502090.502096. URL https://doi.org/10.1145/502090.502096.

Rishabh K. Iyer. A unified framework of robust submodular optimization. *CoRR*, abs/1906.06393, 2019. URL http://arxiv.org/abs/1906.06393.

Rishabh K. Iyer. Robust submodular minimization with applications to cooperative modeling. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang, editors, *ECAI 2020 - 24th European Conference on Artificial Intelligence*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 451–458. IOS Press, 2020. doi: 10.3233/FAIA200125. URL https://doi.org/10.3233/FAIA200125.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. In *TMLR*, 2022.

Rajiv Khanna, Ethan Elenberg, Alex Dimakis, Sahand Negahban, and Joydeep Ghosh. Scalable Greedy Feature Selection via Weak Submodularity. In Aarti Singh and Jerry Zhu, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54 of *Proceedings of Machine Learning Research*, pages 1560–1568. PMLR, Apr 2017. URL https://proceedings.mlr.press/v54/khanna17b.html.

Andreas Krause and Volkan Cevher. Submodular dictionary selection for sparse representation. In Johannes Fürnkranz and Thorsten Joachims, editors, *International Conference on Machine Learning (ICML)*, pages 567–574. Omnipress, 2010. URL https://icml.cc/Conferences/2010/papers/366.pdf.

Andreas Krause, H Brendan McMahan, Carlos Guestrin, and Anupam Gupta. Robust submodular observation selection. *Journal of Machine Learning Research*, 9(12), 2008.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, University of Toronto, 2009.

Jon Lee, Vahab S. Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In Michael Mitzenmacher, editor, *STOC*, pages 323–332. ACM, 2009. doi: 10.1145/1536414.1536459. URL https://doi.org/10.1145/1536414.1536459.

Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1049–1065, 2015. doi: 10.1109/FOCS.2015.68. URL https://doi.org/10.1109/FOCS.2015.68.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, 2022.

Shihui Li, Yi Wu, Xinyue Cui, Honghua Dong, Fei Fang, and Stuart Russell. Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. In *Conference on Artificial Intelligence (AAAI)*, pages 4213–4220. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33014213. URL `https://doi.org/10.1609/aaai.v33i01.33014213`.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL*, 2021.

Hui Lin and Jeff A. Bilmes. A class of submodular functions for document summarization. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 510–520. The Association for Computer Linguistics, 2011. URL `https://aclanthology.org/P11-1052/`.

Tianyi Lin, Chi Jin, and Michael I. Jordan. On gradient descent ascent for nonconvex-concave minimax problems. In *International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 6083–6093. PMLR, 2020. URL `http://proceedings.mlr.press/v119/lin20a.html`.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *EMNLP*, 2022.

Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Deletion-robust submodular maximization: Data summarization with "the right to be forgotten". In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 2449–2458. PMLR, 2017. URL `http://proceedings.mlr.press/v70/mirzasoleiman17a.html`.

Marko Mitrovic, Ehsan Kazemi, Morteza Zadimoghaddam, and Amin Karbasi. Data summarization at scale: A two-stage submodular approach. In *International Conference on Machine Learning*, pages 3596–3605. PMLR, 2018.

Slobodan Mitrovic, Ilija Bogunovic, Ashkan Norouzi-Fard, Jakub Tarnawski, and Volkan Cevher. Streaming robust submodular maximization: A partitioned thresholding approach. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 4557–4566, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/3baa271bc35fe054c86928f7016e8ae6-Abstract.html`.

Aryan Mokhtari, Asuman E. Ozdaglar, and Sarath Pattathil. Convergence rate of O(1/k) for optimistic gradient and extragradient methods in smooth convex-concave saddle point problems. *SIAM J. Optim.*, 30 (4):3230–3251, 2020. doi: 10.1137/19M127375X. URL `https://doi.org/10.1137/19M127375X`.

Loay Mualem and Moran Feldman. Resolving the approximability of offline and online non-monotone dr-submodular maximization over general convex sets. *arXiv preprint arXiv:2210.05965*, 2022a.

Loay Mualem and Moran Feldman. Using partial monotonicity in submodular maximization. *arXiv preprint arXiv:2202.03051*, 2022b.

George L. Nemhauser and Laurence A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Math. Oper. Res.*, 3(3):177–188, 1978. doi: 10.1287/moor.3.3.177. URL `https://doi.org/10.1287/moor.3.3.177`.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions–I. *Math. Program.*, 14(1):265–294, 1978. doi: 10.1007/BF01588971. URL `https://doi.org/10.1007/BF01588971`.

James B. Orlin, Andreas S. Schulz, and Rajan Udwani. Robust monotone submodular function maximization. *Math. Program.*, 172(1-2):505–537, 2018. doi: 10.1007/s10107-018-1320-2. URL `https://doi.org/10.1007/s10107-018-1320-2`.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

Shayan Oveis Gharan and Jan Vondrák. Submodular maximization by simulated annealing. In Dana Randall, editor, *SODA*, pages 1098–1116. SIAM, 2011. doi: 10.1137/1.9781611973082.83. URL `https://doi.org/10.1137/1.9781611973082.83`.

Benjamin Qi. On maximizing sums of non-monotone submodular and linear functions. In Sang Won Bae and Heejin Park, editors, *International Symposium on Algorithms and Computation (ISAAC)*, volume 248 of *LIPIcs*, pages 41:1–41:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi: 10.4230/LIPIcs.ISAAC.2022.41. URL `https://doi.org/10.4230/LIPIcs.ISAAC.2022.41`.

Sharon Qian and Yaron Singer. Fast parallel algorithms for statistical subset selection problems. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 5073–5082, 2019. URL `https://proceedings.neurips.cc/paper/2019/hash/ae587cfeea5ac21a8f1c1ea51027fef0-Abstract.html`.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *EMNLP*, 2016.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP*, 2019.

Mehraveh Salehi, Amin Karbasi, Dustin Scheinost, and R. Todd Constable. A submodular approach to create individualized parcellations of the human brain. In Maxime Descoteaux, Lena Maier-Hein, Alfred M. Franz, Pierre Jannin, D. Louis Collins, and Simon Duchesne, editors, *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, volume 10433 of *Lecture Notes in Computer Science*, pages 478–485. Springer, 2017. doi: 10.1007/978-3-319-66182-7_55. URL `https://doi.org/10.1007/978-3-319-66182-7_55`.

Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *J. Comb. Theory, Ser. B*, 80(2):346–355, 2000. doi: 10.1006/jctb.2000.1989. URL `https://doi.org/10.1006/jctb.2000.1989`.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *EMNLP*, 2020.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. Prompting GPT-3 to be reliable. In *ICLR*, 2023.

Adish Singla, Ilija Bogunovic, Gábor Bartók, Amin Karbasi, and Andreas Krause. Near-optimally teaching the crowd to classify. In *International Conference on Machine Learning (ICML)*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 154–162. JMLR.org, 2014. URL http://proceedings.mlr.press/v32/singla14.html.

Matthew Staib, Bryan Wilder, and Stefanie Jegelka. Distributionally robust submodular maximization. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *The 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89 of *Proceedings of Machine Learning Research*, pages 506–516. PMLR, 2019. URL http://proceedings.mlr.press/v89/staib19a.html.

Zoya Svitkina and Lisa Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. *SIAM J. Comput.*, 40(6):1715–1737, 2011. doi: 10.1137/100783352. URL https://doi.org/10.1137/100783352.

Alfredo Torrico, Mohit Singh, Sebastian Pokutta, Nika Haghtalab, Joseph (Seffi) Naor, and Nima Anari. Structured robust submodular maximization: Offline and online algorithms. *INFORMS J. Comput.*, 33(4):1590–1607, 2021. doi: 10.1287/ijoc.2020.0998. URL https://doi.org/10.1287/ijoc.2020.0998.

Sebastian Tschiatschek, Rishabh K. Iyer, Haochen Wei, and Jeff A. Bilmes. Learning mixtures of submodular functions for image collection summarization. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1413–1421, 2014. URL https://proceedings.neurips.cc/paper/2014/hash/a8e864d04c95572d1aece099af852d0a-Abstract.html.

Uber. Uber pickups in new york city, 2015. https://www.kaggle.com/datasets/fivethirtyeight/uber-pickups-in-new-york-city.

John Von Neumann and Oskar Morgenstern. *Theory of games and economic behavior, 2nd rev.* Princeton university press, 1947.

Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM J. Comput.*, 42(1):265–304, 2013. doi: 10.1137/110832318. URL https://doi.org/10.1137/110832318.

Jingkang Wang, Tianyun Zhang, Sijia Liu, Pin-Yu Chen, Jiacen Xu, Makan Fardad, and Bo Li. Adversarial attack generation empowered by min-max optimization. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, pages 16020–16033, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/85e5526a360b0bcf082d8d42e7bf100b-Abstract.html.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. In *EMNLP*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. *CoRR*, abs/2302.03668, 2023. URL http://arxiv.org/abs/2302.03668.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*, 2022. URL https://arxiv.org/abs/2211.01910.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No]
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [No]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A ADDITIONAL RELATED WORK FOR PROMPT ENGINEERING FOR NATURAL LANGUAGE PROCESSING

In-context learning (Dong et al., 2022) has emerged as a powerful technique to leverage very large language models (Brown et al., 2020; Chen et al., 2021; Ouyang et al., 2022) for Natural Language Processing (NLP) tasks to new tasks without fine-tuning. Recent works, such as Min et al. (2022); Wang et al. (2022); Wei et al. (2022), show the importance of crafting good natural language prompts for these models.

Our prompt engineering experiments build on related works which use a neural retrieval model to prompt large language models for open-domain question answering (Si et al., 2023) and dialog state tracking (Hu et al., 2022). While these previous works only use the Top-k candidates based on embedding similarity, we formulate a novel combinatorial optimization problem for each application.

Some works suggested algorithmic approaches to prompt engineering that learn parameters using gradient-based optimization (Lester et al., 2022; Li and Liang, 2021; Shin et al., 2020; Wen et al., 2023). More recently, Zhou et al. (2022) designed prompts by ranking generations from a secondary language model combined with iterative Monte Carlo search. All of these methods are complex, computationally expensive, and challenging to interpret.

B PROOFS OF SECTION 2.1

B.1 Proof of Theorem 2.1

In this section we prove Theorem 2.1, which we repeat here for convenience.

Theorem 2.1. *Assume that there exists an α-approximation algorithm ALG for the problem of maximizing a non-negative submodular function g subject to $\mathcal{F}_2$. Then, for every polynomially small $\varepsilon \in (0, \alpha]$, there exists a polynomial time algorithm that (i) outputs a set $\hat{Y} \in \mathcal{F}_2$ and the value $\min_{X \subseteq \mathcal{N}_1} f(X \cup \hat{Y})$; and (ii) guarantees that, with probability at least $2/3$, $\min_{X \subseteq \mathcal{N}_1} f(X \cup \hat{Y})$ falls within the range $[\tau/(\alpha + \varepsilon), \tau]$, where $\tau = \max_{Y \in \mathcal{F}_2} \min_{X \subseteq \mathcal{N}_1} f(X \cup Y)$. Furthermore, if f is $\mathcal{N}_2$-monotone, then it suffices for ALG to obtain α-approximation when g is guaranteed to be monotone (in addition to being non-negative and submodular).*

The majority of the section is devoted to proving the slightly different version of the last theorem given by Proposition B.1. If *ALG* is a deterministic algorithm, then the algorithm whose existence is guaranteed by Proposition B.1 is also deterministic, and immediately implies Theorem 2.1. However, if *ALG* is a randomized algorithm, then it might be necessary to use repetitions to get the result stated in Theorem 2.1. Specifically, by a Markov-like argument, the probability that $\min_{X \subseteq \mathcal{N}_1} f(X \cup \hat{Y}) \geq \tau/(\alpha + \varepsilon)$ must be at least ε/α^2, and therefore, by executing the algorithm from Proposition B.1 $O(\alpha^2/\varepsilon)$ times, the probability of getting a set $\hat{Y}$ for which $\min_{X \subseteq \mathcal{N}_1} f(X \cup \hat{Y}) \geq \tau/(\alpha + \varepsilon)$ can be made to be at least $2/3$.

Proposition B.1. *Assume that there exists an α-approximation algorithm ALG for the problem of maximizing a non-negative submodular function g subject to $\mathcal{F}_2$, then there exists a polynomial time algorithm that outputs a set $\hat{Y} \in \mathcal{F}_2$ and the value $\min_{X \subseteq \mathcal{N}_1} f(X \cup \hat{Y})$, and guarantees that (i) $\min_{X \subseteq \mathcal{N}_1} f(X \cup \hat{Y}) \leq \tau$, and (ii) the expectation of $\min_{X \subseteq \mathcal{N}_1} f(X \cup \hat{Y})$ is at least τ/α. Furthermore, if f is $\mathcal{N}_2$-monotone, then it suffices for ALG to obtain α-approximation when g is guaranteed to be monotone (in addition to being non-negative and submodular).*

To prove Proposition B.1, let us define, for every set $Y \subseteq \mathcal{N}_2$, $g(Y) = \min_{X \subseteq \mathcal{N}_1} f(X \cup Y)$. It is well-known that g is a submodular function, and we prove it in the next lemma for completeness (along with additional properties of g).

