NEW DIRECTIONS IN MULTI-OBJECTIVE OPTIMIZATION WITH APPLICATIONS

A Dissertation
Presented to
The Academic Faculty

By

Jai Moondra

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Computer Science
Algorithms, Combinatorics, and Optimization

Georgia Institute of Technology

December 2025

NEW DIRECTIONS IN MULTI-OBJECTIVE OPTIMIZATION WITH APPLICATIONS

Thesis committee:

Dr. Swati Gupta
Sloan School of Management
Massachusetts Institute of Technology

Dr. Sahil Singla
School of Computer Science
Georgia Institute of Technology

Dr. Mohit Singh
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Milind Tambe
School of Engineering and Applied Sciences
Harvard University

Dr. Santosh S. Vempala
School of Computer Science
Georgia Institute of Technology

Date approved: 14 November 2025

I do not know what I may appear to the world, but to myself I seem to have been only like a boy playing on the sea-shore, and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me.

Isaac Newton

For my brother Yash

ACKNOWLEDGMENTS

I am greatly indebted to many people for the completion of this thesis. It is impossible to list everything that everyone has done for me, but one can always make an attempt.

First, I would like to thank my advisors, Swati Gupta – who gave me many new problems and challenged me to expand my intellectual boundaries, and Mohit Singh – who always gave me new ways to think about every problem. I am particularly thankful to both of them for being patient with my writing (which is still a work in progress).

I would also like to thank my committee members, Santosh Vempala, Sahil Singla, and Milind Tambe. Santosh taught me to see unexpected connections between different areas. Sahil gave me new ways to think about discrete optimization. Milind taught me the skill (and the importance) of juggling between theory and application.

My parents and my brother have always supported me in my endeavors. My father taught me the very first math and algorithms I ever knew. My mother taught me perseverance, which is perhaps the most invaluable tool in a mathematician's toolbox. My brother has taught me the value of playfulness, particularly in difficult situations.

My gratitude also extends to my friends and colleagues at Georgia Tech, including Aman Khullar (who taught me the value of balance), Harini Sridharan (who opened me to new perspectives), Harsh Maheshwari (who taught me much about converting theory to practice), Kethaki Varadhan (who was a kind sparring partner), Haripriya Rajagopalan (who shared and amplified my love for cooking), Hassan Mortagy (who taught me much about continuous optimization), Majid Farhadi (who was a mentor and a patient collaborator), Jad Salem (who inspired me to bridge continuous and discrete methods), Reuben Tate (who taught me much about quantum computing), and Max Dabagia (who inspired me to venture into new areas through his work).

I would also like to thank my co-authors during my PhD, including Majid Farhadi, Bryan Gard, Rohan Ghuge, Creston D. Herold, Sameer Kailasa, Cheol Woo Kim, Lingkai

Kong, Philip C. Lotshaw, Greg Mohler, Hassan Mortagy, Madeleine Pollack, Joel Rajakumar, Mirabel Reid, Reuben Tate, Prasad Tetali, Alejandro Toriello, and Shresth Verma. I have learnt a great deal from each of them, and will forever be grateful for the opportunity to have worked alongside them. In particular, Chapter 3, Chapter 4, Chapter 5, and Chapter 7 are joint works with Swati Gupta and Mohit Singh, and Chapter 6 is joint work with Cheol Woo Kim, Shresth Verma, Madeleine Pollack, Lingkai Kong, Milind Tambe, and Swati Gupta.

Finally, I would like to thank my undergraduate mentors Amit Kumar, Amitabha Tripathi, and Naveen Garg at IIT Delhi. Their teachings have shaped my thinking about computer science and mathematics, and still influence my work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

MLIJ MACHINELOADSIDENTICALJOBS

FSFL Fair Subsidized Facility Location

MDP Markov Decision Process

MORL Multi-objective reinforcement learning

OCO Online Convex Optimization

OEG Online Exponentiated Gradient

OMD Online Mirror Descent

OPGD Online Projected Gradient Descent

RL reinforcement learning

RLHF Reinforcement learning with human feedback

RMAB Restless multi-armed bandit

SUMMARY

This thesis introduces the *portfolio* framework for optimization with multiple objectives. A portfolio is a small set of solutions that approximately optimizes every objective under consideration. This approach recognizes the inherent plurality of objectives and provides a structured way to navigate competing goals. Instead of insisting on a single 'best' solution, portfolios offer a small number of high-quality solutions that together span the space of possible preferences. This work discusses the theoretical foundations, algorithmic techniques, and practical applications of this framework across various problems in machine learning and combinatorial optimization.

CHAPTER 1
INTRODUCTION

Optimization formalizes the task of choosing the 'best' decision under constraints. Discrete optimization, in particular, captures many of the central challenges of computer science – from foundational questions in algorithm design and complexity theory to applications in machine learning, resource allocation, and scientific modeling. Much of the literature on optimization agrees on the importance of translating the underlying goal of the problem into a mathematical objective function. Still, it assumes that this task has been carried out and often begins with a well-specified objective function.

This thesis studies *portfolios* for optimization problems, which are small sets of solutions that (approximately) optimize a large class of objective functions. There are at least three reasons to study how an optimization problem behaves as the objective function changes:

First, the task of translating the goals of the underlying problem into the objective function of the optimization involves making modeling choices. It is sometimes preferable for these choices to be made by the policy-maker (such as an elected representative or an organization official) rather than the algorithm designer. Just as importantly, the policy-maker should be able to understand how different modeling choices affect the outcomes. In practice, neither of these may hold. For example, consider a facility location problem where we seek to open some new pharmacies in an area while prioritizing poorer residents. This can be modeled as a standard facility location problem with an objective function where distances traveled by poorer residents get a higher weight. Both the choice of this form of objective function and the choice of the appropriate weight are modeling choices, often made by the algorithm designer. Understanding how the optimization problem behaves for different objectives helps understand the impact of these modeling choices and potentially

make decisions robust to these choices.

The second reason to study optimization problems with different objective functions is that different stakeholders often disagree on the goals of the optimization. In the facility location example, different residents might disagree on what is fair[1], and their notions of fairness will then translate into different objectives (even if they all agree on the modeling choices). This is particularly pertinent given the line of work that has established the incompatibility of different notions of fairness in various settings like resource allocation [8, 9] and machine learning [10, 11]. Similarly, the goals of *efficiency* and *fairness* are often at odds with each other in many settings, and the right tradeoff between these is unclear.

Finally, understanding how the (approximate) optimum of an optimization problem changes with a change in the objective gives insights into the structural and algorithmic aspects of the problem. For example, we could ask what algorithms for classical facility location problems[2] generalize to the fair version of the problem described above, and how the optimum solution changes as we change the weight on distances traveled by poorer residents. More broadly, such questions connect to fundamental algorithmic and structural problems: (1) robustness – how sensitive is the optimal solution to changes in the objective, and how broadly do existing algorithms generalize? (2) approximation – to what extent does a solution optimal for one objective approximate another objective? and (3) covering – what is the smallest collection of solutions required so that every objective has a near-optimal representative?

1.1 Portfolios

These challenges motivate a framework that does not insist on a single solution but instead provides a structured set of options. The central idea of this thesis is such a framework: *portfolios*.

[1] As has been widely discussed in philosophy [1, 2, 3], economics [4, 5, 6], and more recently computer science [7, 8].

[2] Classical facility location problems typically minimize some combination of user distances and facility

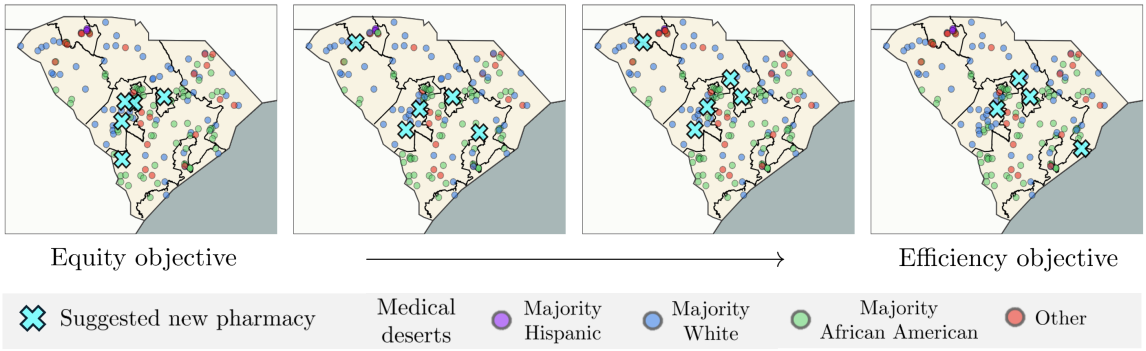


Figure 1.1: A portfolio of four solutions for opening ≤ 5 new pharmacies in South Carolina, USA to reduce 'medical deserts', which are poor regions that are far away from their nearest pharmacy. The left-most solution prioritizes equity, the right-most prioritizes efficiency, and the middle two represent intermediate trade-offs (see Chapter 3 for details).

Concretely, we develop methods that generate a portfolio of options satisfying two guarantees: (1) there is some option in this portfolio, no matter how the goals of optimization are balanced, and (2) there are not too many portfolio options. Formally, instead of selecting a single composite objective function that explicitly balances these various objectives, we develop techniques to provide a small set of "good enough" solutions that provably approximate any function from a given set of potential objectives of interest. For example, Figure 1.1 provides a portfolio of four options for opening five new pharmacies in the state of South Carolina, USA. Different options are obtained by optimizing objectives that interpolate between (some notion of) efficiency and equity, and prioritize different regions. Figure 1.2 gives another example.

Traditional approaches from multiobjective optimization [14, 15, 16, 17, 18, 19] typically fix specific notions of efficiency and equity and then study solutions along their Pareto frontier. This perspective has been fruitful, but it often does not provide approximation guarantees for an arbitrary (possibly infinite) class of objectives, nor does it analyze the trade-off between solution quality and the number of solutions one must retain (typically exponential in the problem dimension). In contrast, our framework of solution portfolios directly addresses both issues by asking for a small set of solutions that simultaneously

costs [12, 13].

approximate all objectives of interest in a given class of objectives.

Intuitively, a portfolio is like a 'menu' of candidate solutions: no matter which objective you care about, at least one item on the menu will perform nearly as well as the best possible solution for that objective. We now formalize this intuition:

Definition 1.1 (Minimization Portfolio)**.** *Given a minimization problem with feasible solutions $\mathcal{D}$, a (potentially infinite) class* $\mathbf{C}$ *of objective functions and a desired approximation $\alpha \geq 1$, a set $X \subseteq \mathcal{D}$ is called an α-approximate portfolio if for each $h \in \mathbf{C}$, there exists $x \in X$ that is an α-approximate solution to h, i.e., $h(x) \leq \alpha \min_{y \in \mathcal{D}} h(y)$.*

For maximization problems, these are defined completely analogously:

Definition 1.2 (Maximization Portfolio)**.** *Given a maximization problem with feasible solutions $\mathcal{D}$, a (potentially infinite) class* $\mathbf{C}$ *of objective functions and a desired approximation $\alpha \in (0, 1]$, a set $X \subseteq \mathcal{D}$ is called an α-approximate portfolio if for each $h \in \mathbf{C}$, there exists $x \in X$ that is an α-approximate solution to h, i.e., $h(x) \geq \alpha \max_{y \in \mathcal{D}} h(y)$.*

Small portfolios are important because they are practically useful: a decision-maker can only evaluate and compare a handful of solutions. The approximation factor measures the *quality* of the portfolio – poorer approximations imply that at least some objectives are poorly optimized by the portfolio. When both small size and good approximation are achieved, portfolios are not only usable in practice but also interpretable and theoretically robust.

If the portfolio size is restricted to 1, then this reduces to asking for a simultaneous approximation across all objectives (e.g., see [8, 9]), that is, a single solution that works reasonably well for all objectives at once. However, good simultaneous approximations rarely exist, and it can be challenging to find them efficiently (both in theory and practice).

For example, in facility location problems, no single solution may minimize the average person/group's distance (objective 1) as well as the most distant person/group's distance (objective 2). In scheduling problems, no single schedule may minimize the load on the

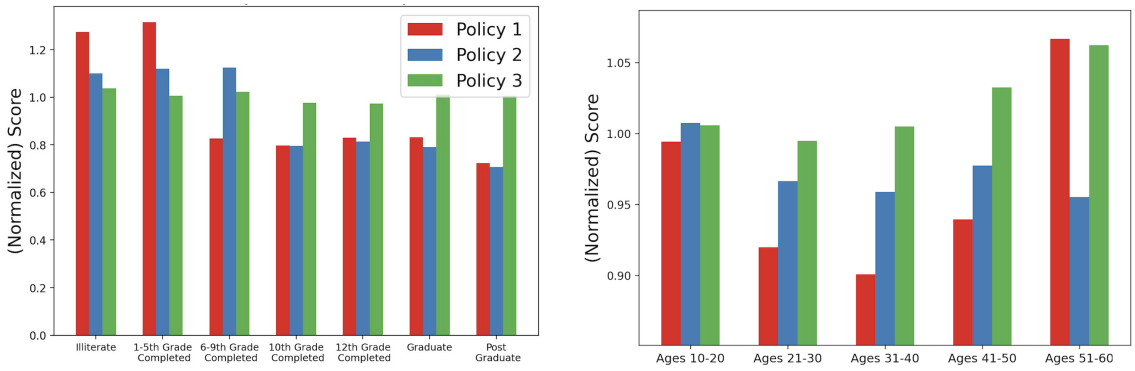


Figure 1.2: An example of a portfolio of three policies for the healthcare resource allocation problem considered in Chapter 6, modeled through reinforcement learning. The bar plots show total 'score' induced by each policy across different education (top) and age (bottom) brackets. A 'score' of 1.0 corresponds to a baseline policy (not shown) used for comparison. The three policies impact various education and age brackets differently, offering a diverse set of options for decision-makers. Policy 1 favours the less educated brackets, as well as the very young and older brackets. Policy 2 also favours the less educated population, but is more balanced. Policy 3 is the most balanced across education brackets.

average machine (objective 1) as well as the load on the most loaded machine (objective 2). In a resource allocation problem (modeled, for example, using reinforcement learning), no single solution may maximize the average person/group's utility (objective 1) as well as the minimum utility (objective 2) of a person/group. As we consider more than 2 objectives, the number of solutions required to satisfy them all (up to given approximation α) only grows.

That is, for various problems, as the set $\mathbf{C}$ of objectives grows larger, small portfolios may not even exist. Further, even for a given class $\mathbf{C}$ of objectives, it is not clear what the minimum size of an α-approximate portfolio needed to achieve a given approximation factor is. Larger portfolios are needed for better approximations, and the goal is to keep the size $|X|$ of the portfolio small. This thesis explores portfolios for various combinatorial and machine learning settings, aiming to understand the trade-offs between portfolio size $|X|$ and approximation α for different classes $\mathbf{C}$ of objectives and domains $\mathcal{D}$. See Figure 1.3 for an illustrative example.

Similar to classical (single-objective) optimization, the existence of an α-approximate

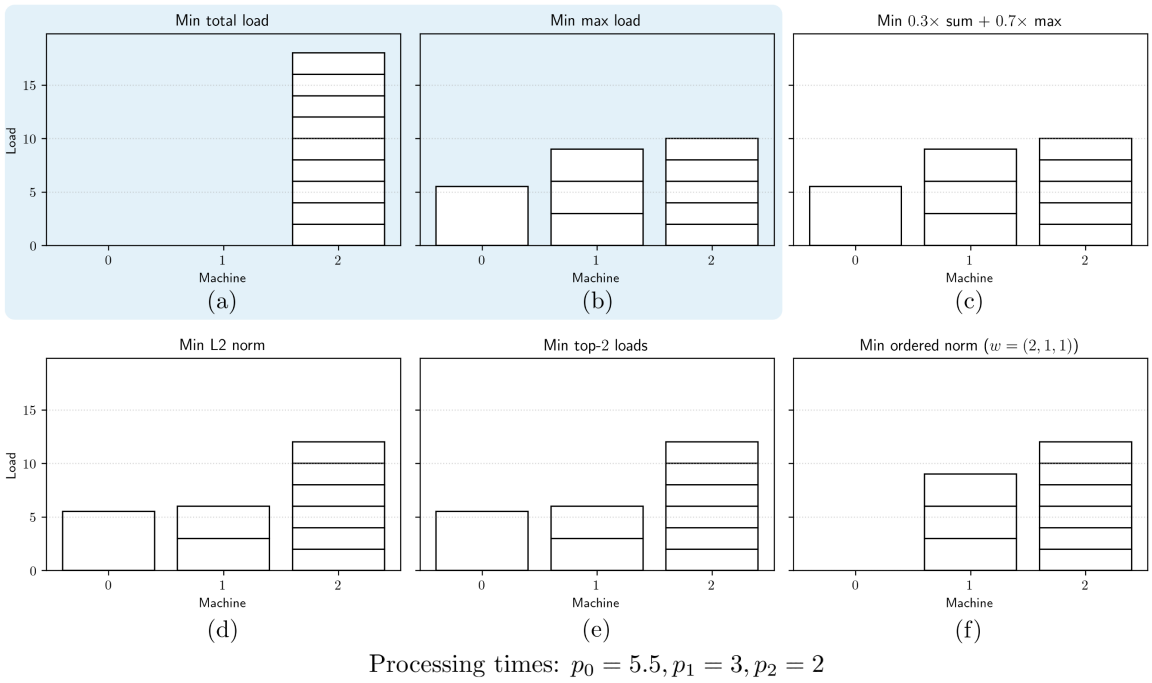


Figure 1.3: An example illustrating how different objective functions can yield distinct optimal solutions. We consider a scheduling problem with $d = 3$ machines with processing times $p_0 = 5.5$, $p_1 = 3$, and $p_2 = 2$, and $n = 9$ identical jobs. The load vector $x = (x_0, x_1, x_2)$ denotes the total load on each machine. The six subfigures correspond to minimizing: (1) the total load $\sum_i x_i$, (2) the maximum load $\max_i x_i$, (3) a convex combination $0.3 \sum_i x_i + 0.7 \max_i x_i$, (4) the L_2 norm $\|x\|_2$, (5) the sum of the two highest loads, and (6) $2 \times$ highest load + second highest load + least load, i.e., the ordered norm with weights $w = (2, 1, 1)$ (see Chapter 2). These six objectives yield four distinct optimal schedules (a, b, d, f), implying that an optimal ($\alpha = 1$) portfolio for these objectives must have size 4. The two solutions highlighted in blue (a, b) form a 1.1-approximate portfolio for all 6 objectives. The single schedule (b) minimizing $\max_i x_i$ forms a 1.4-approximate portfolio for these 6 objectives.

portfolio does not automatically imply that we can find one efficiently. This leads to two distinct questions: (1) What is the minimum size of an α-approximate portfolio, regardless of computation? (2) What is the minimum size of an α-approximate portfolio that can be efficiently constructed? A central goal of this thesis is to extend the analysis of classical approximation algorithms to this portfolio setting. Whenever possible, our existence results will be constructive, yielding (typically polynomial-time) algorithms for computing portfolios. However, we will also note examples (see Chapter 5) of gaps between answers to these questions.

Chapter 3 through Chapter 5 discuss portfolios for minimization problems, while Chapter 6 discusses portfolios for maximization problems. Chapter 7 discusses portfolios of algorithms in the Online Convex Optimization (OCO) setting.

Setting. Throughout Chapter 3 to Chapter 6, we will work with a set or *domain* $\mathcal{D}$ of *feasible solutions* to an optimization problem. For example, $\mathcal{D}$ could be the set of all possible combinations of open facilities in a facility location or clustering problem, the set of all schedules in a scheduling problem, or the set of all feasible policies in a reinforcement learning problem. This set is often specified implicitly, and is infinite or exponential in the input size.

We assume that the outcome for stakeholder $i \in [d]$ can be represented as a function $h_i : \mathcal{D} \to \mathbb{R}_{\geq 0}$ of the chosen solution. While simple, this assumption captures many canonical problems. That is, we will be given d *base objective functions* $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ over $\mathcal{D}$. For instance, in the facility location problem, h_i could represent the mean distance to open facilities travelled by the population in the ith group for $i \in [d]$. In a scheduling problem with d machines, h_i could represent the load on machine $i \in [d]$. In a reinforcement learning problem, h_i could represent the reward accumulated by the ith stakeholder.