Lemma B.2. *The function $g \colon 2^{\mathcal{N}_2} \to \mathbb{R}_{\geq 0}$ is a non-negative submodular function, and there exists a polynomial time implementation of the value oracle of g. Furthermore, if f is $\mathcal{N}_2$-monotone, then g is monotone (in addition to being non-negative and submodular).*

Proof. We begin the proof by considering Algorithm 2. One can observe that this algorithm describes a way to implement a value oracle for g because, by the definitions of X' and h_Y,

$$f(X' \cup Y) = h_Y(X') = \min_{X \subseteq \mathcal{N}_1} h_Y(X) = \min_{X \subseteq \mathcal{N}_1} f(X \cup Y) = g(Y) \ .$$

Algorithm 2: Value oracle implementation (Y)

```
1 Define $h_Y(X) \triangleq f(X \uplus Y)$ for every set $X \subseteq \mathcal{N}_1$.
2 Find a set $X' \subseteq \mathcal{N}_1$ minimizing $h_Y(X')$.
3 return $f(X' \uplus Y)$.
```

Furthermore, Algorithm 2 can be implemented to run in polynomial time using any polynomial time algorithm for unconstrained submodular minimization because h_Y is a submodular function.

The non-negativity of g follows from the definition of g and the non-negativity of f. Proving that g is also submodular is more involved. Let Y_1 and Y_2 be two arbitrary subsets of $\mathcal{N}_2$, and let us choose a set $X_i \in \arg\min_{X \subseteq \mathcal{N}_1} f(X \uplus Y_i)$ for every $i \in \{1, 2\}$. Then,

$$\begin{aligned} g(Y_1) + g(Y_2) &= f(X_1 \uplus Y_1) + f(X_2 \uplus Y_2) \\ &\geq f((X_1 \cap X_2) \uplus (Y_1 \cap Y_2)) + f((X_1 \cup X_2) \uplus (Y_1 \cup Y_2)) \\ &\geq \min_{X \subseteq \mathcal{N}_1} f(X \uplus (Y_1 \cap Y_2)) + \min_{X \subseteq \mathcal{N}_1} f(X \uplus (Y_1 \cup Y_2)) = g(Y_1 \cap Y_2) + g(Y_1 \cup Y_2) \enspace , \end{aligned}$$

where the first inequality holds by the submodularity of f since $X_1 \cup X_2 \subseteq \mathcal{N}_1$ is disjoint from $Y_1 \cup Y_2 \subseteq \mathcal{N}_2$. This completes the proof that g is submodular.

It remains to prove that g is monotone whenever f is $\mathcal{N}_2$-monotone. Therefore, in the rest of the proof we assume that f indeed has this property. Then, if the sets Y_1 and Y_2 obey the inclusion $Y_1 \subseteq Y_2$, then they also obey

$$g(Y_2) = f(X_2 \cup Y_2) \geq f(X_2 \cup Y_1) \geq f(X_1 \cup Y_1) = g(Y_1) \enspace ,$$

where the first inequality follows from the $\mathcal{N}_2$-monotonicity of f, and the second inequality follows from the definition of X_1. □

We are now ready to prove Proposition B.1.

Proof of Proposition B.1. Note that Lemma B.2 implies that g has all the properties necessary for *ALG* to guarantee α-approximation for the problem of $\min_{Y \in \mathcal{F}_2} g(Y)$. Therefore, we can use *ALG* to implement in polynomial time the procedure described by Algorithm 3 (since *ALG* runs in polynomial time given a polynomial time value oracle implementation for the objective function). Since the definition of g implies $\max_{Y \in \mathcal{F}_2} g(Y) =$

Algorithm 3: Approximate using *ALG*

```
1 Use ALG to get a set $Y' \in \mathcal{F}_2$ such that $\alpha^{-1} \cdot \max_{Y \in \mathcal{F}_2} g(Y) \leq \mathbb{E}[g(Y')] \leq \max_{Y \in \mathcal{F}_2} g(Y)$.
2 return the set $Y'$ and the value $g(Y')$.
```

$\max_{Y \in \mathcal{F}_2} \min_{X \subseteq \mathcal{N}_2} f(X \uplus Y) = \tau$, the value $g(Y') = \min_{X \subseteq \mathcal{N}_1} f(X \uplus Y') \leq \max_{Y \in \mathcal{F}_2} \min_{X \subseteq \mathcal{N}_2} f(X \uplus Y)$ produced by Algorithm 3 is at most τ and in expectation at least τ/α. Therefore, Algorithm 3 has all the properties guaranteed by Proposition B.1. □

B.2 Proofs of Theorems 2.2 and 2.3

In this section we prove the inapproximability results stated in Theorems 2.2 and 2.3. The proofs of both theorems are based on the reduction described by the following proposition. Below, we use $\mathbb{N}_0$ and $\mathbb{N}$ to denote the set of natural numbers with and without 0, respectively. Additionally, recall that for a non-negative integer i, $[i] = \{1, 2, \ldots, i\}$. In particular, this implies that $[0] = \varnothing$, which is a property we employ later in the section.

Proposition B.3. *Fix any family F_2 of pairs of ground set $\mathcal{N}_2$ and constraint $\mathcal{F}_2 \subseteq 2^{\mathcal{N}_2}$. Additionally, let $\alpha \colon \mathbb{N}_0 \times F_2 \to [1, \infty)$ be an arbitrary function (intuitively, for every pair $(\mathcal{N}_2, \mathcal{F}_2) \in F_2$, $\alpha(m, \mathcal{N}_2, \mathcal{F}_2)$ is an approximation ratio that we assign to this pair when the ground set $\mathcal{N}_1$ has a size of m). Assume that there exists* *a (possibly randomized) polynomial time algorithm ALG which, given a ground set $\mathcal{N}_1$, a pair $(\mathcal{N}_2, \mathcal{F}_2) \in F_2$, and a non-negative disjointly submodular function $f\colon 2^{\mathcal{N}_1 \cup \mathcal{N}_2} \to \mathbb{R}_{\geq 0}$, outputs a value v such that, with probability at least 2/3,*

$$\tfrac{1}{\alpha(|\mathcal{N}_1|, (\mathcal{N}_2, \mathcal{F}_2))} \cdot \max_{Y \in \mathcal{F}_2} \min_{X \subseteq \mathcal{N}_1} f(X \cup Y) \leq v \leq \max_{Y \in \mathcal{F}_2} \min_{X \subseteq \mathcal{N}_1} f(X \cup Y) \enspace .$$

Then, there also exists a polynomial time algorithm that given a pair $(\mathcal{N}_2, \mathcal{F}_2) \in F_2$ and non-negative submodular functions $g_1, g_2, \ldots, g_m\colon 2^{\mathcal{N}_2} \to \mathbb{R}_{\geq 0}$ outputs a value v such that, with probability at least 2/3,

$$\tfrac{1}{\alpha(m-1, (\mathcal{N}_2, \mathcal{F}_2))} \cdot \max_{Y \in \mathcal{F}_2} \min_{1 \leq i \leq m} g_i(Y) \leq v \leq \max_{Y \in \mathcal{F}_2} \min_{1 \leq i \leq m} g_i(Y) \enspace .$$

Furthermore, if the functions $g_1, g_2, \ldots, g_m\colon 2^{\mathcal{N}_2} \to \mathbb{R}_{\geq 0}$ are all guaranteed to be monotone (in addition to being non-negative and submodular), then it suffices for ALG to have the above guarantee only when f is $\mathcal{N}_2$-monotone (in addition to being non-negative and disjointly submodular).

Before proving Proposition B.3, let us show that it indeed implies Theorems 2.2 and 2.3.

Theorem 2.2. *When f is only guaranteed to be non-negative and disjointly submodular, no polynomial time algorithm for calculating $\max_{Y \subseteq \mathcal{N}_2} \min_{X \subseteq \mathcal{N}_1} f(X \cup Y)$ has a finite approximation ratio unless $BPP = NP$.*

Proof. Fix the family $F_2 = \{([k], 2^{[k]}) \mid k \in \mathbb{N}\}$. Assume that there exists a polynomial time algorithm for calculating $\max_{Y \subseteq \mathcal{N}_2} \min_{X \subseteq \mathcal{N}_1} f(X \cup Y)$ that has a polynomial approximation ratio. By plugging this algorithm and the family F_2 into Proposition B.3, we get that there exists a polynomial time algorithm ALG and a polynomial function $\alpha\colon \mathbb{N} \times \mathbb{N} \to [1, \infty)$ such that, given integer $k \in \mathbb{N}$ and m non-negative monotone submodular functions $g_1, g_2, \ldots, g_m$, the algorithm ALG produces a value v such that, with probability at least 2/3,

$$\tfrac{1}{\alpha(m,k)} \cdot \max_{Y \subseteq [k]} \min_{1 \leq i \leq m} g_i(Y) \leq v \leq \max_{Y \subseteq [k]} \min_{1 \leq i \leq m} g_i(Y) \enspace .$$

In particular, ALG answers correctly with probability at least 2/3 whether the expression $\max_{Y \subseteq [k]} \min_{1 \leq i \leq m} g_i(Y)$ is equal to zero. Therefore, to prove the theorem it suffices to show that that exists some NP-hard problem such that every instance I of this problem can be encoded in polynomial time as an expression of the form $\max_{Y \subseteq [k]} \min_{1 \leq i \leq m} g_i(Y)$ that takes the value 0 if and only if the correct answer for the instance I is "No".

In the rest of this proof, we show that this is indeed the case for the NP-hard problem `SAT`. Every instance of `SAT` consists a CNF formula ϕ over n variables $x_1, x_2, \ldots, x_n$ that has ℓ clauses. To encode this instance, we need to construct $n + \ell$ functions over the ground set $[2n]$. Intuitively, for every integer $1 \leq i \leq n$ the elements $2i - 1$ and $2i$ of the ground set correspond to the variable x_i of ϕ. The element $2i - 1$ corresponds to an assignment of 1 to this variable, and the element $2i$ corresponds to an assignment of 0. For every integer $1 \leq i \leq n$, the objective of the function g_i is to make sure that exactly one value is assigned to x_i. Formally, this is done by defining $g_i(Y) \triangleq |\{2i - 1, 2i\} \cap Y| \bmod 2$ for every $Y \subseteq [2n]$. One can note that $g_i(Y)$ takes the value 1 only when exactly one of the elements $2i - 1$ or $2i$ belongs to Y. Furthermore, one can verify that g_i is non-negative and submodular.

Next, we need to define the functions $g_{n+1}, g_{n+2}, \ldots, g_{n+\ell}$. To define these functions, let us denote by $c_1, c_2, \ldots, c_\ell$ the clauses of ϕ. Additionally we denote by $c_j(x_i = v)$ an indicator that gets the value 1 if assigning the value v to x_i guarantees that the clause c_j is satisfied. In other words, $c_j(x_i = v)$ equals 1 only if $v = 1$ and c_j includes the positive literal x_i, or $v = 0$ and c_j includes the negative literal $\bar{x}_j$. For every integer $1 \leq j \leq \ell$, the function $g_{n+j}(Y)$ corresponds to the clause w_j and takes the value 1 only when this clause is satisfied by some element of Y. Formally,

$$g_{n+j}(Y) = \max_{i \in Y} c_j(x_{\lceil i/2 \rceil} = (i \bmod 2))$$

(notice that $x_{\lceil i/2 \rceil}$ is the index of the variable corresponding to element i, and $i \bmod 2$ is the value assigned to this variable by the element i). One can verify that $g_{n+j}(Y)$ is a non-negative submodular (and even monotone) function.

Let us now explain why $\max_{Y \subseteq [2n]} \min_{1 \leq i \leq m} g_i(Y)$ takes the value 0 if and only if ϕ is not satisfiable. First, if there exists a satisfying assignment a for ϕ, then one can construct a set $Y \subseteq [2n]$ that encodes a. Specifically, for every integer $1 \leq i \leq n$, Y should include $2i - 1$ (and not $2i$) if a assigns the value 1 to x_i, and otherwise Y should include $2i$ (and not $2i - 1$). Such a choice of Y will make all the above functions $g_1, g_2, \ldots, g_{n+\ell}$ take the value 1, and therefore, $\max_{Y \subseteq [2n]} \min_{1 \leq i \leq m} g_i(Y) = 1$ in this case. Consider now the case in which ϕ does not have a satisfying assignment. Then, for every set $Y \subseteq [2n]$ we must have one of the following. The first option is that Y includes either both $2i-1$ and $2i$, or neither of these elements, for some integer i, which makes g_i evaluate to 0 on Y. The other option is that Y corresponds to some legal assignment a of values to $x_1, x_2, \ldots, x_n$ that violates some clause c_j, and thus, g_{n+j} evaluates to 0 on Y. In both cases $\min_{1 \leq i \leq m} g_i(Y) = 0$. □

Theorem 2.3. *When f is only guaranteed to be non-negative, $\mathcal{N}_2$-monotone and disjointly submodular, no polynomial time algorithm for calculating $\max_{Y \subseteq \mathcal{N}_2, |Y| \leq \rho} \min_{X \subseteq \mathcal{N}_1} f(X \uplus Y)$, where ρ is a parameter of the problem, has a finite approximation ratio unless $BPP = NP$.*

Proof. The proof of this theorem is very similar to the proof of Theorem 2.2, and therefore, we only describe here the differences between the two proofs. First, the family F_2 should be chosen this time as $\mathcal{F}_2 = \{([2k], \{Y \subseteq [2k] \mid |Y| \leq k\} \mid k \in \mathbb{N}\}$. This modification implies that we now need to encode ϕ as an instance of

$$\max_{\substack{Y \subseteq [2n] \\ |Y| \leq n}} \min_{1 \leq i \leq m} g_i(Y) \enspace ,$$

where the functions $g_i(Y)$ are all non-negative monotone submodular functions over the ground set $[2n]$. We do this using $n + \ell$ functions like in the proof of Theorem 2.2. Moreover, the functions $g_{n+1}, g_{n+2}, g_{n+\ell}$ are defined exactly like in the proof of Theorem 2.2.

For every integer $1 \leq i \leq n$, the function g_i still corresponds to the variable x_i, but now the role of g_i is only to guarantee that x_i gets at least a single value. This is done by setting $g_i(Y) = \min\{|Y \cap \{2i-1, 2i\}|, 1\}$, which means that g_i takes the value 1 only when at least one of the elements $2i-1$ or $2i$ belongs to Y. Note that g_i is indeed non-negative, monotone and submodular, as necessary. The main observation that we need to make is that if Y is a set of size at most n for which all the functions $g_1, g_2, \ldots, g_n$ return 1, then Y must include at least one element of the pair $\{2i-1, 2i\}$ for every integer $1 \leq i \leq n$. Since these are n disjoint pairs, and Y contains at most n elements, we get that Y contains *exactly* one element of each one of the pairs $\{2i-1, 2i\}$. In other words, $\min_{1 \leq i \leq n} g_i(Y) = 1$ if and only if Y corresponds to assigning exactly one value to every variable x_i, which is exactly the property that the functions $g_1, g_2, \ldots, g_n$ need to have to allow the rest of the proof of Theorem 2.2 to go through. □

Remark. The above proof of Theorem 2.3 plugs into Proposition B.3 the observation that an expression of the form $\max_{Y \subseteq \mathcal{N}_2, |Y| \leq k} \min_{1 \leq i \leq m} g_i(Y)$ can capture an NP-hard problem. The last observation was already shown by Theorem 3 of Krause et al. (2008) (for the Hitting-Set problem). Thus, Theorem 2.3 can also be obtained as a corollary of Proposition B.3 and Theorem 3 of Krause et al. (2008). However, for completeness and consistency, we chose to provide a different proof of Theorem 2.3 that closely follows the proof of Theorem 2.2.

We now get to the proof of Proposition B.3. One can observe that to prove this proposition it suffices to show the following lemma (the algorithm whose existence is guaranteed by Proposition B.3 can be obtained by simply applying ALG to the ground set $\mathcal{N}_1$ and function f defined by Lemma B.4).

Lemma B.4. *Given non-negative submodular functions $g_1, g_2, \ldots, g_m \colon 2^{\mathcal{N}_2} \to \mathbb{R}_{\geq 0}$, there exists a ground set $\mathcal{N}_1$ and a non-negative disjointly submodular function $f \colon 2^{\mathcal{N}_1 \uplus \mathcal{N}_2} \to \mathbb{R}_{\geq 0}$ such that*

- *the size of the ground set $\mathcal{N}_1$ is $m-1$.*
- *given sets $X \subseteq \mathcal{N}_1$ and $Y \subseteq \mathcal{N}_2$, it is possible to evaluate $f(X \uplus Y)$ in polynomial time.*
- *for every set $Y \subseteq \mathcal{N}_2$, $\min_{X \subseteq \mathcal{N}_1} f(X \uplus Y) = \min_{1 \leq i \leq m} g_i(Y)$.*
- *when the functions $g_1, g_2, \ldots, g_m$ are all monotone (in addition to being non-negative and submodular), then the function f is guaranteed to be $\mathcal{N}_2$-monotone (in addition to being non-negative and disjointly submodular).*

The rest of this section is devoted to proving Lemma B.4. Let us start by describing how the ground set $\mathcal{N}_1$ and the function f are constructed. We assume without loss of generality that $\mathcal{N}_2 \cap [m-1] = \varnothing$, which allows us to choose $\mathcal{N}_1 = [m-1]$. Given a set $X \subseteq [m-1]$, let us define $c(X) \triangleq \max\{i \in \mathbb{N}_0 \mid [i] \subseteq X\}$ (in other words, $c(X)$ is the largest integer such that all the numbers 1 to i appear in X). Additionally, we choose M to be a number obeying $g_i(Y) \leq M/2$ for every $i \in [m]$ and $Y \subseteq \mathcal{N}_2$ (such a number can be obtained in polynomial time by running the 2-approximation algorithm of Buchbinder and Feldman (2018) for unconstrained submodular maximization on the functions $g_1, g_2, \dots, g_m$, and then setting M to be four times the largest number returned). Using this notation, we can now define, for every two sets $X \subseteq \mathcal{N}_1$ and $Y \subseteq \mathcal{N}_2$,

$$f(X \mathbin{\dot\cup} Y) \triangleq g_{c(X)+1}(Y) + (|X| - c(X)) \cdot M \enspace .$$

The following observation states some properties of f that immediately follow from the definition of f and the fact that $c(X)$ is at most $|X|$ by definition.

Observation B.5. *The function f is non-negative and can be evaluated in polynomial time. Furthermore, f is $\mathcal{N}_2$-monotone when the functions $g_1, g_2, \dots, g_m$ are monotone because $g_{c(X)+1}(Y) + (|X| - c(X)) \cdot M$ is a monotone function of Y for any fixed set $X \subseteq \mathcal{N}_1$.*

The following two lemmata prove additional properties of f.

Lemma B.6. *The function f is disjointly submodular, i.e., it is submodular when restricted to either $\mathcal{N}_1$ or $\mathcal{N}_2$.*

Proof. For every fixed set $X \subseteq \mathcal{N}_1$, there exists a value $i \in [m]$ and another value c, both depending only on X, such that $f(X \mathbin{\dot\cup} Y) = g_i(Y) + c$. Since adding a constant to a submodular function does not affect its submodularity, this implies that f is submodular when restricted to $\mathcal{N}_2$. In the rest of the proof we concentrate on showing that f is also submodular when restricted to $\mathcal{N}_1$.

Consider now an arbitrary element $i \in \mathcal{N}_1$. For every two sets $X \subseteq \mathcal{N}_1 - i$ and $Y \subseteq \mathcal{N}_2$,

$$f(i \mid X \mathbin{\dot\cup} Y) = g_{c(X+i)+1}(Y) - g_{c(X)+1}(Y) + (1 + c(X) - c(X+i)) \cdot M \enspace .$$

To show that f is submodular when restricted to $\mathcal{N}_1$, we need to show that the last expression is a down-monotone function X, i.e., that its value does not increase when elements are added to X. To do that, it suffices to show that the addition to X of any single element $j \in \mathcal{N}_1 \setminus (X + i)$ does not increase the value of this expression; which we show below by considering a few cases.

The first case we need to consider is the case of $[i-1] \not\subseteq X + j$. Clearly, in this case $c(X) = c(X+i)$ and $c(X+j+i) = c(X+j)$, and therefore,

$$\begin{aligned} f(i \mid (X+j) \mathbin{\dot\cup} Y) &= g_{c(X+j+i)+1}(Y) - g_{c(X+j)+1}(Y) + (1 + c(X+j) - c(X+j+i)) \cdot M \\ &= M = g_{c(X+i)+1}(Y) - g_{c(X)+1}(Y) + (1 + c(X) - c(X+i)) \cdot M = f(i \mid X \mathbin{\dot\cup} Y) \enspace . \end{aligned}$$

The second case we consider the case in which $[i-1] \subseteq X + j$, but $[i-1] \not\subseteq X$. In this case

$$\begin{aligned} f(i \mid (X+j) \mathbin{\dot\cup} Y) &= g_{c(X+j+i)+1}(Y) - g_{c(X+j)+1}(Y) + (1 + c(X+j) - c(X+j+i)) \cdot M \\ &\leq g_{c(X+j+i)+1}(Y) - g_{c(X+j)+1}(Y) \leq g_{c(X+i)+1}(Y) - g_{c(X)+1}(Y) + M \\ &= g_{c(X+i)+1}(Y) - g_{c(X)+1}(Y) + (1 + c(X) - c(X+i)) \cdot M = f(i \mid X \mathbin{\dot\cup} Y) \enspace , \end{aligned}$$

where the first inequality holds since the definition of the case implies $c(X+j+i) \geq i = 1 + c(X+j)$, the second inequality follows from the definition of M, and the penultimate equality holds since the definition of the case implies $c(X) = c(X+i)$.

The third case we need to consider is when $[i-1] \subseteq X$ and $c(X+i) = c(X+i+j)$. Since we also have in this case $c(X) = i - 1 = c(X+j)$, we get

$$\begin{aligned} f(i \mid (X+j) \mathbin{\dot\cup} Y) &= g_{c(X+j+i)+1}(Y) - g_{c(X+j)+1}(Y) + (1 + c(X+j) - c(X+j+i)) \cdot M \\ &= g_{c(X+i)+1}(Y) - g_{c(X)+1}(Y) + (1 + c(X) - c(X+i)) \cdot M = f(i \mid X \mathbin{\dot\cup} Y) \end{aligned}$$

The last case we need to consider is when $[i-1] \subseteq X$ and $c(X+i) < c(X+j+i)$. In this case

$$\begin{aligned} f(i \mid (X+j) \mathbin{\dot\cup} Y) &= g_{c(X+j+i)+1}(Y) - g_{c(X+j)+1}(Y) + (1 + c(X+j) - c(X+j+i)) \cdot M \\ &\leq g_{c(X+j+i)+1}(Y) - g_{c(X+j)+1}(Y) - M \leq g_{c(X+i)+1}(Y) - g_{c(X)+1}(Y) \\ &= g_{c(X+i)+1}(Y) - g_{c(X)+1}(Y) + (1 + c(X) - c(X+i)) \cdot M = f(i \mid X \mathbin{\dot\cup} Y) \enspace , \end{aligned}$$

where the first inequality holds since the definition of the case implies $c(X+j) = i-1 = c(X+i)-1 < c(X+j+i)-1$, the second inequality follows from the definition of M, and the penultimate equality holds since the definition of the case implies $c(X) = i - 1 = c(X+i) - 1$. □

Lemma B.7. *For every set $Y \subseteq \mathcal{N}_2$, $\min_{X \subseteq \mathcal{N}_1} f(X \cupdot Y) = \min_{1 \leq i \leq m} g_i(Y)$.*

Proof. Observe that for every integer $1 \leq i \leq m$, we have $f([i-1] \cupdot Y) = g_i(Y)$ because $|[i-1]| = c([i-1]) = i-1$. Therefore,

$$\min_{1 \leq i \leq m} f([i-1] \cupdot Y) = \min_{1 \leq i \leq m} g_i(Y) \enspace . \tag{4}$$

Consider now an arbitrary subset X of $\mathcal{N}_1$ that is not equal to $[i-1]$ for any integer $1 \leq i \leq m$. For such a subset we must have $c(X) \leq |X| - 1$, and therefore,

$$f(X \cupdot Y) = g_{c(X)+1}(Y) + (|X| - c(X)) \cdot M \geq g_{c(X)+1}(Y) + M \geq M \geq \min_{1 \leq i \leq m} g_i(Y) \enspace ,$$

where the second inequality follows from the non-negativity of $g_{c(X)+1}$, and the last inequality holds by the definition of M. Combining this inequality with Equation (4) completes the proof of the lemma. □

Lemma B.4 now follows by combining Observation B.5, Lemma B.6 and Lemma B.7.

C OMITTED PROOFS OF SECTION 2.2

C.1 Proof of Theorem 2.4

In this section we prove Theorem 2.4, which we repeat here for convenience.

Theorem 2.4. *Assuming $\{u\} \in \mathcal{F}_2$ for every $u \in \mathcal{N}_2$, there is a polynomial time algorithm that, given a non-negative disjointly submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$, returns a set $\hat{X} \subseteq \mathcal{N}_1$ and a value v such that both $\max_{Y \in \mathcal{F}_2} f(\hat{X} \cupdot Y)$ and v fall within the range $[\tau, (|\mathcal{N}_2| + 1) \cdot \tau]$, where $\tau \triangleq \min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \cupdot Y)$.*

The algorithm that we use to prove Theorem 2.4 is given as Algorithm 4. We note that the function g defined by this algorithm is the average of $|\mathcal{N}_2| + 1$ submodular functions (since f is submodular once the subset of $\mathcal{N}_2$ in the argument set is fixed), and therefore, g is also submodular. As written, Algorithm 4 is good only for the case in which $\varnothing \in \mathcal{F}_2$, and for simplicity, we assume throughout the section that this is indeed the case. If $\varnothing \notin \mathcal{F}_2$, then the term $f(X)$ should be dropped from the definition of g in Algorithm 4, which allows the proof to go through.

Algorithm 4: Estimating the min-max using singletons

```
1 Define a function $g\colon 2^{\mathcal{N}_1} \to \mathbb{R}_{\geq 0}$ as follows. For every set $X \subseteq \mathcal{N}_1$, $g(X) \triangleq f(X) + \sum_{u \in \mathcal{N}_2} f(X \cupdot \{u\})$.
2 Use an unconstrained submodular minimization algorithm to find $X' \subseteq \mathcal{N}_1$ minimizing $g(X')$.
3 return the set $X'$ and the value $g(X')$.
```

The analysis of Algorithm 4 is based on the observation that $g(X)$ provides an approximation for $\max_{Y \in \mathcal{F}_2} f(X \cupdot Y)$.