In Chapter 3 to Chapter 5, these base objectives will represent some notion of *cost* paid

by the stakeholders (as in the facility location and scheduling examples), so that we seek to minimize these. In Chapter 6, these base objectives will represent some notion of *utility* obtained by the stakeholders (as in the reinforcement learning example), so that we seek to maximize these. It is *a priori* unclear which stakeholder or objective should be prioritized, or if a single stakeholder should be prioritized at all.

1.2 Contributions

This section outlines our main contributions in this thesis. In summary, we formally define minimization and maximization portfolios and establish basic properties. We (1) bound the portfolio sizes and (2) study the trade-off between portfolio size and approximation quality for several problems. Through various applications, we show that small portfolios are practically useful. We also design new approximation and online algorithms across a variety of combinatorial and learning problems.[3]

Canonical classes of objectives. We provide general bounds on portfolio sizes for canonical classes of objective functions, for arbitrary feasible sets $\mathcal{D}$ of solutions. These classes of functions model different ways of balancing stakeholder preferences.

1. Conic (positive weighted and linear) combinations of the base objective functions, which represent a natural way to balance the stakeholder preferences in different ways. Under mild technical assumptions, we provide an exponential (in d) upper bound on the portfolio size, and also show that the exponential size is unavoidable.

2. Function classes that *interpolate monotonically* between equity objective $\max_{i\in[d]} h_i$ and efficiency objective $\sum_{i\in[d]} h_i$, such as L_p norms and top-ℓ norms. We give a logarithmic (in d) upper bound on the portfolio size and show that it is tight up to constant factors.

[3] See Chapter 2 for formal problem definitions.

3. Richer classes of objectives that capture more nuanced trade-offs, such as ordered and symmetric monotonic norms. Ordered norms provide a way to assign systematically higher weights to larger costs, interpolating between equity and efficiency in multiple ways. Building on previous work, we discuss upper and lower bounds on portfolio sizes.

4. p-mean objectives that generalize classical welfare measures (utilitarian, egalitarian, Nash) in the maximization setting. We give an upper bound on portfolio sizes.

Canonical optimization problems. As concrete applications of the portfolio framework, we develop portfolios for various canonical optimization problems, including:

1. We introduce the Fair Subsidized Facility Location (FSFL) problem, which generalizes classical uncapacitated facility location and k-clustering problems. FSFL addresses the fundamental tension between profitability and access: it models how a central planner might strategically place pharmacies to ensure coverage in underserved areas, allowing some facilities to operate with losses subsidized by more profitable locations. We provide a new approximation algorithm and portfolios for L_p norms for FSFL. Our approximation algorithm generalizes previously known techniques and introduces a new combinatorial rounding subroutine.

2. For various scheduling and covering problems, we give portfolios for ordered and symmetric monotonic norms with size polylogarithmic in d, improving over the general polynomial in d bound [20]. Our algorithm uses a new primal-dual counting technique, wherein we count objects in a suitable dual space to bound the size of the portfolio in the primal space.

   For the specific problem of balancing machine loads with identical jobs, we provide nearly matching upper and lower bounds on portfolio sizes *for all* approximation factors $\alpha > 4$, thus characterizing the trade-off between portfolio size and approxi-

mation factor for this problem.

3. We apply our results for p-mean functions to Multi-Objective Reinforcement Learning (MORL). Unlike traditional MDPs that give a unique reward for each transition, MORL allows for a vector of rewards for each transition – usually representing the gains of different stakeholders. These multiple reward functions can be aggregated in different ways. We use the p-means of these rewards for various $p \leq 1$ to aggregate these, and each choice of p results in a different optimal policy. We obtain portfolios of policies to (approximately) optimize all possible aggregations (i.e., for all p-means).

Simultaneous approximations. Conversely, we develop an algorithmic framework, called `IterativeOrdering`, which yields new and improved simultaneous approximations (portfolios of size 1) for several combinatorial optimization problems, including job completion time minimization in scheduling and the traveling salesman problems.

Portfolios of algorithms. Moving beyond solution sets, we explore portfolios of algorithms in the Online Convex Optimization (OCO) framework. Unlike the usual portfolio setting where we seek to optimize multiple objectives, there is a single objective to optimize in OCO: the regret of the algorithm. We show that using a portfolio of algorithms can lead to improved regret bounds. Specifically, we construct portfolios of mirror-descent algorithms based on *block norms*, and demonstrate how combining them with multiplicative weights techniques can adaptively achieve low regret rates even when the geometry of the problem is unknown.

Real-world applications. We apply our algorithms and portfolios to various real-world settings, including the following.

1. *Recommending new pharmacies in the US.* We build a web tool that identifies medical deserts in each state in the US – poorer regions that are significantly far away from

their nearest major pharmacy (see Figure 1.1 and Chapter 3). Further, we construct a portfolio of options to open new pharmacies in each state. Different solutions in the portfolio prioritize different regions, but would all potentially lead to a significant reduction in the number of medical deserts.

2. *Healthcare intervention.* We consider a real-world healthcare intervention problem, modeled as a Multi-Objective Reinforcement Learning (MORL) problem described in Chapter 6 and [21]. ARMMAN [22], an NGO based in India, runs large-scale maternal and child mobile health programs. One of their initiatives delivers critical health information via weekly automated voice messages. To enhance engagement, a limited number of beneficiaries receive direct calls from health workers each week, with call assignments determined by a policy. The problem involves prioritizing different socio-demographic groups among the beneficiaries. We build several portfolios of policies that benefit different economic and income groups differently (see Figure 1.2 for one such portfolio).

Table 1.1 summarizes some of these contributions. Together, these results build a unified theory of portfolios across discrete, continuous, and online optimization, highlighting the fundamental trade-offs between solution quality, portfolio size, and algorithmic efficiency. These require developing new algorithmic tools, and we now outline the main challenges.

Table 1.1: A summary of the various settings covered in different chapters of this thesis. For formal problem definitions, see Chapter 2.

Chapter	Class of functions $\mathbf{C}$	Sense	Feasible set $\mathcal{D}$ or problem	Reference
Chapter 3	Conic combinations	Minimization	Arbitrary	Theorem 3.1, Lemma 3.1
	Monotonically interpolating families like L_p norms, top-ℓ norms, and convex combinations		Arbitrary	Theorem 3.2
			Fair Subsidized Facility Location, k-Clustering, and Uncapacitated Facility Location	Theorem 3.4, Corollary 3.1, Theorem 3.5
Chapter 4	Ordered norms and symmetric monotonic norms	Minimization	Arbitrary	Theorem 4.3, Lemma 4.5
			Machine Loads Identical Jobs	Theorem 4.1, Lemma 4.10
			Covering Polyhedron	Theorem 4.2
Chapter 5	Symmetric monotonic norms	Minimization	Completion Times	Theorem 5.1
			Ordered TSP	
			Ordered Set Cover and Ordered Vertex Cover	
			k-Clustering and Uncapaciated Facility Location	Theorem 5.2, Theorem 5.3
Chapter 6	p-Mean functions	Maximization	Arbitrary	Theorem 6.1, Corollary 6.1
			Multi-Objective Reinforcement Learning	

1.3 Algorithmic Challenges

We highlight some algorithmic challenges that arise in constructing small portfolios.

Beyond counting functions. A natural approach to obtaining portfolios is to count the number of distinct objectives in the class $\mathbf{C}$ of objectives, up to α-approximation, and then optimize for each objective. This yields small portfolios in some cases: for example, $O(1)$-approximate portfolios of size $O(\log d)$ for L_p norms [23], and $\mathrm{poly}(d)$-sized portfolios for ordered norms [20]. However, this strategy ignores the feasible set $\mathcal{D}$ entirely, and can greatly overestimate the necessary portfolio size. Further, the number of distinct functions in $\mathbf{C}$ may be infinite (as with p-means for any constant α), making the approach inapplicable. Obtaining optimal bounds, therefore, often requires reasoning about both $\mathbf{C}$ and $\mathcal{D}$ simultaneously.

Loss of structure beyond simultaneous approximations. For portfolios of size 1, powerful structural results exist: e.g., [9] showed that if a solution is an α-approximation for all top-ℓ norms, then it is also an α-approximation for the broader class of symmetric monotonic norms. Such results collapse large families of objectives into smaller, more manageable ones. Unfortunately, this connection breaks down once we move beyond size-1 portfolios. Thus, new techniques are needed to handle richer classes of objectives where these results no longer apply.

Designing lower bounds. Given an optimization problem, proving lower bounds on the size of the smallest α-approximate portfolio involves careful constructions that appropriately trade off various objectives.

Problem-specific barriers. In addition to these general challenges, individual problems such as FSFL, scheduling, reinforcement learning, online learning, etc, have their own technical challenges. We discuss these in detail in the respective chapters.

1.4 Overview

Here, we present an overview of the various chapters and results in the thesis.

Chapter 3. We begin with a discussion of portfolios for conic combinations and L_p norms of the base objectives, and apply these results to a novel facility location problem.

First, under mild technical assumptions, we first give portfolios for the class $\mathbf{C}$ of weighted linear or conic combinations of given base objective functions. However, the size of this portfolio has an exponential dependence on the number d of base objectives, and we show that this dependence is necessary in the worst case. This is not unexpected in such a general setting: each stakeholder $i \in [d]$ could have widely varying preferences, and balancing them in various ways forces us to consider a large number of solutions that satisfy these varied combinations.

One way to meaningfully reduce portfolio sizes is to narrow down the class $\mathbf{C}$ of objectives. We do this by instead looking at objectives that interpolate between an *efficiency* and an *equity* objective. These two fundamentally important objectives are defined next.

When base objective h_i represents the cost paid by stakeholder $i \in [d]$, a natural objective is to minimize the sum of costs paid by all stakeholders. This is the classically studied *efficiency* or *utilitarian* objective, where we seek to minimize the sum of costs borne by all individuals, or equivalently, the L_1 norm of $\mathbf{h} := (h_1, \ldots, h_d)$ defined as $\|\mathbf{h}(x)\|_1 := \sum_{i \in [d]} h_i(x)$ for each solution $x \in \mathcal{D}$. At the other extreme, we have the *equity* or the *egalitairian* objective, which is the *maximum* cost borne by any stakeholder, or equivalently, the L_∞ norm $\|\mathbf{h}(x)\|_\infty := \max_{i \in [d]} h_i(x)$ of $\mathbf{h}$.

Next, we consider classes $\mathbf{C}$ of objectives that *interpolate monotonically* between these efficiency and equity objectives. Examples of such classes include convex combinations of the efficiency and equity objectives, L_p norms of the base objectives, and top-ℓ norms of the base objectives. Given an oracle to find β-approximations to each objective $h \in \mathbf{C}$, we obtain portfolios for $\mathbf{C}$ with size logarithmic in d:

Theorem 3.2. *Let $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ be d nonnegative functions on feasible set $\mathcal{D}$, and denote $\mathbf{h} = (h_1, \ldots, h_d)$. Let $\mathbf{C}$ denote a class of objectives that interpolate monotonically between $\|\mathbf{h}\|_1$ and $\|\mathbf{h}\|_\infty$, and let $\mathcal{O}_\beta$ be an oracle that gives a β-approximate solution to $\arg\min_{x \in \mathcal{D}} h(x)$ for any $h \in \mathbf{C}$. There exists an algorithm that given any $\varepsilon \in (0, 1]$, finds a $\beta(1+\varepsilon)$-approximate portfolio of size at most $O\left(\frac{\log \beta d}{\varepsilon}\right)$ in $\text{poly}(d, \frac{1}{\varepsilon})$ number of oracle calls to $\mathcal{O}_\beta$.*

We also show that this portfolio size is tight up to constant factors for L_p norms. That is, a logarithmic number of solutions is sometimes necessary for L_p norms. In particular, simultaneous approximations may not exist.

To illustrate this result on a concrete setting, we introduce the *Fair Subsidized Facility Location* (FSFL) problem. This problem generalizes two well-studied problems in computer science and operations research – the uncapacitated facility location and k-clustering problems. In FSFL, each client generates revenue for the open facility to which it is assigned. Some of the profits from the profit-making facilities can be diverted to the loss-making locations to sustain them, while improving access throughout all neighborhoods. We obtain a new approximation algorithm for FSFL for a large class of convex objectives, which then leads to portfolios for FSFL.

Chapter 4. Next, we study portfolios for ordered and symmetric monotonic norms. Since the objectives $h_1, \ldots, h_d$ represent some notion of cost borne by various stakeholders, it is reasonable to consider some 'size' or a *norm* of the vector $\mathbf{h}(x)$ of costs as the objective, with different norms giving different objectives. Chapter 3 deals with designing portfolios for any $\mathbf{C}$ of norms that interpolate monotonically between the L_1 norm and L_∞ norm of $\mathbf{h}$, including L_p norms, top-ℓ norms, and convex combinations of efficiency and equity. However, many natural classes of norms do not interpolate monotonically between $\|\mathbf{h}\|_1$ and $\|\mathbf{h}\|_\infty$. Two such classes are ordered norms and symmetric monotonic norms.

Given d positive *weights*, the ordered norm of the vector $(h_1(x), \ldots, h_d(x))$ is obtained

by weighting the highest cost among $h_1(x), \ldots, h_d(x)$ with the highest weight, the second highest with the second highest weight, and so on. Ordered norms generalize top-ℓ norms. Unlike top-ℓ norms, they do not always admit logarithmic-sized portfolios.

Symmetric monotonic norms are an even more general class of norms and include all norms that are (1) invariant to the permutation of coordinates and (2) monotone in each coordinate. Since each coordinate represents some notion of cost borne, monotonicity is a natural requirement. Symmetry means that the cost on one stakeholder is not treated differently from the cost on another stakeholder.[4]

The sizes of $O(1)$-approximate portfolios can be polynomial in d in the worst case. We ask if we can reduce this size to polylogarithmic in d for some optimization problems. First, we study the MACHINELOADSIDENTICALJOBS problem, where n identical jobs must be scheduled on d unidentical machines. A schedule consists of an assignment of jobs to machines. Given a schedule, the cost h_i on machine $i \in [d]$ is the total load on machine i. The efficiency objective corresponds to minimizing the total load on all machines. The equity objective corresponds to minimizing the maximum load on any machine and is known classically as the makespan minimization problem. We give upper and lower bounds on the size of α-approximate portfolios for this problem:

Theorem 4.1. *There is a polynomial-time algorithm that, given any instance of the* MACHINELOADSIDENTICALJOBS *problem with d machines and any $\alpha > 4$, finds a portfolio X of size*

$$|X| = O\left(\frac{\log d}{\log(\alpha/4)}\right)$$

that is (i) α-approximate for ordered norms and (ii) $O(\alpha \log d)$-approximate for symmetric monotonic norms. Further, for all $\alpha > 1$, there exists a family of instances of MACHINELOADSIDENTICALJOBS *for which the size of any α-approximate portfolio for ordered norms is $\Omega\left(\frac{\log d}{\log \alpha + \log\log d}\right)$.*

[4]Up to the design of the cost functions $h_i, i \in [d]$.

Next, we generalize this to portfolios for arbitrary COVERINGPOLYHEDRON $\{x \in \mathbb{R}^d_{\geq 0} : Ax \geq b\}$ with $A \geq 0$ where the cost $h_i(x) = x_i$. When the constraint matrix A has a constant number of constraints, we show that the covering polyhedron admits a portfolio of size $\mathrm{polylog}(d)$ that is $O(1)$-approximate for ordered norms and $O(\log d)$-approximate for symmetric monotonic norms:

Theorem 4.2. *For* COVERINGPOLYHEDRON *in d dimensions and r constraints, for any $\varepsilon \in (0, 1]$, there is a portfolio X of size*

$$|X| = O\big(\log(d/\varepsilon)/\varepsilon\big)^{3r^2-2r},$$

which is (i) $(1+\varepsilon)$-approximate for ordered norms, and (ii) $O(\log d)$-approximate for symmetric monotonic norms. There exists an algorithm to find this portfolio with running time that is polynomial in d and $(\log(d)/\varepsilon)^{r^2}$.

Chapter 5. In each of our results thus far, simultaneous approximations (provably) do not exist: all portfolio sizes are larger than 1. In Chapter 5, building on previous work, we introduce a general algorithmic framework called `IterativeOrdering` to obtain new or improved simultaneous approximations for various combinatorial optimization problems.

In particular, we give the first constant-factor simultaneous approximation for a scheduling problem (called COMPLETIONTIMES) where we seek to minimize different norms of the completion times of jobs[5], and show the existence of an improved simultaneous approximation factor for generalizations of the traveling salesman problem (denoted ORDEREDTSP) and the set cover problem (denoted ORDEREDSETCOVER). More generally, we define a class of problems called γ-COMPOSABLE problems, which generalizes each of the above problems, and obtain simultaneous approximations for it:

[5]Contrast this with the result in Chapter 4 where we discuss scheduling problems to minimize machine loads instead.

Theorem 5.1. *1. For any γ-*COMPOSABLE *problem, there always exists a simultaneous $(\sqrt{\gamma}+1)^2$-approximation.*

2. For ORDEREDSETCOVER*,* ORDEREDVERTEXCOVER*, and* COMPLETIONTIMES*, there always exists a simultaneous 4-approximation.*

3. For ORDEREDTSP*, there always exists a simultaneous $(3+2\sqrt{2})$-approximation.*

4. For COMPLETIONTIMES*, a simultaneous 8-approximation can be found in polynomial-time.*

Chapter 6. We pivot to maximization problems next and discuss portfolios for p-mean functions in Chapter 6. These are natural analogues of L_p norms for maximization. In particular, we discuss portfolios for reinforcement learning settings where, instead of a single reward function, d stakeholders each have their own reward functions.

For $p \leq 1$, the p-mean of a vector $z \in \mathbb{R}^d_{>0}$ is defined as the pth root of the mean of the pth power of coordinates of z, i.e., $\left(\frac{1}{d}\sum_{i\in[d]} z_i^p\right)^{1/p}$. p-Means generalize the arithmetic mean or utilitarian welfare function ($p=1$), the minimum or the egalitarian welfare function ($p=-\infty$), and the geometric mean or the Nash welfare function ($p=0$).

Assuming that the base objectives are bounded $1 \leq h_i(x) \leq \kappa$ for all $x \in \mathcal{D}$ and $i \in [d]$, we obtain an α-approximate portfolio of size $O\left(\frac{\ln \kappa}{\ln(1/\alpha)}\right)$ for all p-mean functions. Our algorithm assumes an oracle for finding the $x^* \in \mathcal{D}$ that maximizes the p-mean of $\mathbf{h}(x)$, for any given p. Our results also hold in the setting when the base functions h_i are *random* – a useful generalization for the reinforcement learning setting described below.

Theorem 6.1. *Given feasible set $\mathcal{D}$, (possibly random) base objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$ with condition number κ, and desired approximation factor $\alpha \in (0,1)$, Algorithm 7 (p-*`MeanPortfolio`*) returns an α-approximate portfolio of policies X for the class $\mathbf{C}$ of all p-mean functions for $p \leq 1$. Further,*

1. *The portfolio size*

$$|X| = O\left(\frac{\ln \kappa}{\ln(1/\alpha)}\right),$$

2. *The number of oracle calls by the algorithm is upper-bounded by*

$$\widetilde{O}\left(\frac{(\ln \kappa)^2 \ln \ln d}{\ln(1/\alpha)}\right),$$

where $\widetilde{O}$ hides all lower order terms.

As an application, we consider a Multi-Objective Reinforcement Learning (MORL) setting where we are given an MDP with d reward functions (each bounded between 1 and κ) that represent the preferences of d stakeholders. These rewards can be aggregated in different ways using different p-mean functions. Using the above result, we obtain an α-approximate portfolio of $O\left(\frac{\ln \kappa}{\ln(1/\alpha)}\right)$ policies for this MORL setting, with $\widetilde{O}\left(\frac{(\ln \kappa)^2 \ln \ln d}{\ln(1/\alpha)}\right)$ oracle calls.

Our algorithm assumes that we can find the optimal policy for any given p-mean aggregation of the reward functions. This *oracle call* is expensive in practice, and therefore, any algorithm is practical only if it uses a small number of oracle calls.