Lemma C.1. *For every set $X \subseteq \mathcal{N}_1$, $\max_{Y \in \mathcal{F}_2} f(X \cupdot Y) \leq g(X) \leq (|\mathcal{N}_2| + 1) \cdot \max_{Y \in \mathcal{F}_2} f(X \cupdot Y)$.*

Proof. Let Y' be the set in $\mathcal{F}_2$ maximizing $f(X \cupdot Y')$, then the disjoint submodularity of f guarantees that

$$\begin{aligned}\max_{Y \in \mathcal{F}_2} f(X \cupdot Y) = f(X \cup Y') \leq{}& f(X) + \sum_{u \in Y'} f(u \mid X) \\ &\leq f(X) + \sum_{u \in Y'} f(X \cupdot \{u\}) \leq f(X) + \sum_{u \in \mathcal{N}_2} f(X \cupdot \{u\}) = g(X) \enspace ,\end{aligned}$$

where the second and last inequalities hold by the non-negativity of f. This completes the proof of the first inequality of the lemma. To see why the other inequality holds as well, we note that $g(X)$ is the sum of $|\mathcal{N}_2| + 1$ terms, each of which is individually upper bounded by $\max_{Y \in \mathcal{F}_2} f(X \cupdot Y)$. □

Using the last lemma, we can now prove Theorem 2.4.

Proof of Theorem 2.4. Let X^* be the set minimizing $\max_{Y \in \mathcal{F}_2} f(X^* \uplus Y)$. Then, by Lemma C.1 and the choice of X' by Algorithm 4,

$$\begin{aligned}\tau = {} & \min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y) \leq \max_{Y \in \mathcal{F}_2} f(X' \uplus Y) \leq g(X') \leq g(X^*) \\ & \leq (|\mathcal{N}_2| + 1) \cdot \max_{Y \in \mathcal{F}_2} f(X^* \uplus Y) = (|\mathcal{N}_2| + 1) \cdot \min_{X \subseteq \mathcal{N}_1} \max_{Y \in \mathcal{F}_2} f(X \uplus Y) = (|\mathcal{N}_2| + 1)\tau \enspace . \end{aligned}$$ □

C.2 Proof of Theorem 2.5

In this section we would like to prove Theorem 2.5. However, the majority of the section is devoted to proving the following slightly different theorem, which implies Theorem 2.5.

Theorem C.2. *For every constant $\varepsilon \in (0, 1/2)$, there exists a polynomial time algorithm that given a non-negative disjointly submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ returns a set $\hat{X}$ and a value v such that*

- *v's expectation falls within the range $[\tau, (4 + \varepsilon/2)\tau]$, where $\tau \triangleq \min_{X \subseteq \mathcal{N}_1} \max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y)$, and*
- *with probability at least $1 - \varepsilon/[8(|\mathcal{N}_2| + 1)]$, both v and $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$ fall within the range $[\tau, (4 + \varepsilon/2)\tau]$.*

Before getting to the proof of Theorem C.2, let us show that it indeed implies Theorem 2.5, which we repeat here for convenience.

Theorem 2.5. *For every constant $\varepsilon \in (0, 1)$, there exists a polynomial time algorithm that given a non-negative disjointly submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ returns a set $\hat{X} \subseteq \mathcal{N}_1$ and a value v such that the expectations of both $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$ and v fall within the range $[\tau, (4 + \varepsilon)\tau]$, where $\tau \triangleq \min_{X \subseteq \mathcal{N}_1} \max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y)$. Furthermore, the probability that both $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$ and v fall within this range is at least $1 - O(|\mathcal{N}_2|^{-1})$.*

Proof. Since the guarantee of Theorems 2.5 becomes stronger as ε becomes smaller, it suffices to prove the theorem for $\varepsilon \in (0, 1/2)$. Furthermore, the only way in which the algorithm guaranteed by Theorem C.2 might not obey the properties described in Theorem 2.5 is if the expectation of $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$ for its output set $\hat{X}$ does not fall within the range $[\tau, (4 + \varepsilon)\tau]$. Thus, to prove Theorem 2.5 it is only necessary to show how to modify the output set $\hat{X}$ of Theorem C.2 in a way that does not violate the other properties guaranteed by this theorem, but makes the expectation of $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$ fall into the right range. We do that using Algorithm 5. This algorithm uses a deterministic polynomial time algorithm that obtains 2-approximation for unconstrained submodular maximization. Such an algorithm was given by Buchbinder and Feldman (2018).

Algorithm 5: Best of two (ε)

```
1 Execute the algorithm guaranteed by Theorem C.2. Let $X'$ denote its output set.
2 Use an algorithm for unconstrained submodular maximization to find a set $Y' \subseteq \mathcal{N}_2$ such that
    $\max_{Y \subseteq \mathcal{N}_2} f(X' \uplus Y) \leq 2f(X' \uplus Y') \leq 2 \cdot \max_{Y \subseteq \mathcal{N}_2} f(X' \uplus Y)$.
3 Execute the algorithm guaranteed by Theorem 2.4. Let $X''$ denote its output set.
4 Use an algorithm for unconstrained submodular maximization to find a set $Y'' \subseteq \mathcal{N}_2$ such that
    $\max_{Y \subseteq \mathcal{N}_2} f(X'' \uplus Y) \leq 2f(X'' \uplus Y'') \leq 2 \cdot \max_{Y \subseteq \mathcal{N}_2} f(X'' \uplus Y)$.
5 if $f(X' \uplus Y') \leq 2f(X'' \uplus Y'')$ then return $X'$.
6 else return $X''$.
```

Let us denote the output set of Algorithm 5 by $\hat{X}$, and observe that the choice of the output set in the last five lines of Algorithm 5 guarantees that whenever $\hat{X} = X''$, we also have

$$\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y) = \max_{Y \subseteq \mathcal{N}_2} f(X'' \uplus Y) \leq 2f(X'' \uplus Y'') \leq f(X' \uplus Y') \leq \max_{Y \subseteq \mathcal{N}_2} f(X' \uplus Y) \enspace .$$

Since the inequality $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y) \leq \max_{Y \subseteq \mathcal{N}_2} f(X' \uplus Y)$ trivially applies also when $\hat{X} = X'$, we get that this inequality always hold, and therefore, with probability at least $1 - \varepsilon/[8(|\mathcal{N}_2| + 1)]$ we must have

$$\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y) \leq (4 + \varepsilon/2)\tau$$

because Theorem C.2 guarantees that this inequality holds with at least this probability when $\hat{X}$ is replaced with X'. Furthermore, since we always have $\tau = \min_{X \subseteq \mathcal{N}_1} \max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y) \leq \max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$, the inequality $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y) \leq \max_{Y \subseteq \mathcal{N}_2} f(X' \uplus Y)$ also shows that $\hat{X}$ falls within the range $[\tau, (4+\varepsilon)\tau]$ whenever X' falls within the this range.

Next, we need to prove a second upper bound on $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$. By the choice of the output set in the last five lines of Algorithm 5, when this output set is X', we have

$$\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y) = \max_{Y \subseteq \mathcal{N}_2} f(X' \uplus Y) \leq 2f(X' \uplus Y') \leq 4f(X'' \uplus Y'') \leq 4 \cdot \max_{Y \subseteq \mathcal{N}_2} f(X'' \uplus Y) \ .$$

Since the non-negativity of f implies that the inequality $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y) \leq 4 \cdot \max_{Y \subseteq \mathcal{N}_2} f(X'' \uplus Y)$ applies also when $\hat{X} = X''$, we get by Theorem 2.4,

$$\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y) \leq 4 \cdot \max_{Y \subseteq \mathcal{N}_2} f(X'' \uplus Y) \leq 4(|\mathcal{N}_2| + 1)\tau \ .$$

We are now ready to prove that the expectation of the expression $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$ falls within the range $[\tau, (4+\varepsilon)\tau]$ as is guaranteed by Theorem 2.5. The expectation is at least the lower end of this range because, as mentioned above, it always holds that $\tau = \min_{X \subseteq \mathcal{N}_1} \max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y) \leq \max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$. Additionally, by the law of total expectation and the two above proved upper bounds on $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)$,

$$\begin{aligned}\mathbb{E}\left[\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \uplus Y)\right] &\leq \left(1 - \frac{\varepsilon}{8(|\mathcal{N}_2|+1)}\right) \cdot \left(4 + \frac{\varepsilon}{2}\right)\tau + \frac{\varepsilon}{8(|\mathcal{N}_2|+1)} \cdot 4(|\mathcal{N}_2|+1)\tau \\ &= \left[\left(4 + \frac{\varepsilon}{2}\right) + \frac{\varepsilon}{2}\right]\tau = (4+\varepsilon)\tau \ . \end{aligned}$$ □

It remains to prove Theorem C.2. The algorithm that we use for this purpose is given as Algorithm 6. We note that the function g defined by this algorithm is the average of m submodular functions (since f is submodular once the subset of $\mathcal{N}_2$ in the argument set is fixed), and therefore, g is also submodular.

Algorithm 6: Estimating the min-max via random subsets (ε)

```
1 Let $n_1 = |\mathcal{N}_1|$ and $n_2 = |\mathcal{N}_2|$, and pick $m = \lceil 3200\varepsilon^{-2}[(n_1+1)\ln 2 + \ln(n_2+1) + \ln(8/\varepsilon)]\rceil$ uniformly random
    (and independent) subsets $Y_1, Y_2, \ldots, Y_m$ of $\mathcal{N}_2$.
2 Define a function $g\colon 2^{\mathcal{N}_1} \to \mathbb{R}_{\geq 0}$ as follows. For every $X \subseteq \mathcal{N}_1$, $g(X) \triangleq \frac{1}{m}\sum_{i=1}^{m} f(X \uplus Y_i)$.
3 Use an unconstrained submodular minimization algorithm to find a set $X' \subseteq \mathcal{N}_1$ minimizing $g(X')$.
4 **return** the set $X'$ and the value $(4 + \varepsilon/2) \cdot g(X')$.
```

The analysis of Algorithm 6 uses the following known lemma.

Lemma C.3 (Lemma 2.2 of Feige et al. (2011))**.** *Given a submodular function $f\colon 2^{\mathcal{N}} \to \mathbb{R}_{\geq 0}$ and two sets $A, B \subseteq \mathcal{N}$, if $A(p)$ and $B(q)$ are independent random subsets of A and B, respectively, such that $A(p)$ includes every element of A with probability p (not necessarily independently), and $B(q)$ includes every element of B with probability q (again, not necessarily independently), then*

$$\mathbb{E}[f(A(p) \cup B(q))] \geq (1-p)(1-q) \cdot f(\varnothing) + p(1-q) \cdot f(A) + (1-p)q \cdot f(B) + pq \cdot f(A \cup B) \ .$$

Given a vector $\mathbf{x} \in [0,1]^{\mathcal{N}_2}$, we define $\mathsf{R}(\mathbf{x})$ to be a random subset of $\mathcal{N}_2$ that includes every element $u \in \mathcal{N}_2$ with probability x_u, independently. Given a set $S \subseteq \mathcal{N}_2$, it will also be useful to denote by $\mathbf{1}_S$ the characteristic vector of S, i.e., the vector in $[0,1]^{\mathcal{N}_2}$ that has 1 in the coordinates corresponding to the elements of S, and 0 in the other coordinates. Using this notation, we can now define a function $h\colon 2^{\mathcal{N}_1} \to \mathbb{R}_{\geq 0}$ as follows. For every set $X \subseteq \mathcal{N}_1$, $h(X) \triangleq \mathbb{E}[f(X \uplus \mathsf{R}(1/2 \cdot \mathbf{1}_{\mathcal{N}_2}))]$. The following lemma shows that $h(X)$ is related to $\max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y)$.

Lemma C.4. *For every set $X \subseteq \mathcal{N}_1$, $\max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y) \leq 4h(X)$.*

Proof. Let us denote by $Y'(X)$ an arbitrary set in $\arg\max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y)$, and define $r_X(Y) \triangleq f(X \cup Y)$. Then,

$$\begin{aligned} h(X) &= \mathbb{E}[f(X \uplus \mathsf{R}(1/2 \cdot \mathbf{1}_{\mathcal{N}_2})] = \mathbb{E}[r_X(\mathsf{R}(1/2 \cdot \mathbf{1}_{Y'(X)}) \cup \mathsf{R}(1/2 \cdot \mathbf{1}_{\mathcal{N}_2 \setminus Y'(X)}))] \\ &\geq \tfrac{1}{4} r_X(Y'(X)) = \tfrac{1}{4} f(X \uplus Y'(X)) = \tfrac{1}{4} \cdot \max_{Y \subseteq \mathcal{N}_2} f(X \uplus Y) \ , \end{aligned}$$

where the inequality follows from Lemma C.3 and the observation that for any fixed set $X \subseteq \mathcal{N}_1$ the function r_X is a non-negative submodular function. $\square$

The last lemma shows that the function h is useful. The following lemma complements the picture by showing that the function g defined by Algorithm 6 is a good approximation of h.

Lemma C.5. *With probability at least* $1 - \varepsilon/[8(n_2 + 1)]$*, for every set* $X \subseteq \mathcal{N}_1$ *(at the same time) we have* $|g(X) - h(X)| \leq (\varepsilon/20) \cdot h(X)$.

Proof. Fix some set $X \subseteq \mathcal{N}_1$, and let us define $Z_i \triangleq f(X \cup Y_i)$ for every integer $1 \leq i \leq m$. We would like to study the properties of the random variables $Z_1, Z_2, \ldots, Z_m$. First, note that these random variables are independent since the sets $Y_1, Y_2, \ldots, Y_m$ are chosen independently by Algorithm 6. Second, by the definition of h and the distribution of Y_i, $\mathbb{E}[Z_i] = \mathbb{E}[f(X \cup Y_i)] = h(X)$. We would also like to bound the range of values that the random variables $Z_1, Z_2, \ldots, Z_m$ can take. On the one hand, these random variables are non-negative since f is non-negative. On the other hand, $Z_i = f(X \cup Y_i) \leq \max_{Y \subseteq \mathcal{N}_2} f(X \cup Y) \leq 4h(X)$, where the second inequality holds by Lemma C.4.

Given the above proved properties of the random variables $Z_1, Z_2, \ldots, Z_m$, Hoeffding's inequality shows that

$$\Pr[|g(X) - h(X)| \leq (\varepsilon/20) \cdot h(X)] = \Pr\left[\left|\tfrac{1}{m}\sum_{i=1}^{m} Z_i - \mathbb{E}\left[\tfrac{1}{m}\sum_{i=1}^{m} Z_i\right]\right| > (\varepsilon/20) \cdot h(X)\right]$$
$$\leq 2e^{-\frac{2m^2[(\varepsilon/20)\cdot h(X)]^2}{\sum_{i=1}^{m}[4h(X)]^2}} = 2e^{-\frac{m\varepsilon^2}{3200}} \leq 2e^{-(n_1+1)\ln 2 - \ln(n_2+1) - \ln(8/\varepsilon)} = 2^{-n_1} \cdot \frac{\varepsilon}{8(n_2+1)} \enspace ,$$

where the last inequality follows from the definition of m. The lemma now follows from the last inequality by the union bound since X was chosen as an arbitrary subset of $\mathcal{N}_1$, and there are only 2^{n_1} such subsets. $\square$

Using the above lemmata, we can now prove Theorem C.2.

Proof of Theorem C.2. Let us denote by X^* a set minimizing $\max_{Y \subseteq \mathcal{N}_2} f(X^* \cup Y)$. By the definition of g and the choice of X' by Algorithm 6,

$$g(X') \leq g(X^*) = \tfrac{1}{m}\sum_{i=1}^{m} f(X^* \cup Y_i) \leq \max_{Y \subseteq \mathcal{N}_2} f(X^* \cup Y) = \min_{X \subseteq \mathcal{N}_1} \max_{Y \subseteq \mathcal{N}_2} f(X^* \cup Y) = \tau \enspace .$$

Let us now denote by $\mathcal{E}$ the event that $|g(X) - h(X)| \leq (\varepsilon/20) \cdot h(X)$ for every set $X \subseteq \mathcal{N}$. By Lemma C.5, $\mathcal{E}$ happens with probability at least $1 - \varepsilon/[8(n_2 + 1)]$. Furthermore, conditioned on $\mathcal{E}$, we have

$$\max_{Y \subseteq \mathcal{N}_2} f(X' \cup Y) \leq 4h(X') \leq \frac{4g(X')}{1 - \varepsilon/20} \leq \frac{4\tau}{1 - \varepsilon/20} \leq (4 + \varepsilon/2)\tau \enspace , \tag{5}$$

where the first inequality hold by Lemma C.4, and the last inequality holds for $\varepsilon \in (0, 1/2)$. Since we always also have $\max_{Y \subseteq \mathcal{N}_2} f(X' \cup Y) \geq \min_{X \subseteq \mathcal{N}_1} \max_{Y \subseteq \mathcal{N}_2} f(X \cup Y) = \tau$, the above inequality already proves that $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \cup Y)$ falls within the range $[\tau, (4 + \varepsilon/2)\tau]$ whenever the event $\mathcal{E}$ happens.

We now would like to show that the output value $(4 + \varepsilon/2) \cdot g(X')$ of Algorithm 6 also falls within this range when the event $\mathcal{E}$ happens. Since we already proved that $g(X') \leq \tau$, all we need to show is that $(4 + \varepsilon/2) \cdot g(X')$ is at least τ condition on $\mathcal{E}$. This is indeed the case since Inequality (5) implies

$$(4 + \varepsilon/2) \cdot g(X') \geq \frac{4}{1 - \varepsilon/20} \cdot g(X') \geq \max_{Y \subseteq \mathcal{N}_2} f(X' \cup Y) \geq \tau \enspace ,$$

where the first inequality holds for $\varepsilon \in (0, 1/2)$. In conclusion, we have shown that when the event $\mathcal{E}$ happens the value $(4 + \varepsilon/2) \cdot g(X')$ returned by Algorithm 6 and the expression $\max_{Y \subseteq \mathcal{N}_2} f(\hat{X} \cup Y)$ both fall within the range $(4 + \varepsilon/2)$. Since the probability of the event $\mathcal{E}$ is at least $1 - \varepsilon/[8(n_2 + 1)]$, to complete the proof of the theorem it only remains to show that the expectation of $(4 + \varepsilon/2) \cdot g(X')$ falls within the range $[\tau, (4 + \varepsilon/2)\tau]$, which is what we do in the rest of this proof.

The inequality $\mathbb{E}[(4+\varepsilon/2)\cdot g(X')] \leq (4+\varepsilon/2)\tau$ follows immediately from the above proof that we deterministically have $g(X') \leq \tau$. Using Inequality (5), we can also get that, conditioned on $\mathcal{E}$,

$$\begin{aligned} g(X') &\geq \frac{(1-\varepsilon/20)\cdot \max_{Y\subseteq\mathcal{N}_2} f(X' \cup Y)}{4} \\ &\geq \frac{(1-\varepsilon/20)\cdot \min_{X\subseteq\mathcal{N}_1}\max_{Y\subseteq\mathcal{N}_2} f(X \cup Y)}{4} = \frac{(1-\varepsilon/20)\tau}{4} \enspace . \end{aligned}$$

Thus, we can use the law of total expectation to get

$$\mathbb{E}[g(X')] \geq \Pr[\mathcal{E}]\cdot\mathbb{E}[g(X') \mid \mathcal{E}] \geq \left(1 - \frac{\varepsilon}{8(n_2+1)}\right)\cdot\frac{(1-\varepsilon/20)\tau}{4} \geq \left(\frac{1-9\varepsilon/80}{4}\right)\tau \enspace ,$$

which implies

$$\mathbb{E}[(4+\varepsilon/2)\cdot g(X')] \geq \frac{(4+\varepsilon/2)\cdot(1-9\varepsilon/80)}{4}\cdot\tau \geq \tau \enspace ,$$

where the second inequality holds for $\varepsilon \in [0, 1/2]$. □

D ADDITIONAL APPLICATIONS

D.1 Adversarial Attack on Image Summarization

In this section we consider the application of "Adversarial Attack on Image Summarization", which is an attack version of an application studied by many previous works (see, e.g., Mitrovic et al. (2018); Mualem and Feldman (2022b); Tschiatschek et al. (2014)). The setting for this application includes a collection of images from ℓ disjoint categories (such as birds, airplanes or cats), and a user that specifies $r \in [\ell]$ categories of interest. In the classical version of this application, the objective is to construct a subset of k images summarizing the images belonging to the categories specified by the user. However, here we are interested in mounting an attack against this summarization task. Specifically, our goal is to add a few additional images to the original set of images in a way that undermines the quality of any subsequently chosen summarizing subset.

Formally, we have in this application a (completed) similarity matrix M comprising similarity scores for both the set $\mathcal{N}_2$ of original images and the set $\mathcal{N}_1$ of images that the attacker may add. We aim to choose a set $X \subseteq \mathcal{N}_1$ of images such that adding the images of $\mathcal{N}_1 \setminus X$ simultaneously minimizes the value every possible summarizing subset Y. The value of a summarizing set Y is given by the following objective function.

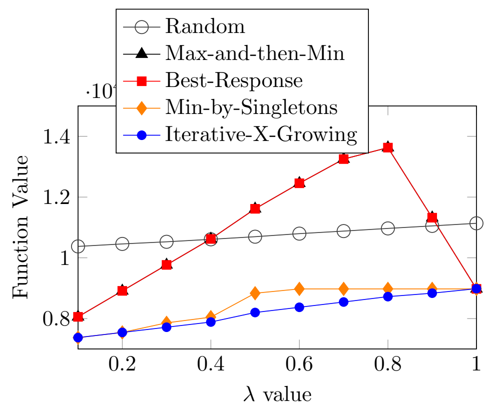
$$f(X \cup Y) = \sqrt[3]{\sum_{v\in\mathcal{N}\setminus X}\sum_{u\in Y} M^3_{u,v}} - \frac{1}{|\mathcal{N}_2|}\sqrt[3]{\sum_{u\in Y}\sum_{v\in Y} M^3_{u,v}} + \lambda\cdot|X|\cdot\sqrt[3]{k} \enspace . \tag{6}$$

Here, $M_{u,v}$ is the similarity score between images u and v, which is assumed to be non-negative and symmetric (i.e., $M_{u,v} = M_{v,u} \geq 0$); and $\lambda \in [0, 1]$ is a regularization parameter affecting the number of elements added by the adversary. Choosing a larger value for λ results in a larger set X, and thus, less adversarial images being added. The objective function f is jointly-submodular and non-negative (the proof is very similar to the proof that the function in Equation (7) has these properties, and therefore, we omit it). Since we are interested in finding an attacker set X that is good against the best summary set Y of size k, the optimization problem that we aim to solve is

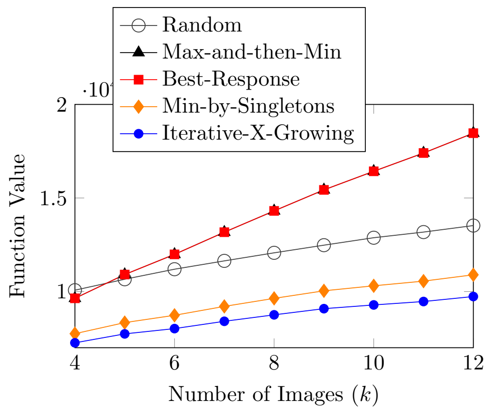
$$\min_{X\subseteq\mathcal{N}_1}\max_{\substack{Y\subseteq\mathcal{N}_2\\|Y|\leq k}} f(X \cup Y) \enspace .$$

Our experiments for this application are based on a subset of the CIFAR-10 data set (Krizhevsky, 2009). This subset includes 10,000 tiny images belonging to 10 classes. Each image consists of 32×32 RGB pixels, and is thus, represented by a 3,072 dimensional vector, and the cosine similarity method was used to compute similarities between images. In order to keep the running time computationally tractable, we randomly sampled from the data set in each experiment disjoint sets $\mathcal{N}_1$ and $\mathcal{N}_2$ of sizes $|\mathcal{N}_1| = 2{,}000$ and $|\mathcal{N}_2| = 250$.

In our experiments, we study the change in the quality of the summaries obtained by the various algorithms and benchmarks as a function of the allowed number k of images and the regularization parameter λ. Figure 3a presents



(a) Results for $k = 5$



(b) Results for $\lambda = 0.5$

Figure 3: Empirical results for adversarial attack on image summarization. Both plots compares the performance of our algorithms Min-by-Singletons and Iterative-X-Growing with 3 benchmarks for different value of the regularization parameter λ and the cardinality parameter k.

the outputs of our algorithms Min-by-Singletons and Iterative-X-Growing (with $\beta = 0.2$) and three benchmarks for $k = 5$ and a varying regularization parameter λ. Figure 3b presents the outputs of the same algorithms and benchmarks for $\lambda = 0.5$ and a varying limitation k on the number of images in the summary. One can observe that both of our algorithms consistently outperform the benchmarks of Best-Response, Max-and-then-Min and Random, with the more involved algorithm Iterative-X-Growing tending to do better than the simpler algorithm Min-by-Singletons. Both figures are based on averaging 400 executions of the algorithms, leading to a standard error of the mean of less than 10 for all data points. It is also worth noting that the basic scarecrow benchmark "Random" outperforms the Best-Response and Max-and-then-Min benchmarks in many cases. This hints that the last heuristics are unreliable despite being natural, and highlights the significance of the methods we propose.

D.2 Robust Ride-Share Optimization

In the "Robust Ride-Share Optimization" application, our primary objective is to determine the most suitable waiting locations for idle taxi drivers based on taxi order history. This problem was previously formalized as a traditional facility location problem (Mitrovic et al., 2018). However, in the current work, we look for a more robust set of waiting locations. Often some locations are inaccessible (for example, due to road maintenance). Hence, we wish to find a robust set of waiting locations that effectively minimizes the distance between each customer and her closest driver even when some of the locations are inaccessible.

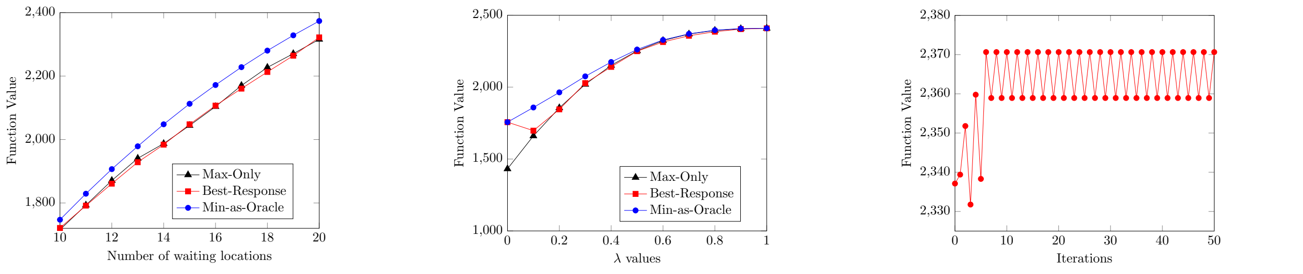
The objective function we use to solve the above problem is technically identical to the jointly-submodular function given by (3). However, now $\mathcal{N}_1$ represents the (client) pickup locations that might be inaccessible due to traffic (while $\mathcal{N}_2$ remains the set of potential waiting locations for idle drivers). Furthermore, we now need to perform max-min optimization on this objective function since we look for a set Y of up to k waiting locations that is good regardless of which pickup locations become inaccessible.