Additionally, we provide a lightweight heuristic algorithm to complement our main algorithm, which makes fewer oracle calls but achieves similar performance in practice. We compare both our main algorithm and the lightweight heuristic in three different settings and show that they significantly outperform natural baselines.

Chapter 7. Previous chapters construct portfolios of *solutions*. In Chapter 7, we ask a different question: can we construct portfolios of *algorithms* to solve optimization problems, and improve performance over standard algorithms?

We partially answer this question in the setting of Online Convex Optimization (OCO). In OCO, a learner repeatedly makes decisions from a convex set while an adversary reveals convex loss functions, and the goal is to minimize regret against the best fixed decision

in hindsight. Online Mirror Descent (OMD) is a family of algorithms for OCO, and includes standard algorithms such as Online Projected Gradient Descent (OPGD) and Online Exponentiated Gradient (OEG).[6]

A straightforward approach to improving regret over OPGD and OEG is to use a different OMD algorithm to better suit the problem geometry. First, building on previous work, we introduce a portfolio of OMD algorithms corresponding to different *block norms* (see Definition 2.1) that smoothly interpolate between OPGD and OEG-like geometries. We show that using an appropriate *block norm* for the OMD algorithm leads to significant regret improvement over OPGD and OEG-like algorithms in certain regimes. Specifically, (1) we give OCO settings where the convex body is the probability simplex, where this regret improvement is $\simeq \sqrt{\ln d}$ in dimension d, and (2) OCO settings on another polytope where this regret improvement is polynomial in d. To the best of our knowledge, this is the first polynomial in dimension improvement in regret over both OPGD and OEG-like algorithms. We complement our theoretical guarantees with numerical simulations over the probability simplex.

However, since the convex loss functions are unknown in advance, the choice of the optimal block norm cannot be ascertained in advance. We combine our portfolio of block norms with a multiplicative-weight update algorithm that automatically chooses the optimal algorithm in the portfolio, while guaranteeing that the regret is at most factor $O(\sqrt{\ln \ln d})$ times the regret of the best block norm in hindsight.

Formally, for an OCO setting on convex body $\mathcal{K}$, convex loss functions $f^{(1)}, \ldots, f^{(T)}$ over a horizon of T steps, we show the following:

Theorem 7.7 (Combining block norms)**.** *Consider an OCO setting with convex body $\mathcal{K}$ that lies within the L_1 norm ball in $\mathbb{R}^d$. Suppose the number of time steps $T \geq 4 \ln \ln d$, and suppose the differential in loss functions $\max_{x,z \in \mathcal{K}, t \in [T]} f^{(t)}(x) - f^{(t)}(z) = 1$. Then, given the Euclidean diameter $D_{\text{euc}} = \max_{x,z \in \mathcal{K}} \|x - z\|_2$ and Euclidean gradient norm*

[6]Over the probability simplex.

bound $G_{\text{euc}} = \max_{x \in \mathcal{K}, t \in [T]} \|\nabla f^{(t)}(x)\|_2$, `MirrorWeights` *(Algorithm 12) with block norms achieves regret at most*

$$\texttt{regret}(T) = O(\sqrt{\ln \ln d}) \cdot \min_{n \in \{2^0, \ldots, 2^{\log_2 d}\}} D_n G_n \sqrt{T},$$

where D_n *is the diameter under the corresponding Bregman divergence* B_{h_n} *and* G_n *is the gradient norm upper bound in the dual norm to* n*th block norm.*

Here, $n \in \{2^0, \ldots, 2^{\log_2 d}\}$ indexes the block norms, and $O(D_n G_n \sqrt{T})$ is the standard upper bound on the regret of OMD with the nth block norm.

1.5 Related Work

Having outlined the contributions, we now situate this work within the broader literature.

A foundational strand of work in algorithms studies approximation guarantees for combinatorial optimization problems, asking when a single solution can simultaneously approximate multiple objectives (i.e., portfolios of size 1). The earliest results go as far back as 1975; [24] study the scheduling problem of minimizing loads on identical machines and show that Graham's [25] greedy algorithm is a 1.5-approximation for all L_p norms. Kumar and Kleinberg [8] studied simultaneous approximations for all symmetric monotonic norms for clustering, scheduling, and flow problems. [23] identify that several combinatorial problems admit a similar algorithmic technique to obtain simultaneous approximations; we formalize and generalize their technique as `IterativeOrdering` in Chapter 5. [9] in particular study simultaneous approximations for various problems and establish a fundamental result that shows that simultaneous α-approximations for top-ℓ norms are simultaneous α-approximations for other classes of norms like L_p norms. When portfolio size is larger than 1, portfolios for top-ℓ norms may *not* be portfolios for L_p norms and other classes (see Chapter 3 and Chapter 4 for examples), thus requiring new techniques to handle such cases. The list of works that give simultaneous approximations for specific problems

is too long to fit here; we mention some examples: [26] and [27] give simultaneous approximations for the traveling salesman problem, while [28] discuss it for scheduling.

To the best of our knowledge, we are the first to explicitly study portfolios for arbitrary infinite classes of objectives. However, similar notions are implicit in the works of [8, 9, 23, 20]. [23] in particular study L_p norms and their results imply logarithmic-sized portfolios for L_p norms. Their technique, however, crucially uses the structure of L_p norms and does not generalize to other classes that monotonically interpolate between L_1 and L_∞ norms. Our technique for portfolio upper bounds for such families closely resembles the technique of [9], who use it to get portfolio-like constructions for top-ℓ norms. [20] essentially construct polynomial-sized portfolios for ordered norms.

Another well-known related concept is the notion of *Pareto frontier approximations* [29]. Pareto frontier of objectives $h_1, \ldots, h_d$ on feasible set $\mathcal{D}$ is the set of all points $\mathbf{h}(x)$ for $x \in \mathcal{D}$ where not all of the function values can be improved, i.e., for all $x' \in \mathcal{D}, h_i(x') > h_i(x)$ for some $i \in [d]$. The $(1+\varepsilon)$-approximate Pareto frontier relaxes this constraint by factor $(1+\varepsilon)$. [29] give algorithms to compute Pareto frontiers for several problems, and a long line of works builds on these results to give Pareto frontier approximations for specific settings [30, 31, 18]. The notions of Pareto frontiers and portfolios coincide in the special case of size equal to 1, i.e, if some feasible solution $x^* \in \mathcal{D}$ is simultaneously optimal for all $h_i(\cdot), i \in [d]$, then the Pareto frontier reduces to the single point $\{\mathbf{h}(x^*)\}$. This also holds true when $x^* \in \mathcal{D}$ is *simultaneouly α-approximate* for all $h_i(\cdot)$ or that $h_i(x^*) \leq \min_{x \in \mathcal{D}} h_i(x)$ for all i, in which case the single point $\{\mathbf{h}(x^*)\}$ coordinate-wise α-approximates the Pareto frontier.

The trade-off between fairness and accuracy has been well-studied in both machine learning and combinatorial optimization communities. Most of these works optimize a fixed trade-off between fairness and accuracy, but do not specify how the trade-off is chosen. [32] introduce a regularization term added to the objective function to penalize any dependence between the model's predictions and a sensitive attribute. [33] formulate fair-

ness as a multi-objective learning problem of finding a data representation that balances the twin goals of preserving predictive information and obfuscating information about the protected group membership. [10] and [11] establish negative results showing that different measures of fairness cannot always hold simultaneously. In combinatorial optimization, the trade-off between fairness and accuracy is studied by optimizing for a *fixed* non-standard objective, such as an L_p or top-ℓ norm. The list is too long to fit here; we give some representative examples. [23, 34, 27, 35, 36] study L_p norm objectives, [9, 37] study top-ℓ norm objectives, [20, 38] study ordered norm and symmetric monotonic norm objectives, and [39, 40, 41, 42] study p-means objectives.

Next, we discuss the OCO problem from Chapter 7. In this setting, it is known that algorithms other than OPGD and OEG can achieve asymptotically better regret rates in certain settings, motivating a portfolio of OMD algorithms that is agnostic to problem geometry. For example, [43] give logarithmic in dimension improvement in regret. However, to the best of our knowledge, we are the first to show polynomial in dimension improvement in regret over both OPGD and OEG (or an equivalent proxy). Once we have a portfolio of online algorithms, the Multiplicative Weight Update (MWU) method [44] gives a natural way to combine them. Existing approaches use MWU to make specific OMD algorithms agnostic to parameters [45, 46, 47], and our approach of combining different OMD algorithms is a natural extension.

Notions similar to portfolios – where allowing more than a single solution is desired – appear in other settings as well. For example, [48] study combinatorial problems with stochastic objectives and seeks a portfolio of solutions to optimize the expected value of the best solution. As another example, in Online Convex Optimization in the *bandit setting* – i.e., when the player can only observe the loss function partially – being able to evaluate the loss function at multiple points can significantly reduce regret (see, for example, [49]).

In voting theory, *Condorcet winning sets* provide a natural analogue of the solution portfolios studied in this thesis: in ranked-choice voting (i.e., when each voter provides a

strict ranking over the m candidates), rather than insisting on a single universally optimal alternative (called a *Condorcet winner*), one seeks a small committee of candidates such that a majority of voters do not prefer a candidate to each candidate in the committee. [50] showed that such a committee of $O(\log m)$ candidates always exists in the worst case. More recently, [51] proved that a *constant*-size set suffices: specifically, every election admits a Condorcet winning set of at most six candidates, improving the long-standing logarithmic bound. These examples parallel the central theme of this thesis: insisting on a single solution is often unnecessarily restrictive, whereas allowing small portfolios can dramatically expand what guarantees are achievable.

Finally, in many settings, the objective function $f \in \mathbf{C}$ itself is unknown, and one seeks to infer it based on querying the feasible solutions (e.g., inverse optimization [52], preference elicitation [53] etc).

CHAPTER 2

BACKGROUND

In this chapter, we develop some notation, cover some background on approximations and norms, and define the various combinatorial problems that we study in this thesis.

2.1 Notation and Preliminaries

Minimization and maximization problems. An optimization problem is specified by a set or *domain* of feasible solutions $\mathcal{D}$ and an *objective function* $h : \mathcal{D} \to \mathbb{R}$. In a minimization problem, the goal is to find $x^* \in \mathcal{D}$ with $h(x^*) = \min_{y \in \mathcal{D}} h(y)$. In a maximization problem, the goal is to find $x^* \in \mathcal{D}$ with $h(x^*) = \max_{y \in \mathcal{D}} h(y)$.

Approximations. For a minimization problem with feasible set $\mathcal{D}$ and nonnegative objective $h : \mathcal{D} \to \mathbb{R}_{\geq 0}$, we say that $x \in \mathcal{D}$ is an α-approximate solution if

$$h(x) \leq \alpha \cdot \min_{y \in \mathcal{D}} h(y),$$

for some factor $\alpha \geq 1$. For maximization problems, x is an α-approximate solution if

$$h(x) \geq \alpha \cdot \max_{y \in \mathcal{D}} h(y),$$

for some $\alpha \in (0, 1]$.

Portfolios. While the above definitions concern a single objective function, the central concept of this thesis is the notion of *portfolios*, which generalizes approximation from one objective to an entire class of objectives. We formally introduced minimization and maximization portfolios in the Introduction (Definition 1.1 and Definition 1.2), and will build

on those throughout the thesis. When X is a 1-approximate portfolio for some feasible set $\mathcal{D}$ and class $\mathbf{C}$ of functions, we say that X is an *optimal* portfolio. α-approximate portfolio $X = \{x\}$ is singleton, we say that x is a simultaneous α-approximation.

Norms. A recurring motif in this thesis is the study of optimization under various ways of measuring fairness. When a vector $x \in \mathbb{R}^d_{\geq 0}$ represents *costs* on various stakeholders, the natural way to measure whether x is fair or not is to measure its 'size'. Next, we define the various norms of a vector considered in this work that measure this size.

Given vector $x \in \mathbb{R}^d_{\geq 0}$, we denote by $x^{\downarrow} \in \mathbb{R}^d_{\geq 0}$ the vector formed by sorting the coordinates of x in decreasing order. Given a subset $S \subseteq [d]$ of coordinates, we denote $x_S \in \mathbb{R}^S$ to be the restriction of x to coordinates in S.

Definition 2.1 (Norms)**.** *Given a vector $x \in \mathbb{R}^d$,*

1. *Given $p \geq 1$, the L_p* norm *of x, denoted $\|x\|_p$ is defined as*

$$\|x\|_p := \left(\sum_{i \in [d]} |x_i|^p \right)^{1/p}. \tag{2.1}$$

2. *Given $\ell \in [d]$, the* top-ℓ norm *of x, denoted $\|x\|_{(\mathbf{1}_\ell)}$, is defined as the sum of the ℓ highest coordinates of x by absolute value:*

$$\|x\|_{(\mathbf{1}_\ell)} := \sum_{i \in [\ell]} |x|^{\downarrow}_i. \tag{2.2}$$

3. *Given a nonzero vector $w \in \mathbb{R}^d_{\geq 0}$ with $w_1 \geq w_2 \geq \ldots \geq w_d \geq 0$, the corresponding* ordered norm *of x, denoted $\|x\|_{(w)}$, is defined as*

$$\|x\|_{(w)} := \sum_{i \in [d]} w_i |x|^{\downarrow}_i. \tag{2.3}$$

w is called the weight vector *associated with the ordered norm.*

4. *A norm $\|\cdot\| : \mathbb{R}^d \to \mathbb{R}$ is called a* symmetric monotonic norm *if it is (1) invariant to the permutation of coordinates, and (2) monotone in each coordinate.*

5. *Given a partition $\mathcal{B} = \{B_1, \ldots, B_n\}$ of the d coordinates into n* blocks*, the* block norm *of x, denoted $\|x\|_{[\mathcal{B}]}$, is defined as the sum of the L_2 norms of each block:*

$$\|x\|_{[\mathcal{B}]} := \sum_{j\in[n]} \|x_{B_j}\|_2 = \sum_{j\in[n]} \sqrt{\sum_{i\in B_j} x_i^2} \tag{2.4}$$

Of crucial importance will be the *duals* of the above norms, defined as follows:

Definition 2.2 (Dual Norm)**.** *Given a norm $\|\cdot\|$ on $\mathbb{R}^d$, its* dual norm $\|\cdot\|_*$ *is defined for any $y \in \mathbb{R}^d$ as*

$$\|y\|^* := \sup_{\|x\|\le 1} \langle x, y\rangle.$$

Majorization. For nonnegative vectors $x, y \in \mathbb{R}^d_{\ge 0}$, we say that y majorizes x or $x \preceq y$ if $\|x\|_{(\mathbf{1}_\ell)} \le \|y\|_{(\mathbf{1}_\ell)}$ for all $\ell \in [d]$. The following lemma shows that majorization implies monotonicity for any symmetric monotonic norm.

Lemma 2.1 ([54])**.** *If $x \preceq y$, then $\|x\| \le \|y\|$ for any symmetric monotonic norm $\|\cdot\|$.*

As [9] show, the above lemma implies that for minimization problems, simultaneous approximations for top-ℓ norms are simultaneous approximations for all symmetric monotonic norms, despite the latter being a much bigger class of norms. Given some feasible set $\mathcal{D}$ and base functions $h_1, \ldots, h_d$, consider x^* that is simultaneously α-approximate for all top-ℓ norms for some $\alpha \ge 1$. Then $\|\mathbf{h}(x^*)\|_{(\mathbf{1}_\ell)} \le \alpha\|\mathbf{h}(y)\|_{(\mathbf{1}_\ell)}$ for all $\ell \in [d]$ and $y \in \mathcal{D}$, i.e., that $\mathbf{h}(x^*) \preceq \alpha\mathbf{h}(y)$ for all $y \in \mathcal{D}$. As an immediate consequence, we get the following result from [9] that we state in a modified form:

Lemma 2.2 ([9], Theorem 2.3)**.** *For any $\mathcal{D}$ and base functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\ge 0}$, if x^* is a simultaneous α-approximation for top-ℓ norms, then x^* is a simultaneous α-approximation for all symmetric monotonic norms.*

It is natural to ask if this result for simultaneous approximations – that is, portfolios of size 1 – extends to portfolios of size greater than 1. That is, if $X \subseteq \mathcal{D}$ is an α-approximate (minimization) portfolio for top-ℓ norms, is it also an α-approximate portfolio for all symmetric monotonic norms, regardless of what $|X|$ is? The answer is no, as the following example shows:

Example 2.1. *Consider set $\mathcal{D} = \{x, y, z\} \in \mathbb{R}^d$ of three feasible vectors $x = (\sqrt{d}, 0, \dots 0)$, $y = (1, \dots, 1)$, and $z = d^{1/3}\left(1, \frac{1}{\sqrt{2}}, \dots, \frac{1}{\sqrt{d}}\right)$, and $h_i(x) = x_i$ for all $i \in [d]$, i.e., $\mathbf{h}(x) = x$. Then, given a top-ℓ norm,*

$$\|x\|_{(\mathbf{1}_\ell)} = \sqrt{d}, \quad \|y\|_{(\mathbf{1}_\ell)} = \ell, \quad \|z\|_{(\mathbf{1}_\ell)} = d^{1/3} \sum_{i \in [\ell]} \frac{1}{\sqrt{i}} = \Theta(d^{1/3}\sqrt{\ell}).$$

For each top-ℓ norm, either x or y is optimal, i.e., $\{x, y\}$ is an optimal portfolio for top-ℓ norms. However, consider the ordered norm for weight vector $w = \left(1, \frac{1}{\sqrt{2}}, \dots, \frac{1}{\sqrt{d}}\right)$:

$$\|x\|_{(w)} = \sqrt{d}, \quad \|y\|_{(w)} = \Theta(\sqrt{d}), \quad \|z\|_{(w)} = d^{1/3} \sum_{i \in [d]} \frac{1}{i} = \Theta(d^{1/3} \log d).$$

Then both x and y are $\Omega\left(\frac{d^{1/6}}{\log d}\right)$-approximations for $\|\cdot\|_{(w)}$, i.e., $\{x, y\}$ is at best a $\mathrm{poly}(d)$-approximate portfolio for ordered norms. □

p-Mean Functions. For maximization problems, we will consider the following analogues to L_p norms, called the p-mean functions:

Definition 2.3 (p-Mean function)**.** *Given a vector $x \in \mathbb{R}^d_{>0}$ and $p \in [-\infty, 1]$, the p-mean of x, denoted $M_p(x)$, is defined as*

$$M_p(x) = \begin{cases} \min_{i \in [d]} x_i & \text{if } p = -\infty, \\ \left(\frac{1}{d} \sum_{i \in [d]} x_i^p\right)^{1/p} & \text{if } p \notin \{-\infty, 0\}, \\ \left(\prod_{i \in [d]} x_i\right)^{1/d} & \text{if } p = 0. \end{cases} \tag{2.5}$$

2.2 Glossary of Problems

This section is intended as a reference glossary. It collects the formal definitions of the combinatorial problems studied in this thesis. Readers may wish to skip it on a first pass and return to it as needed when encountering these problems in later chapters.

2.2.1 Facility Location Problems

First, we define the various facility location problems studied in this thesis, including the well-known UNCAPACITATEDFACILITYLOCATION and k-CLUSTERING problems, as well as the Fair Subsidized Facility Location (FSFL) problem that we introduce. In each of the three problems, we are given a metric space $(C \cup F, \texttt{dist})$ with set C of clients, set F of potential open facilities, and distances $\texttt{dist}$ that form a metric. The goal is to open a subset $F' \subseteq F$ of facilities and assign $\Pi : C \to F'$ the clients to open facilities; this solution is denoted (F', Π). However, there are additional inputs and restrictions in each problem, which we specify next:

UNCAPACITATEDFACILITYLOCATION. The input consists of (1) a metric space $(C \cup F, \texttt{dist})$ and (2) nonnegative facility opening costs $c : F \to \mathbb{R}_{\geq 0}$. For a subset $F' \subseteq F$ of open facilities and an assignment $\Pi : C \to F'$ of clients to open facilities, the classical objective is to minimize the sum of the costs of open facilities and the distances of clients to their assigned facilities, i.e., $\sum_{f \in F'} c_f + \sum_{j \in C} \texttt{dist}_{j,\Pi(j)}$. We will study various other objectives; see Chapter 3 and Chapter 5.

k-CLUSTERING. The input consists of (1) a metric space $(C \cup F, \texttt{dist})$ and (2) an integer $1 \leq k \leq |F|$. Solution (F', Π) is feasible if and only if at most $|F'| \leq k$ facilities are open. Classical objectives include (a) k-median, which is to minimize the L_1 norm of the distances of clients to open facilities, i.e., $\sum_{j \in C} \texttt{dist}_{j,\Pi(j)}$, (b) k-means, which is to minimize the L_2 norm of client distances, and (c) k-center, which is to minimize the L_∞ norm of client distances. We will study various other objectives.

FSFL. The input consists of (1) a metric space $(C \cup F, \texttt{dist})$, (2) nonnegative facility opening or operating costs $c : F \to \mathbb{R}_{\geq 0}$, (3) client revenues $r : C \to \mathbb{R}_{\geq 0}$, and (4) a *subsidy* parameter $\delta > 0$. An open facility $f \in F'$ is *unprofitable* if the total revenue $\sum_{j \in C : \Pi(j) = f'} r_j$ of clients assigned to it is less than its operating cost c_f, and the loss of an unprofitable facility is defined as $c_f - \sum_{j : \Pi(j) = f'} r_j$. Solution (F', Π) is feasible if and only if the total loss of unprofitable facilities is at most a fraction δ of the total client revenue $\sum_{j \in C} r_j$. We study various objectives for this problem; see Chapter 3.

2.2.2 Scheduling and Covering Polyhedra

Next, we define various scheduling problems and covering polyhedra. In a scheduling problem, we are given d *machines*, n *jobs*, and processing times $p_{i,j} > 0$ for each job $j \in [n]$ on each machine $i \in [d]$. A feasible solution consists of an assignment of each job to some machine, and an ordering of the jobs assigned to each machine.

COMPLETIONTIMES. The goal of this problem is to minimize (some function of) the *completion times* of jobs. For job $j \in [n]$ assigned to machine $i \in [d]$ in a feasible solution, the job's completion time is defined as the sum of the processing times of jobs assigned to i before j (including j).

MACHINELOADSIDENTICALJOBS. In this problem, the goal is to minimize (some function of) the loads on various machines, assuming that the n jobs have the same processing time for any given machine. Since the n jobs are identical, we refer to $p_i, i \in [d]$ as the processing time for each machine. The load on a machine is defined as the sum of the processing times of jobs assigned to it. Note that the order of jobs assigned to a machine does not matter in this case.

COVERINGPOLYHEDRON. In this problem, the feasible set $\mathcal{D} = \{x \in \mathbb{R}^d : Ax \geq b, x \geq 0\}$ where $A_{\geq 0} \in \mathbb{R}^{r \times d}$ is a nonnegative matrix and $b \in \mathbb{R}^r_{\geq 0}$, $\mathcal{D}$ is commonly known as a *covering polyhedron*. The goal is to minimize various functions of $x \in \mathcal{D}$.

2.2.3 Other Combinatorial Problems

ORDEREDSETCOVER. The input consists of a ground set of n elements and m subsets $S_1, \ldots, S_m$ of the ground set. The output is an order on the subsets; each output induces a vector of cover times of elements in the ground set, defined for an element as the position of the first set in the order containing it. Given a norm $\|\cdot\|$ on $\mathbb{R}^n$, the objective is to minimize the norm of cover times. Special cases include classical Set Cover (for the L_∞ norm) [55], and Min-Sum Set Cover or MSSC (for the L_1 norm) [56].

ORDEREDVERTEXCOVER. This is a special case of ORDEREDSETCOVER where the ground set corresponds to edges of an undirected graph and the subsets correspond to vertices of the graph. Special cases include classical Vertex Cover (for the L_∞ norm), and Min-Sum Vertex Cover or MSVC (for the L_1 norm) [56].

ORDEREDTSP. The input consists of a metric space on n points or vertices V and a starting vertex $v_0 \in V$. The output is a Hamiltonian tour of the vertices starting at v_0; each tour induces a vector of visit times of the vertices, defined for a vertex as its distance from v_0 along the tour. Given a norm $\|\cdot\|$ on $\mathbb{R}^n$, the objective is to minimize the norm of visit times. Special cases include the Traveling Salesman Problem or TSP (for the L_∞ norm) [57], the Traveling Repairman Problem (for the L_1 norm) [58], and the Traveling Firefighter Problem (for the L_2 norm) [27].

CHAPTER 3
CONIC COMBINATIONS, MONOTONIC INTERPOLATIONS, AND FACILITY LOCATION

3.1 Introduction

In this chapter, given a finite set of base objective functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ (e.g., access costs for groups 1 to d), we consider minimization portfolios for the following canonical classes of functions:

(i) *Conic combinations of base functions*: $\mathbf{C}_1 = \left\{ \sum_{i \in [d]} \lambda_i h_i : \lambda \geq 0 \right\}$. Each conic combination $g_\lambda := \sum_{i \in [d]} \lambda_i h_i$ is another objective on $\mathcal{D}$, and represents a unique way to balance the base objectives. Different conic combinations can have different minimizers in $\mathcal{D}$.

(ii) *Interpolating functions*: $\mathbf{C}_2 = \{g_\lambda : \lambda \in [a, b]$ where $g_a(x) = \sum_{i \in [d]} h_i(x)$, $g_b(x) = \max_{i \in [d]} h_i(x)\}$, which is any parametric class that interpolates monotonically between the egalitarian/equity (i.e., min max) and utilitarian/efficiency (i.e., min sum) objectives [60, 61].

Special cases of $\mathbf{C}_2$ include the classes of (a) L_p norm objectives $\left\{ \|(h_1, \ldots, h_d)\|_p : p \geq 1 \right\}$ [62], (b) convex combinations of equity and efficiency objectives [63, 64], and (c) top-ℓ norm objectives [65]. We explore these in more detail in Section 3.4, and our results will apply to all these settings.

Beyond the foundational understanding of portfolios for these classes of functions, we next shift our focus to applications. In particular, we discuss a novel extension of the facility location problem (described next) motivated by the formation of medical deserts [66] due to

This chapter is based on joint work with Swati Gupta and Mohit Singh. A preliminary version appeared in the *Proceedings of the Twenty-Fourth ACM Conference on Economics and Computation (EC) 2023* [59]. An extended version is currently under submission.

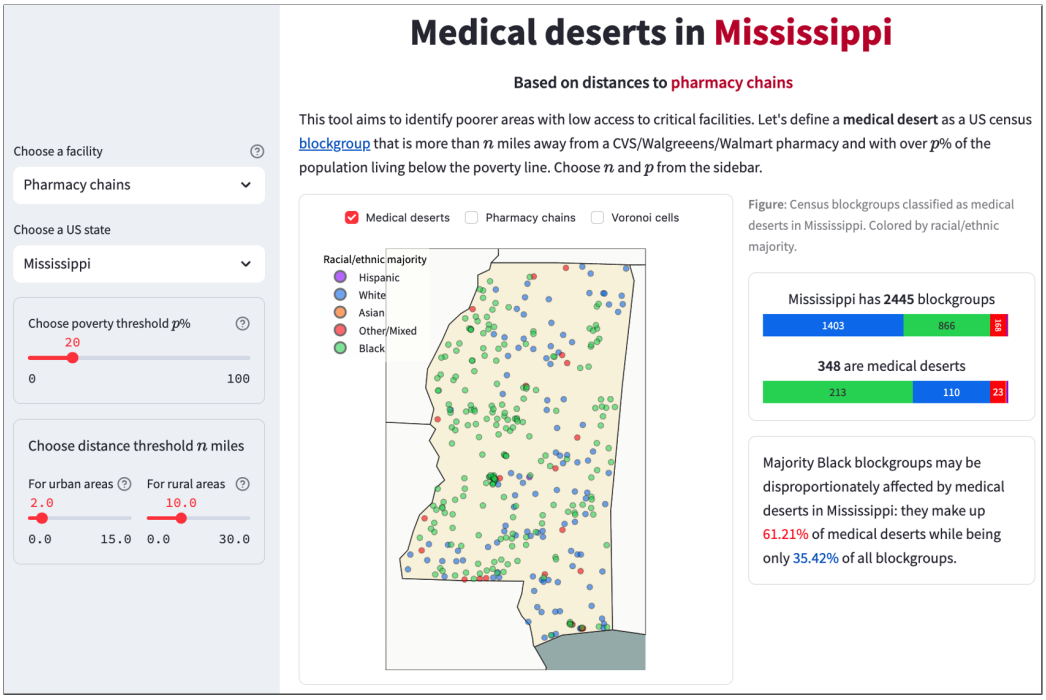
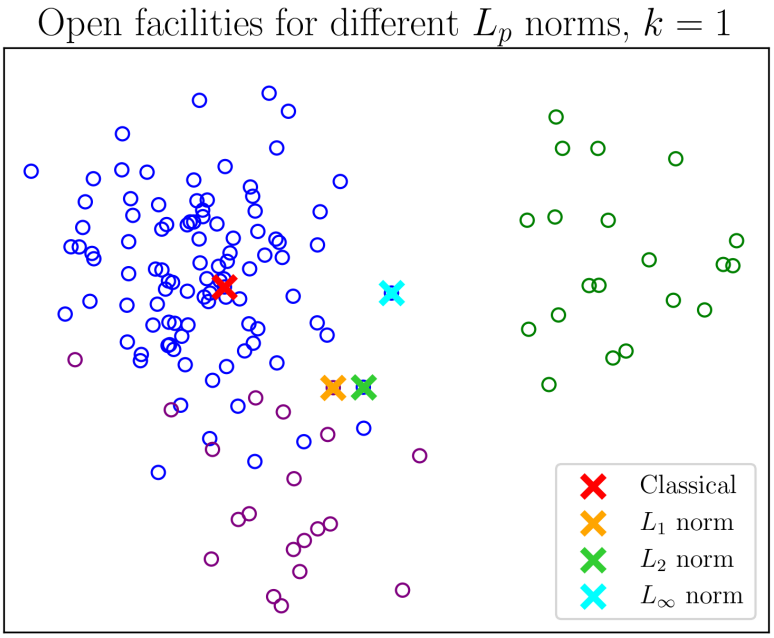


Figure 3.1: A screenshot from our online tool depicting medical deserts in Mississippi, USA. Note that the majority Black blockgroups (35.42% of all blockgroups) make up 61.03% of all medical deserts. The tool can be found at https://usa-medical-deserts.streamlit.app/.



Group	Classical	L_1 norm	L_2 norm	L_∞ norm
1 (blue)	0.21	0.42	0.47	0.46
2 (green)	0.89	0.71	0.66	0.54
3 (purple)	0.51	0.32	0.36	0.55
all clients	0.35	0.45	0.48	0.48

Figure 3.2: An illustrative example for k-CLUSTERING with three client groups C_1, C_2, C_3 (in blue, green, and purple, respectively) that partition client set C. We seek to open a single facility f anywhere in C. The optimal solution for the classical objective $\sum_{j \in C} \texttt{dist}(j, f)$ opens facility f near the center of the blue group. If we minimize the L_p norm of vector $\left(\frac{1}{|C_s|} \sum_{j \in C_s} \texttt{dist}(j, f)\right)_{s=1,2,3}$ of average group distances, then f moves closer to the center of all groups as p increases from 1 to ∞. The adjacent table shows average group distances for optimal solutions to different objectives.

the closure of pharmacies. We design an approximation algorithm for this problem, which then results in portfolios for facility location. We develop a web tool (see Figure 3.1) which can be useful for policymakers to guide investment decisions. We also discuss several other applications in Section 3.2.

Fair Subsidized Facility Location. Inequity in the placement of critical facilities is a well-documented problem [67, 68, 69, 63]. For instance, it is suspected that profit maximization by grocery chains has led to the formation of food deserts spread widely across

the U.S., which are defined as regions with low-income populations and low access to fresh food (e.g., people with no cars, and no grocery stores within a mile of their house) [68, 70]. Similarly, a recent New York Times study [66] showed that the closure of local pharmacies across the U.S. has resulted in the creation of pharmacy deserts. These smaller local stores often cannot sustain themselves in the wake of rising operating costs and drug prices, thus leading to closure. On the other hand, large pharmacy chains collect their revenues from a much broader source of users and have a higher negotiating power for drug prices, thus being more robust to financial difficulties.

We first consider a fair facility location problem, where given a $p \geq 1$, the goal is to minimize the L_p norm of the distances traveled by each group of people. Our notion of portfolios is directly applicable here. We show that a portfolio of solutions can help us get around the choice of p, so the decision-maker can focus on the placement of facilities suggested by the portfolio, instead of debating modeling choices.

Specifically to optimize for profit-sustaining facilities, we propose a novel subsidized model of facility location where some of the profits from the large profit-making pharmacies can be diverted to the loss-making locations to sustain them, while improving access throughout all neighborhoods. Our model addresses the fundamental tension between *profitability* and *healthcare access*. This problem models how a central planner might strategically place pharmacies to ensure coverage in underserved areas, allowing some facilities to operate at a loss, which is subsidized by more profitable locations. These losses are bound by a fraction δ (called *subsidy*) of the revenue that the planner can specify in advance, thus allowing them to be fairer to underserved communities while keeping losses limited. Our model generalizes the extensively-studied classical k-CLUSTERING and UNCAPACITATEDFACILITYLOCATION problems [71, 72, 73, 74]; see Theorem 3.6 in Section 3.7.

Formally, in Fair Subsidized Facility Location (FSFL), we are given a set of clients C with non-negative revenues $r : C \to \mathbb{R}_{\geq 0}$, potential facilities F with nonnegative operating costs $c : F \to \mathbb{R}_{\geq 0}$ and distances $\texttt{dist}$ on $C \cup F$ that form a metric, i.e., satisfy the triangle

inequality. A solution consists of a set of open facilities $F' \subseteq F$ and an assignment of each client to an open facility represented by Π. An open facility $f \in F'$ is profitable if its operating cost c_f is less than the revenue $\sum_{j \in C: \Pi(j)=f} r_j$ brought by clients assigned to it, and unprofitable otherwise. The solution (F', Π) is said to be δ-*subsidized* if the total loss of unprofitable facilities is at most a fraction δ of the total revenue $\sum_{j \in C} r_j$. Each client j in the model can have a fractional[1] group membership (e.g., dependent on geographic locations or socio-economic status) represented by $\mu_{j,s} \geq 0$ for each group $s \in [t]$. The goal is to open a set $F' \subseteq F$ of facilities and provide an assignment $\Pi : C \to F'$ of clients to open facilities, so that (i) the solution is δ-subsidized, and (ii) the distances of clients to facilities are equitable under various objectives, i.e., we would like to minimize the travel costs associated with $\mathbf{C} = \left\{ \|\mathbf{h}(\Pi)\|_p := \left(\sum_{s \in [d]} \left(h(\Pi)_s \right)^p \right)^{1/p} : p \geq 1 \right\}$, where $h(\Pi)_s := \sum_{j \in C} \mu_{j,s} \texttt{dist}_{j,\Pi(j)}$ is the distance travelled by group $s \in [d]$. Opening fewer facilities leads to δ-subsidized solutions, but can increase travel costs for clients. Therefore, we would like to optimize travel costs under various L_p-norms, while respecting that the solution is approximately δ-subsidized. Apart from L_p norms of group distances, we also consider convex and sublinear[2] functions $g : \mathbb{R}^C_{\geq 0} \to \mathbb{R}$ of client distances $\texttt{dist}_{j,\Pi(j)}$.

3.1.1 Technical Results

Having discussed our modeling contributions, we next discuss our key technical results for (i) portfolios for conic combinations $\mathbf{C}_1$, (ii) portfolios for interpolating functions $\mathbf{C}_2$, and (iii) portfolios for FSFL. We end the section with a brief overview of our experimental results on constructing portfolios to recommend subsidized pharmacies in the U.S.

Portfolios for Conic Combinations. To begin with, we give portfolios for the class $\mathbf{C} = \{\sum_{j \in [d]} \lambda_j h_j : \lambda \in \mathbb{R}^d_{\geq 0}\}$ of conic combinations of given base objective functions,

[1]Fractional group memberships have not received much attention in the existing literature at the intersection of optimization, ML, and algorithmic fairness, which usually assumes that clients divide into non-intersecting groups, or considers the finest partition of intersections as unique individual groups [75, 76, 77].

[2]Function $g : \mathbb{R}^C_{\geq 0} \to \mathbb{R}$ is sublinear if $g(\alpha\tau + \tau') \leq \alpha g(\tau) + g(\tau')$ for all $\alpha \geq 0$ and $\tau, \tau' \in \mathbb{R}^C_{\geq 0}$.

over any domain $\mathcal{D}$ such that each objective $\sum_j \lambda_j h_j$ can be optimized over $\mathcal{D}$ up to a β-approximation (for given $\beta \geq 1$, as we are considering minimization problems). The portfolio size is bounded in terms of the *imbalance* $u := \max_{i,j\in[d],x\in\mathcal{D}} \frac{h_i(x)}{h_j(x)}$ of the base objectives, defined as the maximum ratio between any two base functions at any point $x \in \mathcal{D}$ in the feasible set. Note that the base functions need not be convex.

Theorem 3.1. *Let $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$ be positive (base) functions on feasible set $\mathcal{D}$ and some $u \in \mathbb{R}_{>0}$ be such that at any point $x \in \mathcal{D}$ and any two functions h_i, h_j, it holds that $\frac{h_i(x)}{h_j(x)} \leq u$. Let $\mathbf{C} = \{\sum_{i\in[d]} \lambda_i h_i : \lambda \in \mathbb{R}^d_{\geq 0}\}$ denote the class of all conic combinations of $h_1, \ldots, h_d$, and let $\mathcal{O}_\beta$ be an oracle to obtain a β-approximate solution to $\arg\min_{x\in\mathcal{D}} h(x)$ for any $h \in \mathbf{C}$ for given $\beta \geq 1$. Then, a $\beta(1+\varepsilon)$-approximate portfolio of size*

$$|X_\varepsilon| = d \cdot \left(\frac{12 \log(4du/\varepsilon)}{\varepsilon}\right)^{d-1}$$

can be constructed using at most $\mathrm{poly}(|X_\varepsilon|)$ *number of oracle calls, for any $\varepsilon > 0$.*

For example, for base functions $h_1(x) = x + 1, h_2(x) = x^2 + 1, h_3 = x^3 - x^2 + 2$ over domain $\mathcal{D} = [0, 1] \subseteq \mathbb{R}$, we have $u = 2$, and the above theorem gives a $(1+\varepsilon)$-approximate portfolio of size $O\left(\frac{1}{\varepsilon^2}\left(\log\frac{1}{\varepsilon}\right)^2\right)$. Unlike previous results that use a multiplicative ε-mesh to construct such portfolios, our result combines it with an *additive* ε-mesh, leading to better approximation ratios (see Subsection 3.1.3 and Section 3.3 for details). Despite the result's generality, the dependence on d (number of base functions) is exponential, but we show in Subsection 3.3.2 that this is unavoidable, unless there are more structural assumptions on $\mathbf{C}$ or the domain. We next present an example where a portfolio of a much smaller size can be constructed by exploiting the properties of the function class.

Example. $\mathbf{C} = \{\sum_{i\in[d]} \lambda_i h_i : \lambda \geq 0\}$ on $\mathcal{D} = \mathbb{R}$, where $h_i(x) = \exp(\theta|x - y_i|)$, for y_i uniformly sampled from $[0, 1]$ and some fixed $\theta > 0$. Then the grid $X = \left\{\frac{\log(1+\varepsilon)}{\theta}, \frac{2\log(1+\varepsilon)}{\theta}, \ldots, 1\right\}$ of $\frac{\theta}{\log(1+\varepsilon)} \leq \frac{2\theta}{\varepsilon}$ points is a $(1+\varepsilon)$-approximate portfolio for any $\varepsilon \in (0, 1]$.