In our experiments, we used again the Uber data set (Uber) (see Section 3.2). To ensure computational tractability, in each execution of our experiments, we randomly selected from this data set a subset of $|\mathcal{N}| = 6{,}000$ pickup locations within the region of Manhattan. Then, we chose the set $\mathcal{N}_1$ to consist of all the pickup locations that have a latitude value greater than 40.8, or less than 40.73. This set represents the pickup locations that are potentially unavailable (for example, due to traffic). Furthermore, we randomly selected a subset of 400 pickup locations from the set $\mathcal{N}$ to constitute the set $\mathcal{N}_2$. This set represents the potential waiting locations for idle drivers.

In the first experiment, we fixed the regularization parameter λ to 0.35 and varied the number of allowed waiting locations. Figure 4a depicts the outputs for this experiment for our algorithm Min-as-Oracle and two benchmarks (averaged over 10 executions of the experiment). One can observe that Min-as-Oracle consistently surpasses the two benchmarks. The two other benchmarks (Random and Top-k) where also included in this experiment and the next one, but are excluded from the figures since their outputs are worse by a factor of at least 2 comapred to the presented methods. We also note that in both experiments the standard error of the mean is less than 10 for all data points.

In the second experiment, we fixed the maximum number of waiting locations to be 15, and varied λ. The results of this experimented are depicted by Figure 4b (again, averaged over 10 executions of the experiment). Once again, our proposed method, Min-as-Oracle, demonstrates superior performance compared to the bechnmarks, with the gap being significant for lower values of λ.

As the third experiment for this application, we conducted a more in depth analysis of the Best-Response technique. Figure 4c graphically presents the objective function value obtained by a typical execution of Best-Response after a varying number of iterations (for $\lambda = 0.35$ and an upper bound of 20 on the number of waiting locations). It is apparent that Best-Response does not converge for this execution. Furthermore, Min-as-Oracle demonstrates better performance even with respect to the best performance of Best-Response for any number of iterations between 1 and 50.



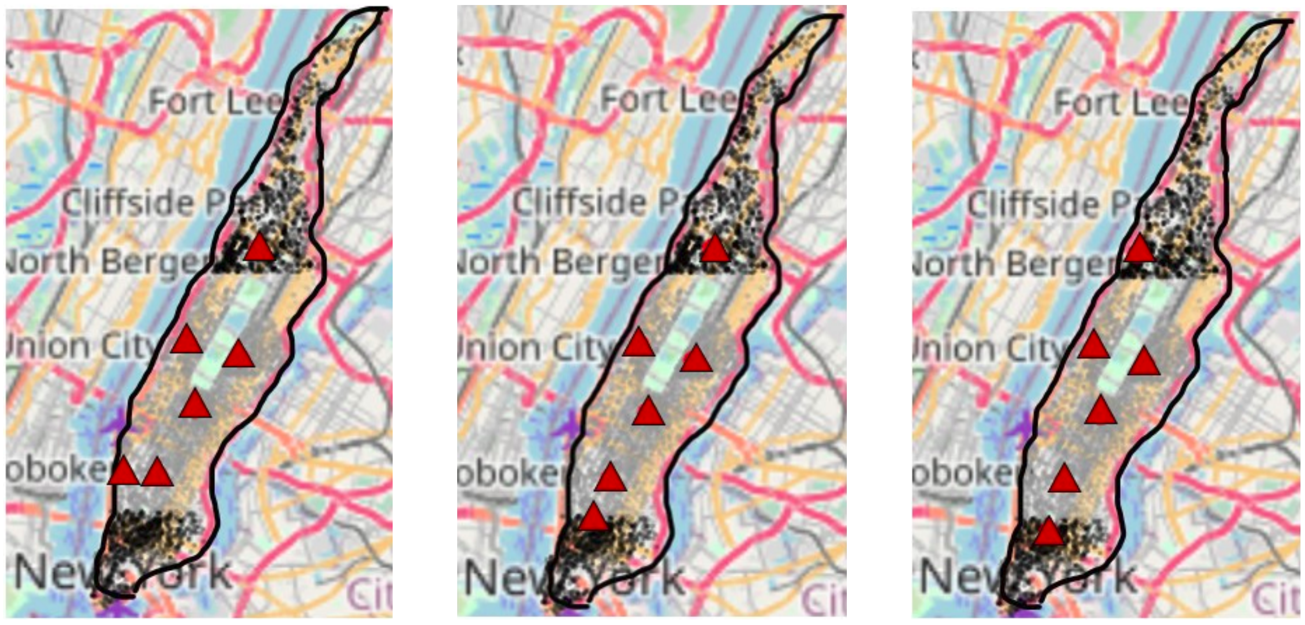
(a) $\lambda = 0.35$ (b) 15 waiting locations (c) Behavior of Best-Response for $\lambda = 0.35$ and 20 waiting locations

Figure 4: Empirical results for robust ride-share optimization. Figures (a) and (b) compare the performance of our algorithm Min-as-Oracle with 2 benchmarks for different value of λ and bounds on the number of weighting locations. Figure (c) depicts the value of the output of the Best-Response method as a function of the number of iterations performed.

Our last experiment for this section aims to give a more intuitive point of view on the performance of our algorithm (Min-as-Oracle). Figure 5 depicts the results of this algorithm on maps of Manhattan for three different values of λ (0.2, 0.4 and 0.8). To make the maps easy to read, we allowed the algorithm to select only 6 waiting locations for idle drivers, and the locations suggested by the algorithm are marked with red triangles on the maps. We have also marked on the maps the pick up locations of $\mathcal{N}$. The black dots represent the waiting locations that are inaccessible, while the light gray dots indicate the accessible pickup locations. Intuitively, the regularization parameter λ captures in this application the probability of pickup locations in $\mathcal{N}_1$ to be accessible. For example, when $\lambda = 0$, it is assumed that all locations in $\mathcal{N}_1$ are inaccessible, whereas $\lambda = 1$ means that all locations in $\mathcal{N}_1$ are assumed to be accessible. This intuitive role of λ is demonstrated in Figure 5 in the following sense. As the value of λ increases, the number of red triangles in the figure inside the areas of the black dots tends to increase, and furthermore, the locations of theses triangles are pushed deeper into these areas.

D.3 Prompt Engineering for Dialog State Tracking

In this section, we consider the problem of selecting example (input, output) pairs for zero-shot in-context learning. In this application, the objective is to design prompts for the task of dialog state tracking (DST) on the MultiWOZ 2.4 data set (Budzianowski et al., 2018). Following Hu et al. (2022), we recast this as a text-to-SQL problem in order to prompt the GPT-Neo (Black et al., 2021) and OpenAI Codex (`code-davinci-002`) (Chen et al., 2021) code generation models. These base models are adapted to DST with a combination of subset selection and in-context learning. First, a corpus of (previous dialog state, current dialog turn, SQL query) tuples is constructed from the training dialogs. Given a new input u_0, our prompt consists of 1) tabular representations of the dialog state ontology, 2) natural language instructions to query these tables using valid SQL given a task-oriented dialog turn, and 3) examples selected from the corpus by maximizing an objective function.



(a) $\lambda = 0.2$ (b) $\lambda = 0.4$ (c) $\lambda = 0.8$

Figure 5: The results of Min-as-Oracle for 3 different values of λ. The red triangles represent waiting locations chosen by the algorithm, the light gray dots represent always accessible pick-up locations, and the black dots represent possibly inaccessible pick-up locations.

Let u_0 be the input query to a large language model. Each input u_0 contains a list of (key, value) pairs representing the previous dialog state predictions along with the text of the current dialog turn. We would like its prompt to be robust to incorrect predictions of the previous dialog states, as well as text variation such as misspellings. Let $\mathcal{N}_1$ be a set of *perturbed* inputs drawn from a small neighborhood around u_0. These perturbed inputs are constructed by randomly editing up to 2 slots and/or values in the dialog state, and additionally dropping up to 15% of tokens from the most recent dialog turn. In cases where u_0 is initially incorrect, examples that are similar to the perturbed inputs from $\mathcal{N}_1$ improve the final prompt. Let $\mathcal{N} = \mathcal{N}_1 \cup \{u_0\}$, and let $\mathcal{N}_2$ be the ground set of candidate examples. Given a set of examples $Y \subseteq \mathcal{N}_2$ and a set of perturbed inputs $X \subseteq \mathcal{N}_1$, we define the following score function.

$$f(X \uplus Y) = \sum_{u \in \mathcal{N} \setminus X} \sum_{v \in Y} s_{u,v} - \frac{\alpha}{|\mathcal{N}_2|} \cdot \sum_{v \in Y} \sum_{u \in Y} s_{u,v} + \lambda \cdot |X| + |\mathcal{N}_2| \ . \tag{7}$$

Here, $0 \leq s_{u,v} \leq 1$ is the symmetric similarity score between examples u, v (the similarity score is computed by embedding both examples with a pretrained SBERT model (Reimers and Gurevych, 2019), and then computing cosine similarity of the two embeddings), and $\lambda \geq 0$ and $0 \leq \alpha \leq 1$ are regularization parameters. The parameter α explicitly trades off recommendation quality and diversity. Since we are interested in finding a set of candidate examples Y that is good against the worst case set X of perturbed inputs, we would like to optimize $\max_{Y \subseteq \mathcal{N}_2, |Y| \leq k} \min_{X \in \mathcal{N}_1} f(X \uplus Y)$, where k is an upper bound on the number of examples to include in the prompt. By an argument similar to the proof of Lemma 3.2, the objective function (7) is a non-negative jointly-submodular function.

Initially, the GPT-Neo-Small generative model was evaluated with all possible combination of values for the regularization parameters from the grid $\lambda \in \{0, 0.5, 0.75, 0.9, 2.5\}$ and $\alpha \in \{0, 0.1, 0.3, 0.5, 0.7\}$. The best parameters ($\lambda = 0.9, \alpha = 0.5$) were then used for the other generative models. Following Section 5 of Hu et al. (2022), all retrieval models were evaluated on inputs obtained by randomly sampling 10% of the MultiWOZ validation set, and all results were averaged over 3 different candidate sets, which are randomly sampled 5%

subsets of the MultiWOZ training set. We set ($k = 5$, $|\mathcal{N}_1| = 20$) for GPT-Neo models and ($k = 10$, $|\mathcal{N}_1| = 4$) for the OpenAI Codex model.

In our experiments, we have compared Min-as-Oracle with the our standard max-min benchmarks Top-k, Max-Only and Best-Response, and also with a baseline termed "Non-robust Top-k" from Hu et al. (2022). For Top-k, Max-Only, Best-Response and Min-as-Oracle, we first retrieved a ground set of size $|\mathcal{N}_2| = 100k$ candidates using the precomputed KD Tree, and only then selected the output set Y using the retrieval algorithm. Table 3 shows the Joint F1 score for each of the above-mentioned methods. Results for GPT-Neo models are averaged over 4 random seeds. One can observe that prompting with our robust formulation outperforms the Non-Robust Top-k baseline by as much as 1.5%. Among the algorithms using the robust formulation, our proposed algorithm Min-as-Oracle is consistently the best or 2nd best. Table 3 also shows that Min-as-Oracle achieves the highest objective value in all cases. Note that Min-as-Oracle has theoretical guarantees for both its convergence and approximation ratio, whereas Sections 3.2 and D.2 demonstrate that the Best-Response heuristic diverges for some instances.

Table 3: Dialog state tracking performance and objective values for different language models and retrieval algorithms. Best values are in **bold**.