Portfolios for Monotonically Interpolating Classes. In the above example, the portfolio size of $2\theta/\varepsilon$ is independent of d, and depends on the quality of the approximation factor and the growth of the functions in the class $\mathbf{C}$. We use similar ideas to generalize existing results (e.g., [62], [9]) to show the existence of $(1+\varepsilon)$-approximate portfolios of size $O\left(\frac{\log d}{\varepsilon}\right)$ for interpolating classes of objectives:

Theorem 3.2. *Let $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ be d nonnegative functions on feasible set $\mathcal{D}$, and denote $\mathbf{h} = (h_1, \ldots, h_d)$. Let $\mathbf{C}$ denote a class of objectives that interpolate monotonically between $\|\mathbf{h}\|_1$ and $\|\mathbf{h}\|_\infty$, and let $\mathcal{O}_\beta$ be an oracle that gives a β-approximate solution to $\arg\min_{x\in\mathcal{D}} h(x)$ for any $h \in \mathbf{C}$. There exists an algorithm that given any $\varepsilon \in (0, 1]$, finds a $\beta(1+\varepsilon)$-approximate portfolio of size at most $O\left(\frac{\log \beta d}{\varepsilon}\right)$ in $\text{poly}(d, \frac{1}{\varepsilon})$ number of oracle calls to $\mathcal{O}_\beta$.*

Note that Theorem 3.1 and Theorem 3.2 use an approximation oracle $\mathcal{O}_\beta$ to construct the portfolio of solutions. Thus, they reduce the problem of finding portfolios to finding a β-approximate oracle for the underlying optimization problem for fixed objectives.

Connections with the Pareto Frontier. Portfolios are closely connected to the notion of *Pareto frontier* of base functions $h_1, \ldots, h_d$. Given points $x, y \in \mathcal{D}$, we say that $x \prec y$ or x dominates y if $h_i(x) \leq h_i(y)$ for all $i \in [d]$ and strict inequality holds for some i. The Pareto frontier $F_{\mathbf{h}} = \{\mathbf{h}(x) : x \in \mathcal{D} \text{ is not dominated by any } y \in \mathcal{D}\}$ is the set of non-dominated function values. Given $\alpha \geq 1$, the α-approximate Pareto frontier $F_{\mathbf{h}}(\alpha) := \left\{\mathbf{h}(x) : x \in \mathcal{D}, \frac{1}{\alpha}\mathbf{h}(x) \in F_{\mathbf{h}}\right\}$ is the set of all function values that are coordinate-wise within an α-approximation of the Pareto frontier. Pareto frontiers require a coordinate-wise dominance, and in general can be much larger than portfolios for conic combinations of $h_1, \ldots, h_d$. For example, consider $\mathcal{D} = \{x \in \mathbb{R}^2 : x_1 + x_2 = 1, x \geq 0\}$ and let h_i simply be identity function for each coordinate i, i.e., $h_i(x) = x_i$ for $i \in \{1, 2\}, x \in \mathbb{R}^2$. Then, $\mathbf{h}(x) = x$ for all $x \in \mathbb{R}^2$. Since each point in $\mathcal{D}$ is non-dominated by any other, the Pareto frontier $F_{\mathbf{h}}$ must contain every point in $\mathcal{D}$. In particular, $F_{\mathbf{h}}$ is an infinite set. However, given any $\lambda_1, \lambda_2 \geq 0$, consider minimizing $\lambda_1 h_1(x) + \lambda_2 h_2(x) = \lambda_1 x_1 + \lambda_2 x_2$

over $\mathcal{D}$. Clearly, $(0,1)$ or $(1,0)$ is always optimal depending on whether λ_1 is greater than λ_2, and therefore, the set $\{(1,0),(0,1)\}$ of two points is an optimal portfolio for conic combinations of functions h_1, h_2.

Bicriteria Oracle for the Facility Location Problem. We next discuss the approximation algorithm for FSFL. Like many other NP-hard problems [78, 79, 16], FSFL does not even admit a β-approximate oracle for any constant $\beta > 0$. In fact, even checking whether a given solution x is feasible (i.e., whether $x \in \mathcal{D}$) is NP-hard for FSFL. We show this via a reduction from the Subset Sum problem (proof included in Section A.1):

Theorem 3.3. *Unless P = NP, (A) FSFL is inapproximable to within any constant factor even when the objective is the sum of client distances, and (B) there is no polynomial-time algorithm to check the feasibility of a solution to FSFL.*

In such cases, the natural next step is to relax one of the constraints by some factor γ to obtain an extension $\mathcal{D}(\gamma) \supseteq \mathcal{D}$ of the feasible set. For example, in the FSFL problem, one can relax the δ-subsidy constraint to require that the total loss is at most $\gamma \times \delta \sum_j r_j$. An algorithm or oracle that returns solutions $x \in \mathcal{D}(\gamma)$ with objective value within factor α of the optimum in $\mathcal{D}$ is called a *bicriteria (α, γ)-approximation*.

Our definition of portfolios is general enough to accommodate bicriteria approximations: *given an optimization problem with feasible solutions $\mathcal{D}$, a class $\mathbf{C}$ of objective functions and a desired (α, γ) bicriteria approximation, a* portfolio *is a set $P \subseteq \mathcal{D}(\gamma)$ such that for each objective $h \in \mathbf{C}$, there is some $x \in P$ that is an (α, γ)-approximate solution to h, i.e., $h(x) \leq \alpha \min_{y \in \mathcal{D}} h(y)$, where $\mathcal{D}(\gamma)$ is the relaxation of $\mathcal{D}$.*

The guarantees given in Theorem 3.1 and Theorem 3.2 also generalize to such bicriteria problems, as long as the required oracle can be constructed. In the case of FSFL, we allow the oracle to increase the total loss beyond a fraction δ of the total revenue (while still bounding it), i.e., the δ-subsidy condition. This extension is necessary for approximations and allows for meaningful trade-offs between profitability and access costs to facilities in

our application. To state our result, we make the θ-*small revenues* assumption, wherein $\theta = \max_{j \in C, f \in F} \frac{r_j}{c_f}$ is defined as the maximum fraction of a facility's operating cost that can be met by a single client's revenue. Most commercial facilities like pharmacies rely on a large number of clients and so θ is typically very small. Our result formally states the following:

Theorem 3.4. *There exists a polynomial-time algorithm that given an instance of Fair Subsidized Facility Location (FSFL) that satisfies θ-small revenues assumption and a subsidy $\delta > 0$, returns a $(2\delta + \theta)$-subsidized solution whose objective value is within factor $O\left(\max\left(1, \frac{1}{\delta}\right)\right)$ of the optimal δ-subsidized solution.*

Our algorithm builds on the Linear Programming rounding approximation algorithm of [71] for k-CLUSTERING and UNCAPACITATEDFACILITYLOCATION (UFL) with linear objectives, and our bound matches their approximations within constant factors. There are two challenges in extending their algorithm to FSFL: first, our objectives are sublinear rather than linear, and second, their algorithm does not account for the new subsidy constraint. We observe that their algorithm generalizes to sublinear objectives; however, the second challenge requires a new combinatorial subroutine (Subsection 3.5.1) that ensures that the total losses are bounded. Moreover, there is no α-approximation for UFL with linear objectives if $\alpha < 1.463$ unless P = NP [80], implying that our bound for $\delta > 1$ is tight up to constants.

As a corollary of the above theorem, we obtain portfolios for FSFL with d client groups:

Corollary 3.1. *There exists a polynomial-time algorithm that given an instance of Fair Subsidized Facility Location (FSFL) with d client groups that satisfies θ-small revenues assumption and a subsidy parameter $\delta > 0$, obtains a portfolio P of size $O\left(\log\left(\frac{d}{\min(1,\delta)}\right)\right)$ such that*

1. *each solution in the portfolio is $(2\delta + \theta)$-subsidized, and*

2. *for each $p \geq 1$, there is a solution in the portfolio with objective value within factor $O\left(\max\left(1, \frac{1}{\delta}\right)\right)$ of the optimum δ-subsidized solution for the L_p norm objective.*

In practice, this means that the algorithm can offer decision-makers a small menu of (say) 3-5 facility layouts that span a wide range of fairness preferences. This is computationally efficient and allows flexibility without overwhelming the user, as we later show in our experiments on US Census data and pharmacy chains CVS, Walgreens, and Walmart. To complement the upper bound, we show the existence of instances of FSFL with d client groups where any $O(1)$-approximate portfolio for L_p norms must have size $\Omega(\log d)$. This shows that the portfolio size obtained by our algorithm is the best possible (up to constant factors):

Theorem 3.5. *There exist instances of FSFL with $\delta > 1$ where any $O(1)$-approximate portfolio for the class $\mathbf{C}$ of L_p norms of group distances with d client groups must have size $\Omega(\log d)$.*

Next, we complement our theoretical results with experiments on U.S. Census data and locations of pharmacy chains CVS, Walgreens, and Walmart.

Experiments. Recall our web tool (Figure 3.1) for identifying medical deserts, which are regions with over 20% poverty rate and that are further than n miles from their nearest CVS, Walgreens, or Walmart pharmacy, where $n = 2$ for urban areas and $n = 10$ for rural areas. In our experiments, we propose 10 new pharmacies alongside 206 existing CVS, Walgreens, and Walmart pharmacies in the state of Mississippi, USA. We divide the population into $d = 16$ groups based on the Congressional district, urbanization levels, and poverty levels, and give a portfolio of 3-5 solutions based on different L_p norm objectives with these groups and for different subsidy parameters δ.

Each solution in the portfolio recommends a different set of facilities, thus *offering a varied choice to the policymaker* (also see Table 3.1). Despite their diversity, they all *reduce the number of medical deserts identified by our tool from 348 to between 297-305*

Table 3.1: Percent reduction in average distance traveled (vis-a-vis status quo) by 16 different groups of people in the portfolio constructed by the portfolio algorithm in Section 3.4 to open new pharmacies in Mississippi, USA, with 2% subsidy ($\delta = 0.02$). Each column corresponds to a different L_p norm solution, to open 10 new facilities to add to the existing 206 CVS, Walgreens, and Walmart pharmacies. Groups are based on rurality, poverty levels, and congressional district. Bolded text represents optimal solutions for different groups (rows). See Section 3.6 for details.

Group			Portfolio Solutions			
Urban/ Rural	Poor/ Not Poor	Congress District	L_1 Norm	$L_{5.4}$ Norm	$L_{13.5}$ Norm	L_∞ Norm
Rural	Not Poor	1	0.0	0.0	**0.60**	0.0
		2	6.0	7.8	**11**	8.1
		3	14	**15**	**15**	13
		4	4.0	**4.1**	1.2	1.1
	Poor	1	0.0	0.0	**0.12**	0.0
		2	12	18	18	**19**
		3	**21**	**21**	**21**	**21**
		4	**14**	10	1.4	0.63
Urban	Not Poor	1	0.0	0.0	0.0	0.0
		2	**3.1**	0.0	0.0	0.0
		3	10	6.9	3.8	**11**
		4	**1.7**	0.0	0.0	0.0
	Poor	1	0.0	0.0	0.0	0.0
		2	**4.9**	0.0	0.0	3.0
		3	1.4	**2.6**	2.4	0.0
		4	**5.3**	0.0	0.0	0.0

(depending on the solution), while opening only 10 new facilities in all of Mississippi, and even with $\delta \leq 0.02$, i.e., while losing only 2% (additional[3]) revenue. Further, 70 to 80% of these ~50 blockgroups are majority Black or African American, thus *mitigating some of the disproportionate impact of medical deserts on the Black population.* These results are presented with further details in Section 3.6.

3.1.2 Outline

In Subsection 3.1.3, we discuss related work and previous approaches to portfolios and facility location problems. In Section 3.2, we present other applications of portfolios for

[3]That is, the total loss of any *new* facilities that we open must be within 2% of the total revenue of all clients.

conic combinations and interpolating functions. In Section 3.3, we prove Theorem 3.1 (upper bound) and Lemma 3.1 (lower bound) for conic combinations. In Section 3.4, we prove Theorem 3.2 that upper bounds portfolio sizes for monotonically interpolating families. In Section 3.5 we discuss Fair Subsidized Facility Location (FSFL) and prove Theorem 3.4, giving the approximation algorithm. We also prove Corollary 3.1 and Theorem 3.5 that bound portfolio sizes for FSFL. The reduction from k-CLUSTERING and UNCAPACITATEDFACILITYLOCATION to FSFL is presented in Section 5.4, while hardness results for FSFL are deferred to Section A.1. Experiments on U.S. Census data are discussed in Section 3.6, and we conclude in Section 3.8. We introduce notation as it arises in each section.

3.1.3 Related Work

Golovin *et al.* [23] study L_p norms and their results imply $O(1)$-approximate portfolios for L_p norms of size $O(\log d)$. Their technique however crucially uses the structure of L_p norms and does not generalize to other classes that monotonically interpolate between L_1 and L_∞ norms. Our technique for portfolio upper bounds for such families closely resembles the technique of Goel and Meyerson [9], who use it to get portfolios for top-ℓ norms.

Another well-known related concept is the notion of *Pareto frontier approximations* [29]. Pareto frontier of objectives $h_1, \ldots, h_d$ on feasible set $\mathcal{D}$ is the set of all points $\mathbf{h}(x)$ for $x \in \mathcal{D}$ where not all of the function values can be improved, i.e., for all $x' \in \mathcal{D}, h_i(x') > h_i(x)$ for some $i \in [d]$. The $(1+\varepsilon)$-approximate Pareto frontier relaxes this constraint by factor $(1+\varepsilon)$. [29] give algorithms to compute Pareto frontiers for several problems, assuming (stronger) bounds $\frac{1}{u} \leq h_i(x) \leq u$ for all $i \in [d]$ and $x \in \mathcal{D}$, and a stronger feasibility oracle[4]. In particular, they obtain $(1+\varepsilon)$-approximate Pareto frontiers of size $O\left(\left(\frac{\log u}{\varepsilon}\right)^d\right)$, and this is nearly-tight. Note that any Pareto-frontier also implies a portfolio,

[4]They require a polynomial-time oracle to check if the set $\mathcal{D} \cap \{x : h_i(x) \leq \lambda_i \; \forall \, i \in [d]\}$ is non-empty for given $\lambda_i, i \in [d]$, while we require only optimizing $\min_{x \in \mathcal{D}} \sum_{i \in [N]} \lambda_i h_i(x)$, which is usually much simpler for many combinatorial problems. For example, if functions h_i represent different path lengths between two vertices in a graph, then the first oracle is NP-hard while the second reduces to another shortest path problem.

and therefore, they essentially get a $(1+\varepsilon)$-approximate portfolio of size $O\left(\left(\frac{\log u}{\varepsilon}\right)^d\right)$. Compared to our result, this is a more restrictive setting, but they obtain a smaller portfolio than ours.

A long line of works builds on these results to give Pareto frontier approximations for specific settings [30, 31, 18]. In particular, [31] extend the results of [81] that use weaker oracles for conic combinations, similar to our result, while still considering the scaling assumption on individual function values. They construct a Pareto frontier (and therefore a portfolio) of size $O\left(\left(\frac{\log u}{\varepsilon}\right)^d\right)$, and one can show this is $d(1+\varepsilon)$-approximate. Note that their result implies that every factor decrease in approximation quality $\varepsilon' = \varepsilon/\gamma$ (for $\gamma > 1$) increases the size of the portfolio exponentially by γ^d. Our setting on the other hand uses a weaker assumption[5] that $\frac{h_i(x)}{h_j(x)} \leq u$ for all $x \in D$, for all $i, j \in [d]$ and generalizes their setting. Further, our Theorem 3.1 improves upon the approximation ratio of [31] by a factor of d, producing a $(1+\varepsilon)$-approximate portfolio, without increasing the size exponentially by around d^d. Our portfolio is only of a slightly larger size $d \cdot \left(\frac{O(\log(du/\varepsilon))}{\varepsilon}\right)^{d-1}$. The main technical difference in our approach is that we use a combination of a multiplicative and additive ε-meshes, as opposed to the above results that use only a multiplicative mesh.

When portfolio size is greater than 1, portfolios for top-ℓ norms may not be portfolios for L_p norms (see Section A.3 for an example). This is in contrast to portfolios of size 1, where [9] prove that portfolios for top-ℓ norms are portfolios for L_p norms (and all symmetric monotonic norms).

L_p norm objectives are widely considered in the approximation algorithms literature as a model for fairness and as interesting theoretical questions [34, 23, 27, 35, 36, 8]. In particular, [23] study fixed norm objectives including L_p norm objectives for k-CLUSTERING. Our FSFL model thus generalizes their setting using the subsidy constraint (Section 3.7). Variants of both UNCAPACITATEDFACILITYLOCATION [72, 82, 71, 80, 83, 84, 73, 85, 86, 87, 88, 74] and k-CLUSTERING [89, 90, 91, 92, 93, 94] are very well-studied in the

[5]For any function that satisfies $1/u \leq h_i(x) \leq u$, it also satisfies $h_i(x)/h_j(x) \leq u^2$ for all $x \in \mathcal{D}$.

Operations Research and Computer Science literature. While all of these algorithms appropriately bound the number/cost of open facilities, they do not bound the total loss of facilities, which is where our new rounding subroutine is needed.

Many other kinds of fairness criteria in facility location problems are well-studied from both theoretical and applied perspectives, e.g., see [95, 96, 97, 98, 35]. Most of these models either fix a fairness criterion or do not discuss how a choice among a suite of fairness criteria must be made.

3.2 Other Applications

This section showcases several example domains where our portfolio framework applies. The Fair Subsidized Facility Location model is expanded on in Section 3.5; we present other applications here.

Bus Routing. School bus routing is a natural multiobjective optimization problem due to the involvement of several costs (maintaining school buses, fuel costs, etc) and several stakeholders (children, parents, schools). This problem presents different conflicting goals, such as minimizing transportation or fuel costs, the number of buses, and minimizing the wait times for different groups of students (e.g., students belonging to different racial groups) [99, 100, 101, 27].

A general approach to dealing with multiple objectives $h_1, \ldots, h_d$ (where h_1 is transportation cost, h_2 is average student waiting time, h_3 is number of buses etc) is to normalize them so their maximum values $\max_{x \in \mathcal{D}} h_i(x) = 1$ for each function $i \in [d]$. Then we define $m > 0$ as the smallest value any h_i takes at any $x \in \mathcal{D}$. *Given an algorithm to optimize* $\sum_{i \in [d]} \lambda_i h_i$ *for any given conic combination* $\lambda \geq 0$ *of* $h_1, \ldots, h_d$*, Theorem 3.1 (Section 3.3) then yields* $O(1)$*-approximate portfolios of size* $d \cdot \left(O\left(\log(\frac{d}{m})\right)\right)^{d-1}$ *for this setting for the class* $\mathbf{C} = \{\sum_i \lambda_i h_i : \lambda \in \mathbb{R}^d_{\geq 0}\}$ *of all conic combinations.*

Finance Portfolios. Consider the following simple model [102, 103] to trade off risk and rewards in finance portfolio optimization: we are given n assets with mean returns

$\mu = (\mu_1, \ldots, \mu_n) \in \mathbb{R}^n$ and variance $\Sigma \in \mathbb{R}^{n \times n}$ for these returns. We seek to diversify our investment among these assets to maximize returns (or minimize regret) while minimizing risk. Formally, we seek a distribution $x \geq 0$ with $\sum_i x_i = 1$ over the assets. One objective is to maximize the return $\mu^\top x$, or equivalently to minimize the reciprocal $h_1(x) = \frac{1}{\mu^\top x}$. The second objective – that models risk – is defined as $h_2(x) := \sqrt{x^\top \Sigma x}$.

Let $\sigma_1, \ldots, \sigma_n$ denote the eigenvalues of Σ, and assume without loss of generality that $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$. Then for any feasible x, we have $\frac{h_1(x)}{h_2(x)} \leq \frac{1}{\mu_1 \min(\sigma_i)} := a$. Similarly, for any feasible x, $\frac{h_2(x)}{h_1(x)} \leq \mu_n \max_i \sigma_i := b$. Our result (Theorem 3.1) yields an $O(1)$-approximate portfolio of size $O(\log(\max(a, b)))$ for the class $\mathbf{C} = \{\lambda_1 h_1 + \lambda_2 h_2 : \lambda_1, \lambda_2 \geq 0\}$ of conic combinations of h_1, h_2. Different solutions in the portfolio trade off returns and risk to different degrees.

This is an example of a more general phenomenon in stochastic optimization, where the mean $\mathbb{E}[X]$ (representing returns) of a nonnegative random variable X is often traded off against its second moment $\mathbb{E}[X^2]$ (representing risk or variance). Other examples include healthcare, where treatment effectiveness and risk can be modeled through the first and second moments respectively, and inventory management, where managing expected stockouts (first moment) and maintaining consistent supply (second moment) are often in conflict with each other. More generally, different moments $\mathbb{E}[X^p]$ of X can be traded off against each other. *When X has a finite support in d coordinates, our Theorem 3.2 gives at most $O(\log d)$ solutions while guaranteeing that for all $p \geq 1$, one of these solutions approximates the pth moment $\mathbb{E}[X^p]$ within factor 2^p of its optimal.*

Fair Representation Clustering. In this problem [104, 76], we are given a metric space $(C, \texttt{dist})$ on clients C, an integer k, and one of d colors $\chi_j \in [d]$ for each client $j \in C$, and some efficiency constraint. The clients $\{j \in C : \chi_j = t\}$ of color $t \in [d]$ are denoted Γ_t, and we denote $r_t = \frac{|\Gamma_t|}{|C|}$ to be the fraction of clients of color t.

We seek clusters $C(1), \ldots, C(k) \subseteq C$ that partition C while satisfying the given efficiency constraint. The goal is to distribute the clients among the clusters in the same ratio

as their population: for cluster $C(i)$ and color $t \in [d]$, the deviation $h_t(i)$ for color t in cluster i is defined as

$$h_t(i) = \left| r_t - \frac{|C(i) \cap \Gamma_t|}{|C(i)|} \right|,$$

and we define $h_t = \max_{i \in [k]} h_t(i)$ to be the maximum deviation for color t among all clusters. Given an instance of the Fair Representation Clustering problem, we define d objectives $h_1, \ldots, h_d$ corresponding to the d colors. Previous works have considered the egalitarian/min-max objective $\|h\|_\infty$ and the utilitarian/min-sum objective $\|h\|_1$ [104]. *Theorem 3.2 yields an $O(1)$-approximate portfolio of size $O(\log d)$ for all L_p norm objectives.*

Risk-averse Stochastic Programming. In standard two-stage stochastic programming, we seek to choose a decision vector $x \in \mathcal{D}_x$ in stage 1, after which a *scenario* $\omega \in \Omega$ is realized according to a given distribution $\mathscr{D}$. At this stage, we must choose another decision vector $y \in \mathcal{D}_{y,\omega}$ based on the realized scenario ω. The goal is to minimize $\mathbb{E}_{\mathscr{D}}[G_\omega(x, y)]$ for a given convex function G that depends on the realized ω.

The distribution $\mathscr{D}$ is often unknown or only partially known ahead of time. In this case, we must make a *risk-averse* [105] decision x in the first stage that is effective even for an adversarial distribution $\mathscr{D}$. Since a single solution must hedge against all possible realizations of ω, its worst-case performance is inherently limited. Instead, if we are allowed to choose a *portfolio* of solutions before the scenario ω is revealed to us, then we can guarantee a good approximation to $G_\omega(x, y)$ irrespective of what distribution $\mathscr{D}$ is. Here, the class of objectives $\mathbf{C} = \{G_\omega : \omega \in \Omega\}$ is indexed by scenarios Ω and the feasible set is the set of decisions $\mathcal{D}_x$. More generally, using portfolios, we can interpolate between two-stage stochastic and two-stage robust optimization.

3.3 Portfolios for Conic Combinations

In this section, we establish an exponential upper bound (Theorem 3.1) on the portfolio size for conic combinations of base objectives $h_i : \mathcal{D} \to \mathbb{R}_{>0}$, $i \in [d]$. Then, we show that

this portfolio size is necessarily exponential in the number d of base objectives.

3.3.1 Portfolio Size Upper Bound

The upper bound result is in terms of the *imbalance* of the base objective functions, defined as follows: we say that d positive functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$ are u-balanced for some $u \geq 1$ if for all $x \in \mathcal{D}$ and for all $i, j \in [d]$, $\frac{h_i(x)}{h_j(x)} \leq u$.

Theorem 3.1. *Let $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$ be positive (base) functions on feasible set $\mathcal{D}$ and some $u \in \mathbb{R}_{>0}$ be such that at any point $x \in \mathcal{D}$ and any two functions h_i, h_j, it holds that $\frac{h_i(x)}{h_j(x)} \leq u$. Let $\mathbf{C} = \{\sum_{i\in[d]} \lambda_i h_i : \lambda \in \mathbb{R}^d_{\geq 0}\}$ denote the class of all conic combinations of $h_1, \ldots, h_d$, and let $\mathcal{O}_\beta$ be an oracle to obtain a β-approximate solution to $\arg\min_{x\in\mathcal{D}} h(x)$ for any $h \in \mathbf{C}$ for given $\beta \geq 1$. Then, a $\beta(1+\varepsilon)$-approximate portfolio of size*

$$|X_\varepsilon| = d \cdot \left(\frac{12 \log(4du/\varepsilon)}{\varepsilon}\right)^{d-1}$$

can be constructed using at most $\text{poly}(|X_\varepsilon|)$ *number of oracle calls, for any $\varepsilon > 0$.*

We present the proof for the case $\beta = 1$, i.e., when the oracle returns the optimal solution for each $g_\lambda := \sum_j \lambda_j h_j$. The generalization to the case $\beta > 1$ is straightforward and omitted.

Proof. The plan is as follows: for each $i \in [d]$, define the $(d-1)$-dimensional hypercube $\mathcal{H}_i := \{\lambda \in [0,1]^d : \lambda_i = 1\}$. We show first that it is sufficient to restrict to convex combinations $g_\lambda = \sum_{i\in[d]} \lambda_i h_i$ where $\lambda \in \mathcal{H}_1 \cup \mathcal{H}_2 \cup \ldots \cup \mathcal{H}_d$ by rescaling the highest coordinate of λ to be 1. We will partition each $\mathcal{H}_i$ using an ε-parameterized *mesh* and choose a single representative λ^* for each part of the mesh so that the optimal point $x(\lambda^*) = \arg\min_{x\in\mathcal{D}} g_{\lambda^*}(x)$ for $g_{\lambda^*} := \sum_{j\in[d]} \lambda^*_j h_j$ will be a $(1+\varepsilon/4)$-approximate solution for g_λ for every other λ in the part. Bounding the number of these parts will give the desired bound on the portfolio size.

To create this mesh, we will find some $\delta > 0$ and partition $\mathcal{H}_i = \mathcal{H}'_i \cup R_i$ where

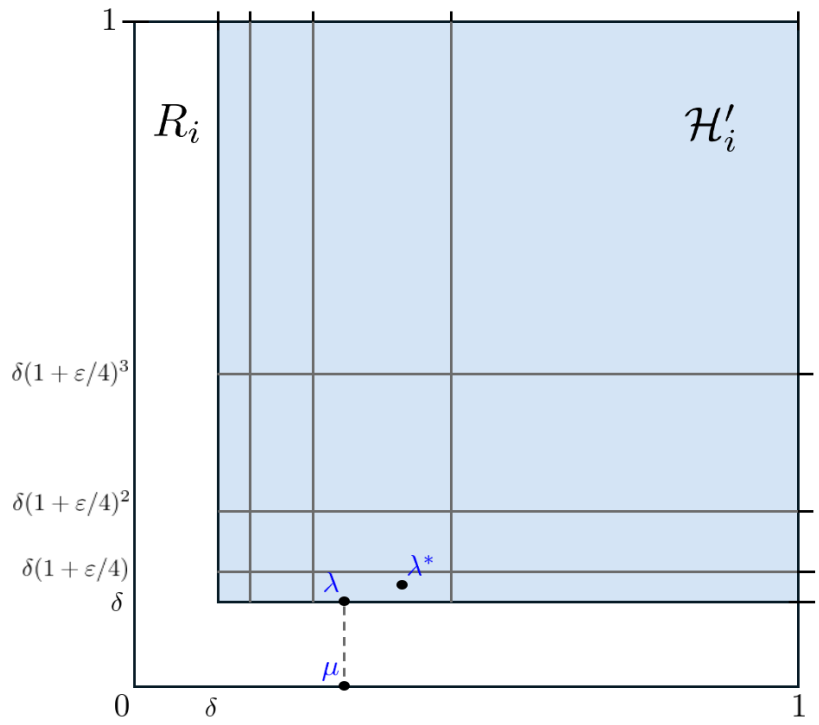


Figure 3.3: An illustration for the mesh for $\mathcal{H}_i$ used in the proof of Theorem 3.1.

$\mathcal{H}'_i = \{\lambda \in [\delta, 1]^d : \lambda_i = 1\}$ is a smaller $(d-1)$-dimensional hypercube contained within $\mathcal{H}_i$, and $R_i = \mathcal{H}_i \setminus \mathcal{H}'_i$ are the remaining points in $\mathcal{H}_i$. On this smaller hypercube $\mathcal{H}'_i$, we put the standard *multiplicative* mesh that partitions it into hyper-rectangles (see Figure 3.3). That is, we define nested boxes using *thresholds* $\{\delta, \delta(1+\varepsilon/4), \delta(1+\varepsilon/4)^2, \ldots, 1\}$ for each coordinate (not equal to i). The number of thresholds is $T+1 := \log_{1+\varepsilon/4}(1/\delta)+1$ for each dimension. For every $\lambda \in \mathcal{H}'_i$, we can assign it to one of the T^{d-1} boxes based on where the coordinates of λ lie within these thresholds, i.e., $\lambda \in [\delta(1+\varepsilon/4)^{k_1}, \delta(1+\varepsilon/4)^{k_1+1}] \times \ldots \times [\delta(1+\varepsilon/4)^{k_d}, \delta(1+\varepsilon/4)^{k_d+1}]$, for appropriate $k_1, \ldots, k_d$.

For the remaining points $R_i = \mathcal{H}_i \setminus \mathcal{H}'_i$, we create an *additive* mesh. The key observation is the following: for all $\lambda, \mu \in \mathbb{R}^d_{\geq 0}$, the optimal point $x(\lambda) := \arg\min_{x \in \mathcal{D}} g_\lambda(x)$ for the linear combination $g_\lambda = \sum_j \lambda_j h_j$ is an approximate solution for g_μ, and the quality of this approximation can be bounded in terms of the L_1 norm distance $\|\lambda - \mu\|_1$. Each $\mu \in R_i$ can first be mapped to some $\lambda \in \mathcal{H}'_i$ with $\|\lambda - \mu\|_1 \leq 2d\delta$, and then this λ can be mapped to the corresponding λ^* in the multiplicative mesh. As we show later, for the appropriate

choice of δ, the first step loses approximation factor $(1+\varepsilon/4)$. The second step loses factor $(1+\varepsilon/4)^2$, and thus the final approximation factor is at most $(1+\varepsilon/4)^3 \leq 1+\varepsilon$ for $\varepsilon \in (0,1]$. We choose $\delta = \frac{\varepsilon}{4ud}$, and we show the remaining claims next.

Approximation guarantee. First, say $\lambda \in \mathcal{H}_i'$. Then, by construction of the multiplicative mesh, we have that any λ^* in the same part of the mesh satisfies that $\frac{1}{1+\varepsilon/4} \leq \frac{\lambda_j}{\lambda_j^*} \leq 1+\varepsilon/4$. This implies that $x(\lambda^*)$ is a $(1+\varepsilon/4)^2$-approximation for g_λ using the optimality of $x(\lambda^*)$ for g_{λ^*}:

$$g_\lambda(x(\lambda^*)) = \sum_j \lambda_j h_j(x(\lambda^*)) \leq (1+\varepsilon/4)\sum_j \lambda_j^* h_j(x(\lambda^*)) \tag{3.1}$$

$$\leq (1+\varepsilon/4)\sum_j \lambda_j^* h_j(x(\lambda)) \leq (1+\varepsilon/4)^2 \sum_j \lambda_j h_j(x(\lambda)) \tag{3.2}$$

$$= (1+\varepsilon/4)^2 \min_{x\in\mathcal{D}} g_\lambda(x).$$

Now, suppose we pick some $\mu \in R_i$. Define $\lambda = (\max(\mu_1,\delta),\ldots,\max(\mu_d,\delta))$. Then $\lambda \in \mathcal{H}_i'$. We will show using a sequence of inequalities that $x(\lambda^*)$ is a $(1+\varepsilon/4)^3$-approximation for g_μ, where as above λ^* is the representative point of the part of the mesh that contains λ. To show this, we need the following claim:

Claim 3.1. *For all $i,j \in [d]$, all $\lambda \in \mathcal{H}_i$ and all $x \in \mathcal{D}$, we have $h_j(x) \leq u \cdot g_\lambda(x)$.*

Proof. We have $g_\lambda(x) = \sum_{\ell\in[d]} \lambda_\ell h_\ell(x) = \|\lambda\|_1 \sum_{\ell\in[d]} \frac{\lambda_\ell}{\|\lambda\|_1} h_\ell(x)$. However, $\sum_\ell \frac{\lambda_\ell}{\|\lambda\|_1} h_\ell(x)$ is a convex combination of the base objectives $h_1(x),\ldots,h_d(x)$. Since the base objectives are u-balanced, $\sum_\ell \frac{\lambda_\ell}{\|\lambda\|_1} h_\ell(x) \geq \sum_\ell \frac{\lambda_\ell}{\|\lambda\|_1}\left(\frac{1}{u}\cdot h_j(x)\right) = \frac{h_j(x)}{u}$. This implies that $g_\lambda(x) \geq \|\lambda\|_1 \frac{h_j(x)}{u}$, or that $h_j(x) \leq \frac{u\cdot g_\lambda(x)}{\|\lambda\|_1}$. Since $\lambda \in \mathcal{H}_i$, the ith coordinate $\lambda_i = 1$, so that $\|\lambda\|_1 \geq 1$. □

With this claim, we have the following bound on the performance of $x(\lambda^*)$ for g_μ:

$$g_\mu(x(\lambda^*)) = \sum_{j\in[d]} \mu_j h_j(x(\lambda^*)) = \sum_{j\in[d]} (\lambda_j + \underbrace{(\mu_j - \lambda_j)}_{\leq 0}) h_j(x(\lambda^*))$$

$$\leq \sum_{j\in[d]} \lambda_j h_j(x(\lambda^*)) = g_\lambda(x(\lambda^*)).$$

However, from Equation 3.1, $g_\lambda(x(\lambda^*)) \leq (1+\varepsilon/4)^2 g_\lambda(x(\lambda))$. Using optimality of $x(\lambda)$ for g_λ,

$$\begin{aligned} g_\lambda(x(\lambda)) \leq g_\lambda(x(\mu)) &= \sum_{j\in[d]} (\mu_j + (\lambda_j - \mu_j)) h_j(x(\mu)) \\ &= g_\mu(x(\mu)) + \sum_j (\lambda_j - \mu_j) h_j(x(\mu)). \end{aligned}$$

Finally, we use Claim 3.1 bounding $h_j(x(\mu)) \leq u \cdot g_\mu(x(\mu))$, so that

$$\begin{aligned} g_\lambda(x(\lambda)) - g_\mu(x(\mu)) &\leq \sum_j (\lambda_j - \mu_j) \times u \cdot g_\mu(x(\mu)) \\ &= u \cdot g_\mu(x(\mu)) \sum_j (\lambda_j - \mu_j) \leq u \cdot g_\mu(x(\mu)) \cdot (d \cdot \delta) \end{aligned}$$

since $0 \leq \lambda_j - \mu_j \leq \delta$ for all j. Since $\delta = \frac{\varepsilon}{4ud}$, we get $g_\lambda(x(\lambda)) \leq \left(1 + \frac{\varepsilon}{4}\right) g_\mu(x(\mu))$. Together, for $\varepsilon \in (0, 1]$,

$$g_\mu(x(\lambda^*)) \leq g_\lambda(x(\lambda^*)) \leq (1+\varepsilon/4)^2 g_\lambda(x(\lambda)) \leq (1+\varepsilon/4)^3 g_\mu(x(\mu)) \leq (1+\varepsilon) g_\mu(x(\mu)).$$

Portfolio size guarantee. Each $\mathcal{H}_i$ is an $(d-1)$-dimensional hypercube, and so its multiplicative mesh has size $\left(\log_{1+\varepsilon/4}(1/\delta)\right)^{d-1} \leq \left(\frac{12}{\varepsilon}\log(1/\delta)\right)^{d-1} = \left(\frac{12}{\varepsilon}\log\left(\frac{4ud}{\varepsilon}\right)\right)^{d-1}$. The portfolio size is upper bounded by the union of the sizes of the meshes:

$$d \times \left(\log_{1+\varepsilon/4}(1/\delta)\right)^{d-1} \leq d\left(\frac{12}{\varepsilon}\log\left(\frac{4ud}{\varepsilon}\right)\right)^{d-1}. \qquad \square$$

3.3.2 Portfolio Size Lower Bound

We show that the portfolio size for conic combinations $\mathbf{C} = \{\sum_{j\in[d]} \lambda_j h_j : \lambda \geq 0\}$ of base functions $h_1, \ldots, h_d$ must be exponential in d in some cases. In fact, our base functions $h_i, i \in [d]$ will be linear.

Lemma 3.1. *For all $d \geq 1$, there exists a set $\mathcal{D}$ and base functions $h_1, \ldots, h_d$ on $\mathcal{D}$ such that any 2-approximate portfolio for the class $\mathbf{C} = \{\sum_{j\in[d]} \lambda_j h_j : \lambda \geq 0\}$ of conic combinations of base functions must have size $\geq 2^d - 1$.*

Proof. We specify the base functions first: $h_i(x) = x_i$ for all $x \in \mathbb{R}^d$ and $i \in [d]$ (i.e., the ith coordinate). Given a set $S \subseteq [d]$, let $\chi_S \in \{0,1\}^d$ denote its characteristic vector. Then for all $S \neq \emptyset$, the function $h_S(x) := \chi_S^\top x \in \mathbf{C}$. We will construct $\mathcal{D} = \{x(S) : S \neq \emptyset\}$ with $|\mathcal{D}| = 2^d - 1$ such that for each S, there will be a unique minimizer $x(S) = \min_{x\in\mathcal{D}} h_S(x)$ and further that for all $T \neq S$, $x(T)$ will not be a 2-approximation for h_S.

Fix constants $a = 3$ and $b = 2da^d$ that we will use to define the vectors $x(S)$. For all $T \neq \emptyset$, define $x(T)$ as follows:

$$x(T)_i = \begin{cases} a^{|T|} & \text{if } i \in T, \\ b & \text{if } i \notin T. \end{cases}$$

Given some $S, T \neq \emptyset$, we get that

$$h_S(x(T)) = \sum_{i\in S\cap T} a^{|T|} + \sum_{i\in S\setminus T} b = |S \cap T|a^{|T|} + b|S \setminus T|.$$

In particular, $h_S(x(S)) = |S|a^{|S|}$. We now show that for all $T \neq S$, $h_S(x(T)) > 2h_S(x(S))$.

Case I: $S \subsetneq T$. Then $h_S(x(T)) = |S|a^{|T|}$. Since $S \neq T$, we must have $|T| > |S|$ and so $h_S(x(T)) > a \cdot |S|a^{|S|} = 3 \cdot h_S(x(S))$.

Case II: $S \setminus T \neq \emptyset$. Then $h_S(x(T)) > b = 2da^d \geq 2|S|a^{|S|} = 2h_S(x(S))$. This completes the proof.

3.4 Portfolios for Interpolating Functions

In this section, given base objective functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ over a domain $\mathcal{D}$ of feasible solutions, we prove Theorem 3.2 that obtains portfolios for classes $\mathbf{C}$ of functions that interpolate monotonically between the L_1 norm objective $\|\mathbf{h}\|_1 := h_1 + \ldots + h_d$ and the L_∞ norm objective $\|\mathbf{h}\|_\infty := \max_{i \in [d]} h_i$.

Formally, given reals $a \leq b$, the class $\mathbf{C} = \{g_\lambda : \lambda \in [a, b]\}$ of objectives on $\mathcal{D}$ interpolates monotonically between $\|\mathbf{h}\|_1$ and $\|\mathbf{h}\|_\infty$ if (1) $g_a = \|\mathbf{h}\|_1$, (2) $g_b = \|\mathbf{h}\|_\infty$, and (3) g_λ is non-increasing with λ, i.e., for all $\mu \geq \lambda$ and $x \in \mathcal{D}$, we must have $g_\lambda(x) \geq g_\mu(x)$. Examples of such a class include L_p norm functions $\left\{ \|\mathbf{h}\|_p := \left(\sum_{i \in [d]} h_i^p \right)^{1/p} : p \geq 1 \right\}$, convex combinations of $\|\mathbf{h}\|_1$ and $\|\mathbf{h}\|_\infty$, and top-ℓ norm functions for $\ell \in [d]$ that sum the ℓ highest coordinates.