Generative Model	Retrieval Algorithm	Joint F1	Objective Value
GPT-Neo-Small	Random	0.0480	25.259
	Non-robust Top-k (Hu et al., 2022)	0.3249	26.165
	Top-k	0.2787	26.125
	Max-Only	0.2783	26.134
	Best-Response	**0.3251**	26.165
	Min-as-Oracle	0.3022	**26.168**
GPT-Neo-Large	Random	0.2275	25.254
	Non-robust Top-k (Hu et al., 2022)	0.4872	26.164
	Top-k	0.4821	26.127
	Max-Only	0.4830	26.134
	Best-Response	0.4845	26.165
	Min-as-Oracle	**0.5020**	**26.168**
Codex-Davinci	Random	0.8273	17.410
	Non-robust Top-k (Hu et al., 2022)	0.8929	19.021
	Top-k	0.8974	18.954
	Max-Only	0.8913	18.953
	Best-Response	0.8972	19.022
	Min-as-Oracle	**0.8991**	**19.027**

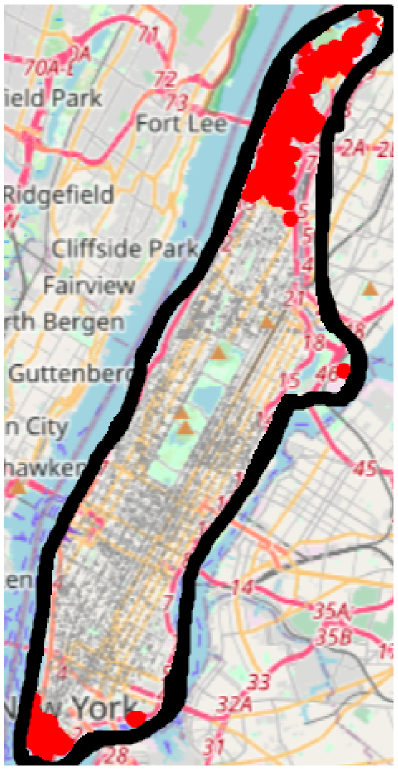
D.4 Additional Figures For Ride-Share Difficulty Kernelization

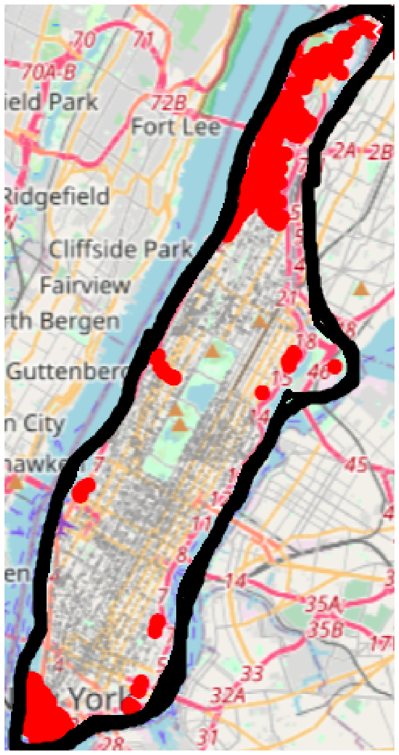
In this section, we provide a graphical representation of the experimental outcomes discussed in Section 3 for the Ride-Share Difficulty Kernelization application. Figure 6 demonstrates that the locations chosen by our algorithm, based on the objective function (3) used for the application, create a spatial arrangement resembling a "frame" encompassing the Manhattan area. This agrees with the intuitive expectation from a well-structured kernelization, demonstrating that the objective function (3) is a good fit for the Ride-Share Difficulty Kernelization application.

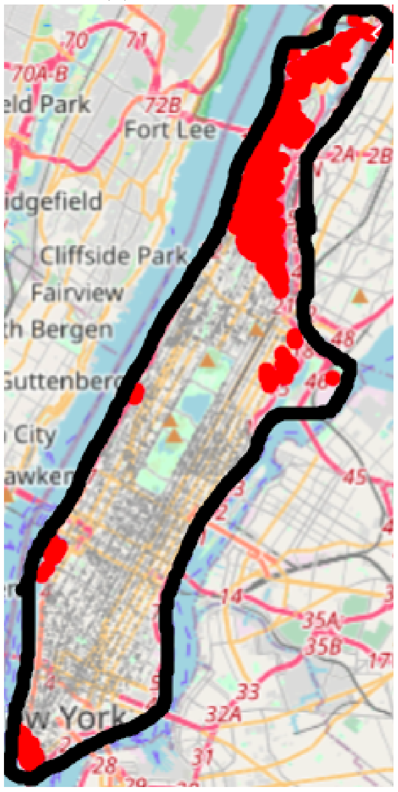
E BENCHMARKS AND ALGORITHM IMPLEMENTATIONS

In this section we define all the benchmarks that we compare in Section 3 and Section D against our algorithms. We then discuss the implementation details of these benchmarks and our algorithms.

- Random: Returns a random feasible solution. In the max-min setting this means random k elements from the ground set $\mathcal{N}_1$, and in the min-max setting this means a random subset of $\mathcal{N}_2$.
- Max-Only: This benchmark makes sense only in the max-min setting. It uses a submodular maximization algorithm to find a feasible set Y (approximately) maximizing the objective for $X = \varnothing$.


(a) $\lambda = 0.1$, $k = 8$


(b) $\lambda = 0.25$, $k = 8$


(c) $\lambda = 0.1$, $k = 4$

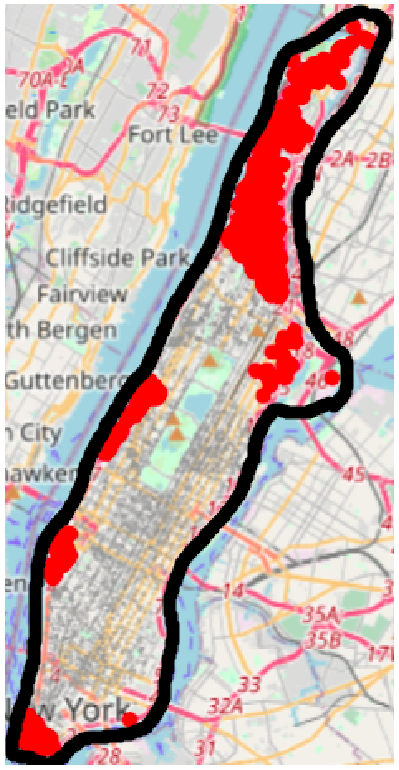

(d) $\lambda = 0.25$, $k = 4$

Figure 6: The results of our algorithm Iterative-X-Growing for different values of λ (the regularization parameter) and k (the number of taxis). The red dots represent the pick-up locations in the difficulty kernel chosen by the algorithm (the other pick-up locations are marked by light gray dots).

- Max-and-then-Min: A variant of Max-Only for use in the min-max setting. It returns a set X minimizing the objective given the set Y chosen by Max-Only. Note that this is essentially equivalent to a single iteration of Best-Response.

- Top-k: This benchmark makes sense only in the max-min setting. It returns the k singletons from $\mathcal{N}_2$ with the maximum value, where the value of every singleton $u \in \mathcal{N}_2$ is defined as $\min_{X \subseteq \mathcal{N}_1} f(X \mathbin{\dot\cup} \{u\})$.

- Best-Response: This benchmark proceeds in iterations. In the first iteration, one obtains a subset $Y \in \mathcal{F}_2$ (approximately) maximizing $f(Y)$ through the execution of a maximization algorithm, which is followed by finding a set $X \subseteq \mathcal{N}_2$ minimizing $f(X \mathbin{\dot\cup} Y)$ by running a minimization algorithm. Subsequent iterations are similar to the first iteration, except that the set Y chosen in these iterations is a set that (approximately) maximizes $f(X \mathbin{\dot\cup} Y)$, where X is the minimizing set chosen in the previous iteration. The output is then the last set Y in the max-min setting, and the last set X in the min-max setting.

In most of our applications, we aim to optimize objectives that are not $\mathcal{N}_2$-monotone, which requires a procedure for (approximate) maximization of non-monotone submodular functions. As mentioned in Section 1.1, the state-of-the-art approximation guarantee for the case in which the objective function f is not guaranteed to be monotone is currently 0.385 (Buchbinder and Feldman, 2019). However, the algorithm obtaining this approximation ratio is quite involved, which limits its practicality. Arguably, the state-of-the-art approximation ratio obtained by a "simple" algorithm is $1/e$-approximation obtained by an algorithm called Random Greedy (Buchbinder et al., 2014). In practice, the performance of this algorithm is comparable to that of the standard greedy algorithm, despite the last algorithm not having any approximation guarantee for non-monotone objective functions. Hence, throughout the experiments, the maximization component used in all the relevant benchmarks and algorithms is either the standard greedy algorithm or an accelerated version of it (suggested by Badanidiyuru and Vondrák (2014)) named Threshold Greedy.

In our experiment we often report the values of the objective function corresponding to the output sets produced by the various benchmarks and algorithms. In the max-min setting, given an output set X, computing the objective value is done by utilizing an efficient minimizing algorithm to identify a minimizing set X. In the min-max setting, the situation is more involved as calculating the true objective value for given an output set X cannot be done efficiently in sub-exponential time (as it corresponds to maximizing a submodular function subject to a cardinality constraint). Therefore, we use Threshold Greedy algorithm mentioned above to find a set Y that approximately maximize the objective with respect to X, and then report the value corresponding to this set Y as a proxy for the true objective value.

Our experiments for the min-max setting use a slightly modified version of Iterative-X-Growing (Algorithm 1). Specifically, we make two modifications to the algorithm.

- Iterative-X-Growing grows a solution in iterations. As written, it outputs the set obtained after the last iteration. However, we chose to output instead the best set obtained after any number of iterations. This is a standard modification often used when applying to practice an iterative theoretical algorithm.

- Line 4 of Iterative-X-Growing looks for a set X'_i that minimizes an expression involving two terms. The first of these terms $\sqrt{n_1} \cdot f(X \cup X_{i-1})$ has the large coefficient $\sqrt{n_1}$. The value of this coefficient was chosen to fit the largest number of possible iterations that the algorithm may perform $(n_1 + 1)$. However, in practice we found that the algorithm usually makes very few iterations. Thus, the use of the large coefficient $\sqrt{n_1}$ becomes sub-optimal. To truly show the empirical performance of Iterative-X-Growing, we replaced the coefficient $\sqrt{n_1}$ with a parameter β whose value is chosen based on the application in question.

All prompt engineering experiments were run using a single NVIDIA A10 GPU. Running inference with the Contriever and GPT-Neo-Large models required a server with 128GB of memory, and all other parts of the pipeline required less than 16GB of memory.

All other experiments were run using 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz CPU, requiring less than 32GB of CPU memory and no GPU.

F ADDITIONAL OMITTED PROOFS

F.1 Proof of Lemma 3.1

In this section we prove Lemma 3.1, which we repeat here for convenience.

Lemma 3.1. *The objective function* (2) *is a non-negative jointly-submodular function.*

Proof. Observe that the objective function (2) is a conical combination of three terms. Below we explain why each one of these terms is non-negative and jointly-submodular, which immediately implies that the entire objective function also has these properties.

The second term is $\sum_{u \in \mathcal{N}_1 \setminus X} \sum_{v \in Y} s_{u,v}$. This term can be viewed as the cut function of a directed bipartite graph in which the elements of $\mathcal{N}_1$ and $\mathcal{N}_2$ form the two sides of the graph, and for every $u \in \mathcal{N}_1$ and $v \in \mathcal{N}_2$ the graph includes an edge from v to u whose weight is $s_{u,v}$. Directed graph functions are known to be non-negative and submodular over the set of elements of the graph, which translates into joint-submodularity in our terminology since the both the elements of $\mathcal{N}_1$ and $\mathcal{N}_2$ are vertices of the graph.

The third term is $|X|$, which is a non-negative linear function, and thus, also jointly-submodular.

It remains to consider the first term, namely $\sum_{v \in \mathcal{N}_1 \setminus X} \max_{u \in Y} s_{u,v}$. This term is clearly non-negative, so we concentrate below on proving that it is jointly submodular. Recall that f is jointly-submodular if

$$f(u \mid Y' \uplus X') \geq f(u \mid Y \uplus X) \qquad \forall\, X' \subseteq X \subseteq \mathcal{N}_1, Y' \subseteq Y \subseteq \mathcal{N}_2, u \in (\mathcal{N}_1 \uplus \mathcal{N}_2) \setminus (X \uplus Y) \enspace .$$

To prove that our objective function obeys this inequality, there are two scenarios to consider, based on whether u belongs to the set $\mathcal{N}_1$ or $\mathcal{N}_2$.

Case 1: The element u belongs to $\mathcal{N}_1$. Here,

$$f(u \mid Y' \uplus X') - f(u \mid Y \uplus X) = \max_{v \in Y} s_{u,v} - \max_{v \in Y'} s_{u,v} \geq 0 \enspace .$$

Case 2: the element u belongs to $\mathcal{N}_2$. Observe that, in this case,

$$\begin{aligned} f(u \mid Y' \uplus X') &= \sum_{\mathcal{N}_1 \setminus X'} \max\{0, s_{u,v} - \max_{v' \in Y'} s_{u,v'}\} \\ &\geq \sum_{\mathcal{N}_1 \setminus X} \max\{0, s_{u,v} - \max_{v' \in Y} s_{u,v}\} = f(u \mid Y \uplus X) \enspace . \end{aligned}$$ □

F.2 Proof of Lemma 3.2

In this section we prove Lemma 3.2, which we repeat here for convenience.

Lemma 3.2. *The objective function* (3) *is a non-negative jointly-submodular function.*

Proof. First, we shall establish that the objective function is non-negative by demonstrating that the first term of the function is consistently greater than the subsequent term. This is established through the following inequality.

$$\sum_{v \in \mathcal{N} \setminus X} \max_{u \in Y} s_{u,v} \geq \sum_{v \in \mathcal{N}_2} \max_{u \in Y} s_{u,v} \geq \frac{1}{|Y|} \cdot \sum_{v \in \mathcal{N}_2} \sum_{u \in Y} s_{u,v} \geq \frac{1}{|\mathcal{N}_2|} \cdot \sum_{v \in Y} \sum_{u \in Y} s_{u,v} \enspace .$$

Next, we demonstrate that the objective function $f(X, Y)$ is jointly-submodular. Recall that f is jointly-submodular if

$$f(u \mid Y' \uplus X') \geq f(u \mid Y \uplus X) \qquad \forall\, X' \subseteq X \subseteq \mathcal{N}_1, Y' \subseteq Y \subseteq \mathcal{N}_2, u \in (\mathcal{N}_1 \uplus \mathcal{N}_2) \setminus (X \uplus Y) \enspace .$$

To prove that our objective function obeys this inequality, there are two scenarios to consider, based on whether u belongs to the set $\mathcal{N}_1$ or $\mathcal{N}_2$.

Case 1: The element u belongs to $\mathcal{N}_1$. Here, $f(u \mid Y' \cupdot X') - f(u \mid Y \cupdot X) = \max_{v \in Y} s_{u,v} - \max_{v \in Y'} s_{u,v} \geq 0$.