We show that given an oracle to find a β-approximate solution for any given $g_\lambda \in \mathbf{C}$, we can obtain a $\beta(1+\varepsilon)$-approximate portfolio of size $O\left(\frac{\log \beta d}{\varepsilon}\right)$ in a polynomial number of oracle calls.

Theorem 3.2. *Let $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ be d nonnegative functions on feasible set $\mathcal{D}$, and denote $\mathbf{h} = (h_1, \ldots, h_d)$. Let $\mathbf{C}$ denote a class of objectives that interpolate monotonically between $\|\mathbf{h}\|_1$ and $\|\mathbf{h}\|_\infty$, and let $\mathcal{O}_\beta$ be an oracle that gives a β-approximate solution to $\arg\min_{x \in \mathcal{D}} h(x)$ for any $h \in \mathbf{C}$. There exists an algorithm that given any $\varepsilon \in (0, 1]$, finds a $\beta(1+\varepsilon)$-approximate portfolio of size at most $O\left(\frac{\log \beta d}{\varepsilon}\right)$ in $\mathrm{poly}(d, \frac{1}{\varepsilon})$ number of oracle calls to $\mathcal{O}_\beta$.*

Given this result, one might naturally ask if portfolios for one class of monotonically interpolating norms (e.g., top-ℓ norms) are also portfolios for another such class (e.g., L_p norms); indeed, a portfolio of size 1 that is α-approximate for top-ℓ norms is α-approximate

for L_p norms [9]. In Section A.3, we show that this is false for portfolios of size > 1. That is, there exist portfolios for top-ℓ norms which are not approximate portfolios for L_p norms.

Proof of Theorem 3.2. Denote the class $\mathbf{C} = \{g_\lambda : \lambda \in [a, b]\}$. We assume we are given an oracle to obtain β-approximation for g_λ over $\mathcal{D}$ for any given $\lambda \in [a, b]$. In a polynomial number of oracle calls, we will find a $\beta(1 + \varepsilon)$-approximate portfolio for $\mathbf{C}$. The portfolio size is $S + 1$ where $S = \log_{1+\varepsilon}(\beta d)$.

We construct a sequence of λ values $a = \lambda(0) < \lambda(1) < \ldots < \lambda(S) = b$ such that the set X of β-approximate solutions for the $S + 1$ objective functions $g_{\lambda(0)}, g_{\lambda(1)}, \ldots, g_{\lambda(S)}$ forms the desired portfolio. For $\lambda \in [a, b]$, let $\mathrm{OPT}_\lambda := \min_{x \in \mathcal{D}} g_\lambda(x)$ be the optimal value for g_λ, with $x(\lambda)$ denoting the β-approximate solution returned by the oracle. The objective value of this solution for g_λ is denoted ALG_λ. Since the oracle is β-approximate, $\mathrm{ALG}_\lambda \leq \beta \cdot \mathrm{OPT}_\lambda$.

We can assume without loss of generality that ALG_λ is non-increasing with λ: indeed, given $\mu > \lambda$, the cost of $x(\lambda)$ for g_μ is $g_\mu(x(\lambda)) \leq g_\lambda(x(\lambda)) = \mathrm{ALG}_\lambda$ by the monotonicity assumption. Therefore, if $g_\mu(x(\mu)) > g_\lambda(x(\lambda))$, we can use $x(\lambda)$ instead of $x(\mu)$ with better a objective value for g_μ.

Next, for $\lambda \in [a, b]$, denote by λ' the minimum value of $\mu \in [a, b]$ such that $\frac{\mathrm{ALG}_\lambda}{(1+\varepsilon)} \geq \mathrm{ALG}_\mu$. Intuitively, we take a 'step' of size $(1+\varepsilon)$ in the objective value. If no such μ exists (i.e., when $\mathrm{ALG}_b > \frac{\mathrm{ALG}_\lambda}{1+\varepsilon}$), define $\lambda' = b$. Construct portfolio X as follows: initially set $\lambda(0) = a$ and $i = 0$. While $\lambda(i)' < b$, keep taking $(1+\varepsilon)$-steps, i.e., setting $\lambda(i+1) = \lambda(i)'$ and increasing the counter i. Suppose $\lambda(0), \ldots, \lambda(S)$ is the sequence of λ values generated by this algorithm. The algorithm outputs the corresponding β-approximations, i.e., $X = \{x(\lambda(i)) : i \in [0, S]\}$.

We claim that for each $i \in [0, S - 1]$ and $\mu \in [\lambda(i), \lambda(i + 1))$, solution $x(\lambda(i))$ is a $\beta(1 + \varepsilon)$-approximation to g_μ. This is sufficient to prove the approximation guarantee of the portfolio. Since $g_\mu \leq g_{\lambda(i)}$, the cost of $x(\lambda(i))$ for g_μ is $g_\mu(x(\lambda(i)) \leq g_{\lambda(i)}(x(\lambda(i)) = \mathrm{ALG}_{\lambda(i)}$. Now, by definition of $\lambda(i + 1)$, we have $\mathrm{ALG}_\mu \geq \frac{\mathrm{ALG}_{\lambda(i)}}{1+\varepsilon}$, so that this cost is at

most $(1+\varepsilon)\mathrm{ALG}_\mu \le (1+\varepsilon)\beta \cdot \mathrm{OPT}_\mu$. This proves the approximation guarantee for X.

We now prove that $|X| = O(\log_{1+\varepsilon}(\beta d))$. By construction, $\mathrm{ALG}_{\lambda(i+1)} \le \mathrm{ALG}_{\lambda(i)}/(1+\varepsilon)$, i.e., the value of $\mathrm{ALG}_{\lambda(i)}$ decreases by factor $\ge (1+\varepsilon)$ in each step (except possibly the last). Therefore, the number of steps S is bounded by $\log_{(1+\varepsilon)} \frac{\mathrm{ALG}_a}{\mathrm{ALG}_b}$. It is now sufficient to prove that $\mathrm{ALG}_a \le (d \cdot \beta) \cdot \mathrm{ALG}_b$. To see this claim, since $x(a)$ is a β-approximation to $g_a = \|\mathbf{h}\|_1$, we have $\mathrm{ALG}_a = \|\mathbf{h}(x(a))\|_1 \le \beta \cdot \|\mathbf{h}\|_1(x(b))$. Next, since $\|\mathbf{h}(x(b))\|_1 = h_1(x(b)) + \ldots + h_d(x(b)) \le d \cdot \|\mathbf{h}(x(b))\|_\infty$, we get $\mathrm{ALG}_a \le \beta d \cdot \mathrm{ALG}_b$ since g_b is the L_∞ norm. □

3.5 Fair Subsidized Facility Location

In this section, we construct a polynomial-time approximate oracle (and consequently portfolios) for the Fair Subsidized Facility Location (FSFL) problem, where the input consists of (1) a metric space $(X, \mathtt{dist})$ on set $X = C \cup F$ of clients C and potential facilities F, (2) nonnegative operating costs $c : F \to \mathbb{R}_{\ge 0}$ for facilities, (3) nonnegative revenues $r : C \to \mathbb{R}_{\ge 0}$ for clients, (4) *subsidy* parameter $\delta > 0$, and (5) an objective function $g : \mathbb{R}^C_{\ge 0} \to \mathbb{R}$ on client distances that is convex and sublinear, i.e., for all $\tau, \tau' \in \mathbb{R}^C_{\ge 0}$ and $\alpha \ge 0$, we must have $g(\alpha\tau + \tau') \le \alpha g(\tau) + g(\tau')$. A feasible solution is a pair (F', Π) where (1) $F' \subseteq F$ is a set of open facilities and (2) $\Pi : C \to F'$ are client assignments to open facilities.

The mild conditions of convexity and sublinearity on the objective lead to a rich class of objectives: it includes classical objectives such as the sum of client distances and the maximum client distance, as well as norms of client group distances when client groups are specified. In particular, this includes the L_p norms of client group distances, where we are given d groups through fractional group memberships $\mu_{j,s} \ge 0$ for each client $j \in C$ and each group $s \in [d]$. For solution (F', Π), the group distance of the sth group is $h_s^{(\Pi)} := \sum_{j \in C} \mu_{j,s} \mathtt{dist}_{j,\Pi(j)}$ and the L_p norm objective for given $p \ge 1$ is to minimize $\|\mathbf{h}^{(\Pi)}\|_p := \left(\sum_{s \in [d]} (h_s^{(\Pi)})^p\right)^{1/p}$. Formally, our result states the following:

Table 3.2: A summary of various steps in the rounding algorithm for FSFL.

Solution	x	y	Distance approximation	Subsidy
(x, y)	fractional	fractional	1	δ
$\downarrow$ α-PointRounding			(Lemma 3.2)	(Lemma 3.2)
$(\overline{x}, \overline{y})$	fractional	fractional	$4 \cdot \max\left(1, \frac{1}{\delta}\right)$	2δ
$\downarrow$ RoundToIntegralFacilities			(Lemma 3.4)	(Lemma 3.5)
(x', y')	fractional	integral	$20 \cdot \max\left(1, \frac{1}{\delta}\right)$	2δ
$\downarrow$ IntegralAssignment			(Lemma 3.6)	(Lemma 3.6)
(x'', y')	integral	integral	$20 \cdot \max\left(1, \frac{1}{\delta}\right)$	$2\delta + \theta$

Theorem 3.4. *There exists a polynomial-time algorithm that given an instance of Fair Subsidized Facility Location (FSFL) that satisfies θ-small revenues assumption and a subsidy $\delta > 0$, returns a $(2\delta + \theta)$-subsidized solution whose objective value is within factor $O\left(\max\left(1, \frac{1}{\delta}\right)\right)$ of the optimal δ-subsidized solution.*

We start by writing a convex relaxation for the problem. We represent feasible solutions through characteristic vectors: the set $F' \subseteq F$ of open facilities is represented by the binary vector $y \in \{0,1\}^F$ such that $y_f = 1$ if and only if $f \in F'$. Similarly, an assignment $\Pi : C \to F'$ is represented by the vector $x \in \mathbb{R}^{C \times F}$ such that $x_{j,f} = 1$ if and only if $\Pi(j) = f$ for all clients $j \in C$ and $f \in F$.

These vectors satisfy two natural constraints: (1) a client is only assigned to an open facility, i.e., $x_{j,f} \leq y_f$ for all $j \in C, f \in F$ and (2) each client $j \in C$ must be assigned to some facility, or $\sum_{f \in F} x_{j,f} = 1$. This is useful for writing convex relaxations [89, 71] of the problem, where we instead allow $x_{j,f}$ and y_f to be in $[0,1]$ for each client $j \in C$ and facility $f \in F$ under the above constraints. Further, the distance τ_j of a client $j \in C$ to its assigned facility can be written as $\sum_{f \in F} x_{j,f} \texttt{dist}_{j,f}$.

We introduce a variable ℓ_f for each $f \in F$ to denote the loss of the facility f, and impose the constraint $\ell_f \geq \max\left(0, y_f c_f - \sum_{j \in C} x_{j,f} r_j\right)$. That is, the $\ell_f = 0$ when $y_f = 0$ or when f is not open. When f is open but profitable, i.e., revenue $\sum_{j \in C} x_{j,f} r_j$ of

clients assigned to it exceeds the operating cost $c_f y_f$, then loss $\ell_f = 0$, otherwise ℓ_f is the difference in the operating cost and revenue. The final constraint is the δ-subsidy constraint that bounds the total loss $\sum_{f \in F} \ell_f$ to be at most $\delta \sum_{j \in C} r_j$.

$$\min g(\tau) \tag{IP}$$

$$\text{s.t. } \tau_j = \sum_{f \in F} x_{j,f} \texttt{dist}_{j,f}, \qquad \forall\, j \in C, \tag{3.3}$$

$$\sum_{f \in F} x_{j,f} = 1, \qquad \forall\, j \in C, \tag{3.4}$$

$$x_{j,f} \leq y_f, \qquad \forall\, j \in C, f \in F, \tag{3.5}$$

$$\ell_f \geq c_f y_f - \sum_{j \in C} x_{j,f} r_j, \qquad \forall\, f \in F, \tag{3.6}$$

$$\sum_{f \in F} \ell_f \leq \delta \sum_{j \in C} r_j, \tag{3.7}$$

$$\ell_f \geq 0, \qquad \forall\, f \in F, \tag{3.8}$$

$$x \in \{0,1\}^{C \times F}, y \in \{0,1\}^F.$$

Due to the convexity of g, we can relax the integrality constraints on x, y to get a convex program and obtain an optimal *fractional solution* (x, y, ℓ, τ) in polynomial time. Since ℓ and τ can be determined using x, y, we will often omit them and denote the solution as (x, y). When both x, y are integral, we call (x, y) an *integral* solution. In this case, (x, y) corresponds to a feasible solution (F', Π) of the original problem. Since g is sublinear, we have that $g(\gamma\tau) \leq \gamma g(\tau)$ for all $\gamma > 0$.

Given the optimal fractional solution (x, y) to the convex program, in Subsection 3.5.1 we round it to a solution (x', y') where y' is integral (but x' may be fractional) using algorithms α-`PointRounding` and `RoundToIntegralFacilities` respectively. Then, we round (x', y') to an integral solution (x'', y') in Subsection 3.5.2, using a subroutine for the generalized assignment problem from [106]. Table 3.2 illustrates these steps in the rounding algorithm. Subsection 3.5.3 discusses portfolios for FSFL.

3.5.1 Finding an Integral Set of Facilities

The first step in our rounding procedure is the α-PointRounding subroutine from [71]. For appropriately chosen $\alpha \in (0,1)$ and fractional optimal (x,y), this algorithm removes the assignment of clients $j \in C$ to facilities that are the farthest and hold at most $(1-\alpha)$ fraction of the assignment for j. The algorithm then rescales the y variables by $1/\alpha$ while maintaining feasibility to the convex relaxation of Equation IP. We state the algorithm in Subsection A.2.1 for completeness. Here, we state the formal guarantees of α-PointRounding for facility location. Given a vector $\Delta \in \mathbb{R}^C_{\geq 0}$, we say that a fractional solution $(\hat{x},\hat{y})$ is Δ-close if for all clients $j \in C$ and all facilities $f \in F$ such that $\hat{x}_{j,f} > 0$, we have $\texttt{dist}_{j,f} \leq \Delta_j$ (note that Δ_j need not all be the same value for all j). Note that for any fractional solution (x,y), the actual distance $\texttt{dist}_{j,f}$ of a client j to any facility f that they are fractionally matched to could be much larger than the expected distance $\tau_j = \sum_{f' \in F} \texttt{dist}_{j,f'} x_{j,f'}$ that the convex program pays for. We show in the following lemma that this can be bounded, while violating the subsidy constraint by a factor ≤ 2:

Lemma 3.2. *The fractional solution $(\overline{x},\overline{y})$ output by α-PointRounding(x,y) (Algorithm 13) satisfies:*

1. *It is Δ-close where $\Delta_j \leq 4\max\left(1,\frac{1}{\delta}\right)\tau_j$ for all $j \in C$, where $\tau_j = \sum_f x_{j,f}\texttt{dist}_{j,f}$ is the expected distance in (x,y). That is, for any facility f,*

$$\overline{x}_{j,f} > 0 \implies \texttt{dist}_{j,f} \leq 4\max\left(1,\frac{1}{\delta}\right)\tau_j.$$

2. *The total loss of $(\overline{x},\overline{y})$ is at most fraction 2δ of the total revenue, i.e., $\sum_{f\in F}\overline{\ell_f} \leq 2\delta\sum_{j\in C} r_j$.*

Crucially, Part 1 states that a client j can now be assigned to *any* facility f such that $\overline{x}_{j,f} > 0$ while being within factor $4\max\left(1,\frac{1}{\delta}\right)$ of its original distance τ_j in the fractional

solution (x, y). This part of the lemma follows similar arguments to [71]. Part 2 of the lemma requires tracking the losses from the original solution (x, y) to the output solution $(\overline{x}, \overline{y})$; we include details of the proof in Section A.2. Hereafter, we denote this set of 'feasible' facilities for client $j \in C$ as

$$\texttt{feas}_j := \{f \in F : \overline{x}_{j,f} > 0\}.$$

Lemma 3.2 implies that if we open a facility in $\texttt{feas}_j$ for each client $j \in C$ and assign j to this facility, then we obtain a solution where the objective g is within a factor $4 \max\left(1, \frac{1}{\delta}\right)$ of the relaxed optimal of Equation IP (since g is sublinear). However, opening a (potentially) different facility for each client can significantly increase the total operating cost of open facilities, and therefore the total loss. Therefore, we need to open fewer facilities while still ensuring that clients are not assigned too far away.

Algorithm `RoundToIntegralFacilities`. Our next algorithm (Algorithm 2) rounds the Δ-close fractional solution $(\overline{x}, \overline{y})$ to solution (x', y') with *integral* facilities, while increasing the client distances by factor ≤ 5 (i.e., (x', y') is 5Δ-close), while not increasing the losses. Our strategy is as follows: we will first find a set of *core clients* $C^* \subseteq C$. Next, we will open a single facility $f(j^*)$ in each $\texttt{feas}_{j^*}$ for each core client $j^* \in C^*$. This is the integral solution y' for open facilities. These core clients and open facilities satisfy the following two key properties simultaneously:

1. For core clients $j^* \in C^*$, the sets $\texttt{feas}_{j*}$ of feasible facilities have to be mutually disjoint.

2. For each client $j \in C$, there exists some core client $j^* \in C^*$ such that the open facility $f(j^*)$ is within distance $5\Delta_j$ of j.

These two properties are formalized in Lemma 3.3. We will then assign clients fractionally to y' while maintaining two properties:

Algorithm 1 $\texttt{CoreClients}(G, \Delta)$

input: an undirected graph $G = (C, E)$ on clients C and a function $\Delta : C \to \mathbb{R}_{\geq 0}$
output: $(C^*, \texttt{paths})$ where $C^* \subseteq C$ is a set of *core clients* and $\texttt{paths}_j$ is a path in G from j to some core client $j^* \in C^*$ for each $j \in C$
given a path $p = j_0 j_1 \dots j_T$ for $j_t \in C$ in G, define the Δ-length of p as $\Delta(p) = \sum_{t=0}^{T} \Delta_{j_t}$

1: if G is empty, return $(\emptyset, \emptyset)$
2: choose core client $j^* = \arg\min_{j \in C} \Delta_j$, initialize arborescence A rooted at j^* and $\texttt{paths}(j^*) = j^*$
3: call edge $jj' \in E$ *growing* if $j \notin A$, $j' \in A$, and $\Delta_j \geq \Delta(\texttt{paths}_{j'})$
4: **while** there is a growing edge $jj' \in E$ **do**
5: add directed edge $j' \to j$ to A
6: set $\texttt{paths}_j$ to be the path from j to j^* in A
7: obtain $(C^*, \texttt{paths}') = \texttt{CoreClients}(G \setminus A, \Delta)$ for the remaining instance
8: **return** $(\{j^*\} \cup C^*, \texttt{paths} \cup \texttt{paths}')$

1. *Bounded Losses*: Since sets $\texttt{feas}_{j^*}$ are disjoint for core clients j^*, clients j that are fractionally assigned to any facility in $\texttt{feas}_{j^*}$ can be reassigned to $f(j^*)$. Thus, the revenue from these clients pays for the facility $f(j^*)$.

2. *Bounded Distances*: From the second property above for core clients, each client j can be assigned to some $f(j^*)$ while keeping the objective g within factor $5 \times 4\max(1, \frac{1}{\delta})$ of the optimal objective.

Together, these two steps give us the required integral set of facilities and the corresponding fractional assignments.