Case 2: The element u belongs to $\mathcal{N}_2$. Let $\phi(Y, w, J) = \sum_{v \in J}(\max_{u \in Y+w} s_{u,v} - \max_{u \in Y} s_{u,v})$; and note that, for every two sets $Y' \subseteq Y \subseteq \mathcal{N}_2$, set $J \subseteq \mathcal{N}_1$ and element $u \in \mathcal{N}_2 \setminus T$, $\phi(Y', u, J) \geq \phi(Y, u, J)$. Using this notation, we get that in this case (the case of $u \in \mathcal{N}_2$)

- $f(u \mid Y' \cupdot X') = \phi(Y', \{u\}, \mathcal{N} \setminus X') - \frac{1}{|\mathcal{N}_2|}\big(2 \cdot \sum_{v \in Y'} s_{u,v} + s_{u,u}\big)$, and
- $f(u \mid Y \cupdot X) = \phi(Y, \{u\}, \mathcal{N} \setminus X) - \frac{1}{|\mathcal{N}_2|}\big(2 \cdot \sum_{v \in Y} s_{u,v} + s_{u,u}\big)$.

Thus,

$$f(u \mid Y' \cupdot X') - f(u \mid Y \cupdot X) = \phi(Y', \{u\}, \mathcal{N} \setminus X') - \phi(Y, \{u\}, \mathcal{N} \setminus X) + \frac{2}{|\mathcal{N}_2|} \cdot \sum_{v \in Y \setminus Y'} s_{u,v} \geq 0 \enspace ,$$

where the last inequality holds since $\phi(Y', \{u\}, \mathcal{N} \setminus X') - \phi(Y, \{u\}, \mathcal{N} \setminus X) \geq 0$ and $s_{u,v} \geq 0$ for any $u, v \in \mathcal{N}$. $\square$

Chapter 7

Conclusions and Future Work

In this thesis, we have developed a series of methods for improving the efficiency and scalability of machine learning systems through coreset constructions and submodular optimization. Our contributions span both theoretical and practical aspects, covering offline and online settings, and addressing a variety of challenges in deep learning and optimization.

We began by proposing novel coreset sampling strategies for deep neural networks, introducing deterministic and sensitivity-aware techniques that improve both accuracy and fairness across classes. These methods demonstrated superior performance in preserving classification metrics compared to traditional loss-based sampling.

Next, we explored the structure of submodular functions, showing that many real-world objectives—though non-monotone—exhibit partial monotonicity. By leveraging this insight, we achieved improved approximation guarantees in submodular maximization, bridging the gap between theory and practice.

We then turned our attention to continuous domains, resolving fundamental questions in the approximability of non-monotone DR-submodular maximization under general convex constraints, in both offline and online settings. Our work introduced tight approximation bounds and efficient algorithms for a large class of problems.

In a related direction, we studied the geometry of constraint sets, presenting new algorithms and hardness results for submodular maximization over general versus down-closed convex sets. This work reveals the profound effect that constraint structure has on achievable guarantees.

Finally, we introduced the submodular minimax optimization framework, presenting algorithms with provable guarantees for adversarial and robust scenarios. We demonstrated its potential in diverse applications, such as prompt engineering and robust training, further illustrating the versatility of submodular optimization tools.

Taken together, these results highlight the importance of structure—whether in data, objectives, or constraints—in enabling efficient algorithmic design. Looking forward, future work could build on the foundations developed in this thesis by improving the training and fine-tuning pipelines of generative models, and extending these techniques to active learning and graph-based learning. Active learning [CGJ96] is particularly important in domains where data acquisition is costly, such as scientific modeling or neural PDE solvers [BWW], while graph learning [WPC$^+$20] presents challenges of scalability and expressiveness due to its non-Euclidean structure. Across these areas, advancing efficiency, principled design, and adaptability of deep learning systems remains an exciting and open frontier for future research.

Bibliography

[AHPV04] P. Agarwal, S. Har-Peled, and K. Varadarajan. Approximating extent measures of points. *Journal of the ACM*, 51(4):606–635, 2004.

[BC08] Mihai Bădoiu and Kenneth L Clarkson. Optimal core-sets for balls. *Computational Geometry*, 40(1):14–22, 2008.

[BEL13] Maria-Florina F Balcan, Steven Ehrlich, and Yingyu Liang. Distributed k-means and k-median clustering on general topologies. In *Advances in Neural Information Processing Systems*, pages 1995–2003, 2013.

[BJKW19] Vladimir Braverman, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for ordered weighted clustering. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 744–753, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[BLK18] Olivier Bachem, Mario Lucic, and Andreas Krause. Scalable k-means clustering via lightweight coresets. In *KDD'18 Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1119–1127. ACM, 2018.

[BLL18] Olivier Bachem, Mario Lucic, and Silvio Lattanzi. One-shot coresets: The case of k-clustering. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 784–792, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR.

[BWW] Johannes Brandstetter, Daniel E Worrall, and Max Welling. Message passing neural pde solvers. In *International Conference on Learning Representations*.

[CGJ96] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.

[CIM+19] Ryan Curtain, Sungjin Im, Ben Moseley, Kirk Pruhs, and Alireza Samadian. On coresets for regularized loss minimization. *arXiv preprint arXiv:1905.10845*, 2019.

[ERR$^+$19] Andre Esteva, Alexandre Robicquet, Bharath Ramsundar, Volodymyr Kuleshov, Mark DePristo, Katherine Chou, Claire Cui, Greg Corrado, Sebastian Thrun, and Jeff Dean. A guide to deep learning in healthcare. *Nature medicine*, 25(1):24–29, 2019.

[Fel20] Dan Feldman. Core-sets: Updated survey. In *Sampling Techniques for Supervised or Unsupervised Tasks*, pages 23–44. Springer, 2020.

[FMSW10] Dan Feldman, Morteza Monemizadeh, Christian Sohler, and David P Woodruff. Coresets and sketches for high dimensional subspace approximation problems. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 630–649. SIAM, 2010.

[FRVR14] Dan Feldman, Guy Rossman, Mikhail Volkov, and Daniela Rus. Coresets for k-segmentation of streaming data. In *NIPS*, 2014.

[GBK$^+$24] Manel Guettala, Samir Bourekkache, Okba Kazar, Saad Harous, et al. Generative artificial intelligence in education: Advancing adaptive and personalized learning. *Acta Informatica Pragensia*, 13(3):460–489, 2024.

[GMS$^+$24] Meir Goldenberg, Loay Mualem, Amit Shahar, Sagi Snir, and Adi Akavia. Privacy-preserving biological age prediction over federated human methylation data using fully homomorphic encryption. *Genome Research*, 34(9):1324–1333, 2024.

[ID23] Sergiu-Alexandru Ionescu and Vlad Diaconita. Transforming financial decision-making: the interplay of ai, cloud computing and advanced data management technologies. *International Journal of Computers Communications & Control*, 18(6), 2023.

[JTMF20] Ibrahim Jubran, Murad Tukan, Alaa Maalouf, and Dan Feldman. Sets clustering. *arXiv preprint arXiv:2003.04135*, 2020.

[KL19] Zohar Karnin and Edo Liberty. Discrepancy, coresets, and sketches in machine learning. In *Conference on Learning Theory*, pages 1975–1993, 2019.

[MEFK24] Loay Raed Mualem, Ethan R Elenberg, Moran Feldman, and Amin Karbasi. Submodular minimax optimization: Finding effective sets. In *International Conference on Artificial Intelligence and Statistics*, pages 1081–1089. PMLR, 2024.

[MF22] Loay Mualem and Moran Feldman. Using partial monotonicity in submodular maximization. *Advances in Neural Information Processing Systems*, 35:2723–2736, 2022.

[MF23] Loay Mualem and Moran Feldman. Resolving the approximability of offline and online non-monotone dr-submodular maximization over general convex sets. In *International Conference on Artificial Intelligence and Statistics*, pages 2542–2564. PMLR, 2023.

[MJTF21] Alaa Maalouf, Ibrahim Jubran, Murad Tukan, and Dan Feldman. Coresets for the average case error for finite query sets. *Sensors*, 21(19):6689, 2021.

[MSSW18] Alexander Munteanu, Chris Schwiegelshohn, Christian Sohler, and David Woodruff. On coresets for logistic regression. In *Advances in Neural Information Processing Systems*, pages 6561–6570, 2018.

[MTF24] Loay Mualem, Murad Tukan, and Moran Feldman. Bridging the gap between general and down-closed convex sets in submodular maximization. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 1926–1934, 2024.

[Phi16] Jeff M Phillips. Coresets and sketches. *arXiv preprint arXiv:1601.00617*, 2016.

[Sha21] Mohammed Yousef Shaheen. Applications of artificial intelligence (ai) in healthcare: A review. *ScienceOpen Preprints*, 2021.

[TMF24] Murad Tukan, Loay Mualem, and Moran Feldman. Practical 0.385-approximation for submodular maximization subject to a cardinality constraint. *arXiv preprint arXiv:2405.13994*, 2024.

[TMM22] Murad Tukan, Loay Mualem, and Alaa Maalouf. Pruning neural networks via coresets and convex geometry: Towards no assumptions. *Advances in Neural Information Processing Systems*, 35:38003–38019, 2022.

[WFD$^+$23] Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023.

[WMX$^+$24] Tsun-Hsuan Wang, Alaa Maalouf, Wei Xiao, Yutong Ban, Alexander Amini, Guy Rosman, Sertac Karaman, and Daniela Rus. Drive anywhere: Generalizable end-to-end autonomous driving with multi-modal foundation models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6687–6694. IEEE, 2024.

[WPC$^+$20] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

טכניקות לבחירת תתי-קבוצות ביישומי למידה עמוקה/למידת מכונה: מהתיאוריה אל הפרקטיקה

לואי מועלם

<u>תקציר</u>

עבודה זו עוסקת בקבוצה רחבה של בעיות בלמידת מכונה ואופטימיזציה, המתמקדות בבחירה של תתי-קבוצות קטנות ומידעיות מתוך הנתונים, לצורך אימון, הסקה, וקבלת החלטות בצורה יעילה. במערכות בינה מלאכותית בקנה מידה גדול, הדרישות החישוביות והזיכרון הגבוהות של האלגוריתמים לעיתים קרובות אינן מאפשרות עיבוד של כלל הנתונים. על כן, עבודה זו בוחנת יסודות תיאורטיים ואלגוריתמיים לבחירת תתי-קבוצות, תוך שימוש בכלים כגון קבוצות-ליבה (coresets), אופטימיזציה תת-מודולרית (עמידה).

התרומות המרכזיות של המחקר כוללות:

1. פיתוח מסגרת גיזום (pruning) המבוססת על קונספטים מקבוצות ליבה (coresets) עבור רשתות נוירונים עמוקות, אשר מזהה ומסירה פרמטרים מיותרים כמעט ללא פגיעה בדיוק, ובכך מאפשרת פריסה מהירה ויעילה יותר.
2. ניתוח חדש לפונקציות תת-מודולריות חלקית-מונוטוניות, והצגה של תוצאות קירוב חזקות יותר, התורמות לבסיס תיאורטי משופר עבור מקסימיזציה תת-מודולרית בהקשרים מעשיים.
3. ביצענו ניתוח אמפירי של אלגוריתמים מתקדמים למקסימיזציה של פונקציות DR-submodular רציפות תחת אילוצים קמורים כלליים, יצרנו גרסאות מקוונות של אלגוריתמים אלה, והוכחנו כי הבטחות הקירוב שלהם הן הטובות ביותר שניתן לקבל.
4. הצעת גישת פירוק חדשה לקבוצות קמורות המגשרת בין אילוצים קמורים סגורים כלפי-מטה לבין אילוצים קמורים כלליים, תוך אפשרות למעבר חלק בין העולמות התיאורטיים המתאימים ויישום במצבים מקוונים ולא מקוונים.
5. גיבוש ופתרון (מקורב) של בעיות אופטימיזציה תת-מודולרית מסוג מינימקס, המדמה מצבי אי-ודאות ותסריטים יריביים, תוך פיתוח אלגוריתמים סקלאביליים עם הבטחות במקרה הגרוע.

לאורך כל העבודה מושם דגש על שילוב בין דיוק תיאורטי לבין יישומים מעשיים. השיטות המוצעות משלבות כלים מאנליזה קמורה, אופטימיזציה קומבינטורית ותיאוריה של למידת מכונה, ותורמות לשיפור הסקלאביליות, ההתאמה והחסינות של מערכות בינה מלאכותית מודרניות בתחומים מגוונים.

.

טכניקות לבחירת תתי-קבוצות ביישומי למידה עמוקה/למידת מכונה: מהתיאוריה אל הפרקטיקה

מאת: לואי מועלם
בהנחיית: פרופ' מורן פלדמן
פרופ' דן פלדמן

חיבור לשם קבלת התואר "דוקטור לפילוסופיה"
דוקטורט מאמרים

אוניברסיטת חיפה
הפקולטה למדעי החברה
החוג למדעי המחשב

יוני 2025

טכניקות לבחירת תתי-קבוצות ביישומי למידה עמוקה/למידת מכונה: מהתיאוריה אל הפרקטיקה

לואי מועלם

חיבור לשם קבלת התואר "דוקטור לפילוסופיה"
דוקטורט מאמרים

אוניברסיטת חיפה
הפקולטה למדעי החברה
החוג למדעי המחשב

יוני 2025