Finding Core Clients. For the first step of finding the core set of clients, we present Algorithm $\texttt{CoreClients}$. For expositional convenience, we first construct the graph $G = (C, E)$ on clients C as follows: there is an edge $jj' \in E$ if and only if $\texttt{feas}_j \cap \texttt{feas}_{j'} \neq \emptyset$, and every vertex (client) $j \in C$ is associated with its Δ_j value. The Δ-length of a simple path $p = j_0 j_1 \dots j_T$ in G for clients $j_t \in C$ is defined as the sum $\sum_{t=0}^{T} \Delta_{j_t}$ of Δ-values along the path. Algorithm $\texttt{CoreClients}$ selects a subset $C^* \subseteq C$ that form an independent set in the graph G, while ensuring a small Δ-length path from any non-core

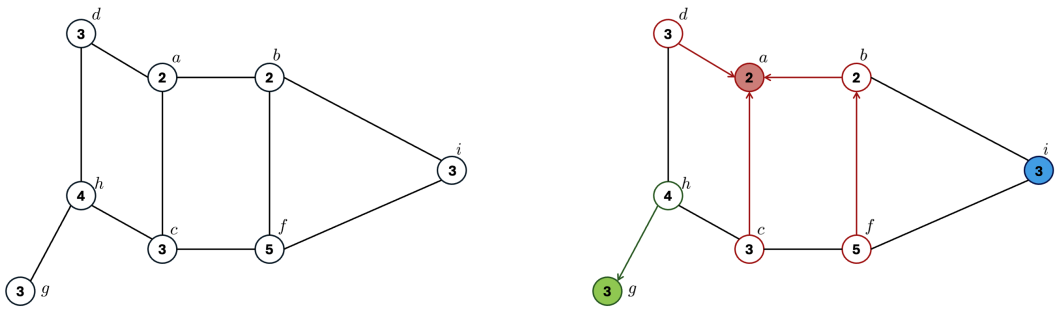


Figure 3.4: An example to illustrate Algorithm 1. (left) The graph $G = (C, E)$ with Δ values for vertices. Initially, the algorithm chooses core client $a = \arg\min_{j \in C} \Delta_j$ and forms an arboresence rooted at a on clients $\{a, b, c, d, f\}$. Then, the algorithm chooses core client g and forms the arborescence on clients $\{g, h\}$, and finally, the algorithm chooses the core client i. (right) The core clients C^* (shaded) and `paths`, represented through arborescences rooted at the core clients.

client to some core client. This is formalized in the following lemma:

Lemma 3.3. *Given the graph $G = (C, E)$ on clients C and $\Delta : C \to \mathbb{R}_{\geq 0}$,* `Core-Clients` *finds a subset $C^* \subseteq C$ and* $\texttt{paths}_j$ *for each $j \in C$ to some $j^* \in C^*$ such that*

1. *C^* is an independent set of vertices in G.*
2. *For each client $j \in C$, the path* $\texttt{paths}_j$ *starts at j and ends at some $j^* \in C^*$, and has Δ-length at most $\Delta(\texttt{paths}_j) \leq 2\Delta_j$.*
3. *For all neighbors $j \in C$ of a core client $j^* \in C^*$, we have $\Delta_j \geq \frac{1}{2}\Delta_{j^*}$.*

Before we discuss the proof, at a high level, algorithm `CoreClients` decomposes the graph G into a set of arborescences that are rooted trees at each core client, while ensuring that each non-core client has a directed path to some core client. This is done by recursively choosing the client j^* with the smallest Δ_{j^*} value (these are the core clients) and adding all 'nearby' clients to obtain the arborescence rooted at j^*. That is, for each client j in this arborescence rooted at j^*, the Δ-length of the path from j to j^* is at most $2\Delta_j$.

We then delete the subgraph spanned by the arborescence and recurse on the remaining components of the graph. The complete description is given in Algorithm 1. We are now ready to prove Lemma 3.3.

Proof. (Part 1) At any recursive call on a connected component H, a core client j^* must have the smallest Δ_{j^*} value in the current graph H. This means that each neighbor j in the current graph must belong to its arborescence, by trivially satisfying Δ-length of the $\Delta(\texttt{paths}_j) = \Delta_j + \Delta_{j^*} \leq 2\Delta_j$. Note that this also implies that any subsequent core client cannot be adjacent to j^*.

(Part 2) Clearly, the algorithm exhaustively assigns each client $j \in C$ to the arborescence A rooted at some core client $j^* \in C^*$. Suppose j was added to A through the edge jj'. Then, by construction, we have that $\Delta_j \geq \Delta(\texttt{paths}_{j'})$. Since $\texttt{paths}_j$ consists of edge jj' followed by $\texttt{paths}_{j'}$, we have $\Delta(\texttt{paths}_{j'}) = \Delta(\texttt{paths}_{j'}) + \Delta_j \leq 2\Delta_j$.

(Part 3) Consider edge $jj^* \in E$ with core client $j^* \in C^*$ and $j \in C$. From Part 1, $j \notin C^*$. Suppose first that j is in the arborescence of j^*. In this case, by Part 2, $\Delta_j + \Delta_{j^*} = \Delta(\texttt{paths}_j) \leq 2\Delta_j$, so that $\Delta_{j^*} \leq \Delta_j \leq 2\Delta_j$.

Next, suppose that j is not in the arborescence of j^*. Then, j must have been added to the arborescence A of some j_1^* chosen in C^* before j^*, but this arborescence did not include j^*. This can only happen if the Δ-path length $\Delta(\texttt{paths}_j) > \Delta_{j^*}$. From Part 2, $2\Delta_j \geq \Delta(\texttt{paths}_j) \geq \Delta_{j^*}$, completing the proof. □

Note that this lemma gives us our key properties discussed earlier: Part 1 implies that $\texttt{feas}_{j_1^*} \cap \texttt{feas}_{j_2^*} = \emptyset$ for all core clients $j_1^*, j_2^* \in C^*$ by definition of the graph G. Using Part 2, for all clients j, there is some core client j^* such that $\texttt{paths}_j$ ends at j^* and therefore $\Delta_{j^*} \leq \Delta(\texttt{paths}_j) - \Delta_j \leq \Delta_j$, so that $\texttt{dist}_{j,f(j^*)} \leq \texttt{dist}_{j,j^*} + \texttt{dist}_{j^*,f(j^*)} \leq 4\Delta_j + \Delta_{j^*} \leq 5\Delta_j$.

Integral Facilities with Fractional Assignments. Taking the Δ-close fractional solution $(\overline{x}, \overline{y})$, we will return a 5Δ-close solution (x', y') with y' integral using the following Algorithm `RoundToIntegralFacilities` (Algorithm 2). Using `CoreClients` as a subroutine, the algorithm runs in two phases, where Phase 1 opens the cheapest facility $f(j^*) \in \texttt{feas}_{j^*}$ for each core client $j^* \in C^*$ and partially assigns clients to bound the

Algorithm 2 `RoundToIntegralFacilities`$(\overline{x}, \overline{y})$

input: Δ-close fractional solution $(\overline{x}, \overline{y})$
output: 5Δ-close solution (x', y') where y' is integral (x' may be fractional)

1: for each client $j \in C$, define $\texttt{feas}_j := \{f \in F : \overline{x}_{j,f} > 0\}$
2: form graph $G = (C, E)$ on vertex set C with edge $j_1 j_2 \in E$ if and only $\texttt{feas}_{j_1} \cap \texttt{feas}_{j_2} \neq \emptyset$
3: $(C^*, \texttt{paths}) \leftarrow \texttt{CoreClients}(G, \Delta)$

Phase 1: open facilities and partially assign clients

4: **for** $j^* \in C^*$ **do**
5: let $f(j^*) := \arg\min_{f \in \texttt{feas}_{j^*}} c_f$ be the cheapest facility in $\texttt{feas}_{j^*}$
6: set $y'_{f(j*)} = 1$, and assign $x_{j^*, f(j^*)} = 1$ ▷ Open facility and assign
7: **for** each neighbor j of j^* in G **do**
8: set $x'_{j,f(j^*)} = \sum_{f \in \texttt{feas}_{j^*}} \overline{x}_{j,f}$ to be the contribution of j to facilities in $\texttt{feas}_{j^*}$

Phase 2: fully assign clients

9: **for** each client $j \in C$ that is partially assigned, i.e., $\sum_f x'_{j,f} < 1$ **do**
10: let $j^* \in C^*$ be the endpoint of path $\texttt{paths}(j)$
11: increase $x'_{j,f(j^*)} \leftarrow x'_{j,f(j^*)} + \left(1 - \sum_f x'_{j,f}\right)$ ▷ Fully assign j
12: set any undefined $x'_{j,f}$ and y'_f to 0
13: **return** (x', y')

losses of these facilities as described above. Each client's contribution to any facility in $\texttt{feas}_{j^*}$ is assigned to the chosen $f(j^*)$. Phase 2 then assigns the remainder contribution of the clients to the open facility $f(j^*)$ where j^* is the end of the path $\texttt{paths}_j$ obtained using `CoreClients`. This keeps their distances to assigned facilities bounded.

In Phase 1, for each core client $j^* \in C^*$, the cheapest facility $f(j^*) \in C^*$ is opened. For each neighbor j of j^* in G, we fractionally assign j to $f(j^*)$, setting $x'_{j,f(j^*)}$ to be the contribution of j to facilities in $\texttt{feas}_{j^*}$, that is, $x'_{j,f(j^*)} = \sum_{f \in \texttt{feas}_{j^*}} \overline{x}_{j,f}$. First, we show that `RoundToIntegralFacilities` returns a valid fractional solution (x', y'). It is clear that (x', y') satisfies Equation 3.5 in the relaxation of Equation IP. It is sufficient to show that $\sum_f x'_{j,f} \leq 1$ for all $j \in C$, since in Phase 2 (line 11) we ensure that $\sum_f x'_{j,f} \geq 1$ for all $j \in C$.

Claim 3.2. *For each client $j \in C$, $\sum_f x'_{j,f} \leq 1$ at the end of Phase 1 in Algorithm* `RoundToIntegralFacilities`.

Proof. $\texttt{feas}_{j_1^*}$ and $\texttt{feas}_{j_2^*}$ are disjoint for all distinct core clients $j_1^*, j_2^* \in C^*$. Therefore, at the end of Phase 1,

$$\sum_f x'_{j,f} = \sum_{j^* \in C^*} x'_{j,f(j^*)} = \sum_{j^* \in C^*} \sum_{f \in \texttt{feas}_{j^*}} \overline{x}_{j,f} \leq \sum_{f \in \texttt{feas}_j} \overline{x}_{j,f} = 1. \qquad \square$$

Our next lemma bounds the increase in client distances:

Lemma 3.4. *For each client $j \in C$ and facility $f \in F$, if $x'_{j,f} > 0$ then $\texttt{dist}_{j,f} \leq 5\Delta_j$.*

Proof. Each client $j \in C$ is potentially (fractionally) assigned to some facilities in Phase 1 and to at most one facility in Phase 2.

Phase 1. Suppose j is fractionally assigned to some $f(j^*)$ for $j^* \in C^*$ in Phase 1. Then j is a neighbor of j^*, so there is some $f \in \texttt{feas}_j \cap \texttt{feas}_{j^*}$, that is, $\overline{x}_{j,f} > 0$ and $\overline{x}_{j^*,f} > 0$. Algorithm `CoreClients` guarantees that $\Delta_j \geq \frac{1}{2}\Delta_{j^*}$ in this case (Lemma 3.3), so that

$$\begin{aligned} \texttt{dist}_{j,f(j^*)} &\leq \texttt{dist}_{j,f} + \texttt{dist}_{j^*,f} + \texttt{dist}_{j^*,f(j^*)} \\ &\leq \Delta_j + \Delta_{j^*} + \Delta_{j^*} \leq \Delta_j + 2\Delta_j + 2\Delta_j = 5\Delta_j. \end{aligned}$$

Phase 2. j is assigned to $f(j_T)$ for $j_T \in C^*$ where $\texttt{paths}(j) = jj_1j_2 \ldots j_{T-1}j_T$ is a path in G. Algorithm `CoreClients` guarantees (Lemma 3.3) that $\Delta_{j_1} + \Delta_{j_2} + \ldots + \Delta_{j_T} \leq \Delta_j$, and therefore the distance of j to $f(j_T)$ is at most

$$2(\Delta_{j_1} + \Delta_{j_2} + \ldots + \Delta_{j_T}) + \Delta_j \leq 3\Delta_j. \qquad \square$$

Our next lemma shows that the total loss of unprofitable facilities in (x', y') is at most the total loss of unprofitable facilities in $(\overline{x}, \overline{y})$:

Lemma 3.5. *The total loss for the rounded solution (x', y') returned by* `RoundTo-IntegralFacilities`$(\overline{x}, \overline{y})$ *is $\sum_{f \in F} \ell'_f \leq \sum_{f \in F} \overline{\ell_f}$.*

The proof is based on the following idea: we only open one facility $f(j^*)$ for each core

client $j^* \in C^*$. Further, the set C^* is an independent set in the graph $G = (C, E)$. Recall that for each $j^* \in C^*$, its neighbors in G contribute some revenue to $f(j^*)$ in Phase 1 of the algorithm. We show that since $f(j^*)$ is chosen as the cheapest facility in $\texttt{feas}_{j^*}$, this contribution from the neighbors of j^* is sufficient to offset the operating cost of $f(j^*)$, up to the loss incurred in the fractional solution $(\overline{x}, \overline{y})$. We defer the proof of this lemma to Subsection A.2.2.

Note that since the total loss $\sum_f \overline{\ell}_f$ of $(\overline{x}, \overline{y})$ is at most $2\delta \sum_{j \in C} r_j$ (Lemma 3.2), we have from the above lemma that $\sum_f \ell'_f \leq 2\delta \sum_{j \in C} r_j$ as well. Now that we have a solution (x', y') with integral open facilities $\{f \in F : y'_f = 1\}$, it remains to round the fractional assignments x'.

3.5.2 Finding Integral Assignments

In our final rounding step, we find integral assignments for the 2δ-subsidized fractional solution (x', y') with integral y' and subsidy at most 2δ. This rounding procedure gives an integral solution (x'', y') but increases the subsidy to at most $2\delta + \theta$. This proof crucially uses the θ-small revenues assumption, i.e., $r_j \leq \theta c_f$ for all clients $j \in C$ and facilities $f \in F$. We have the following more general lemma:

Lemma 3.6. *There exists a polynomial-time rounding algorithm that given a δ'-subsidized Δ'-close fractional solution (x', y') where y' is integral, obtains a $(\delta' + \theta)$-subsidized Δ'-close integral solution (x'', y') for any instance of FSFL that satisfies the θ-small revenues assumption for given $\theta > 0$.*

To prove this lemma, we consider a parallel scheduling problem where the machines are open facilities $F' = \{f \in F : y'_f = 1\}$, jobs are clients C, and processing time $p_{j,f} = r_j$ if $x'_{j,f} > 0$ and $p_{j,f} = \infty$ otherwise (i.e., job j cannot be assigned to machine f). Denote the set of clients that can be assigned to f as $C_f := \{j \in C : x'_{j,f} > 0\}$. Our proof uses [106]'s scheduling algorithm as a subroutine, stated in terms of our problem:

Lemma 3.7 ([106]). *There exists a polynomial-time algorithm that given a fractional schedule x', returns an integral schedule x'' where (1) each job/client j is assigned to exactly one facility/machine in the support of x' and (2) for each machine $f \in F'$, the total load $\sum_{j \in C_f} x''_{j,f} r_j$ on f under x'' is at most the total load $\sum_{j \in C_f} x'_{j,f} r_j$ on f under x', plus the load $r_{j(f)}$ of at most one extra job $j(f) \in C_f$, i.e.,*

$$\sum_{j \in C_f} x''_{j,f} r_j \leq r_{j(f)} + \sum_{j \in C_f} x'_{j,f} r_j.$$

Lemma 3.7 allows us to round the fractional assignment to an integer assignment, while adding revenue from a single client to each facility. However, to bound the total loss in this assignment to prove Lemma 3.6, one needs additional algebraic arguments, which are deferred to Subsection A.2.3.

With Lemma 3.6 in hand, we are ready to complete the proof of our main Theorem 3.4 that gives the bicriteria oracle for FSFL:

Proof of Theorem 3.4. Denote the optimal (integral) δ-subsidized solution as (x^*, y^*) with objective value $g(\tau^*)$. Since (x, y) is the optimal fractional solution, (1) it must be δ-subsidized and (2) the objective value $g(\tau) \leq g(\tau^*)$, where $\tau \in \mathbb{R}^C$ is the vector of client distances for (x, y).

By Lemma 3.2.1, the solution $(\overline{x}, \overline{y})$ returned by algorithm α-`PointRounding` is $\left(4 \max\left(1, \frac{1}{\delta}\right) \tau\right)$-close. The solutions (x', y') and (x'', y') that give integral facilities and integral assignment respectively are both $\left(5 \times 4 \cdot \max\left(1, \frac{1}{\delta}\right) \tau\right)$-close by Lemma 3.4 and Lemma 3.6 respectively. Therefore, since g is sublinear, for distance vector τ'' for integral solution (x'', y'),

$$g(\tau'') = O\left(\max\left(1, \frac{1}{\delta}\right)\right) g(\tau) = O\left(\max\left(1, \frac{1}{\delta}\right)\right) g(\tau^*).$$

The solution $(\overline{x}, \overline{y})$ returned by α-`PointRounding` is 2δ-subsidized by Lemma 3.2.2. The solutions (x', y') returned by `RoundToIntegralFacilities` is also 2δ-subsidized

by Lemma 3.5, and the solution (x'', y') is $(2\delta + \theta)$-subsidized by Lemma 3.6. □

To summarize, we presented a bicriteria oracle for approximating the facility location with subsidies, by step-wise rounding the fractional optimal solution while ensuring that the total losses are bounded by a fraction of the revenue and that the client distances do not blow up by too much.

3.5.3 Portfolios for Fair Subsidized Facility Location

In this section, we obtain polynomial-time portfolios for FSFL for objectives in $\mathbf{C} = L_p$ norms of group distances. Using Theorem 3.2, we can reduce the problem of obtaining portfolios to the problem of designing approximation oracles, and using Theorem 3.4, we get a bicriteria oracle for FSFL, as discussed in the previous section. Putting these together, we get:

Corollary 3.1. *There exists a polynomial-time algorithm that given an instance of Fair Subsidized Facility Location (FSFL) with d client groups that satisfies θ-small revenues assumption and a subsidy parameter $\delta > 0$, obtains a portfolio P of size $O\left(\log\left(\frac{d}{\min(1,\delta)}\right)\right)$ such that*

1. *each solution in the portfolio is $(2\delta + \theta)$-subsidized, and*
2. *for each $p \geq 1$, there is a solution in the portfolio with objective value within factor $O\left(\max\left(1, \frac{1}{\delta}\right)\right)$ of the optimum δ-subsidized solution for the L_p norm objective.*

Note that the portfolio size grows logarithmically in the number of client groups d. Therefore, even with intersectional groups, the number of solutions in the portfolio grows slowly. For example, if we have 2 categories for urbanization, 2 categories for income levels, and 4 categories for geographical region, this still results in only $d = 16$ groups. Similarly, for $\delta < 1$, the dependence on $1/\delta$ is logarithmic, and for $\delta \geq 1$ the portfolio size is independent of δ. Further, these are theoretical worst-case bounds, and in practice, the portfolio sizes are much smaller (see Section 3.6).

Next, we present a lower bound for portfolios for FSFL, which shows that our results are order-optimal. That is, one cannot hope to construct a smaller $O(1)$-approximate portfolio for this problem.

Theorem 3.5. *There exist instances of FSFL with $\delta > 1$ where any $O(1)$-approximate portfolio for the class $\mathbf{C}$ of L_p norms of group distances with d client groups must have size $\Omega(\log d)$.*

The main idea of the proof is as follows: for all large enough d and for all constants $\alpha > 1$, we give an instance of FSFL where any α-approximate portfolio size must be $\simeq \log_{2\alpha} d$. Specifically, we construct an instance with $\simeq \log_{2\alpha} d$ distinct feasible solutions. Further, we give a set of $\simeq \log_{2\alpha} d$ norms such that for each fixed L_p norm in this set, exactly one of the $\simeq \log_{2\alpha} d$ solutions is optimal and every other solution is not an α-approximation. Thus, any portfolio for all L_p norms must contain each of the $\simeq \log_{2\alpha} d$ solutions. We formalize the proof now.

Proof. For all large enough d and for all constant $\alpha > 1$, we give an instance of FSFL where any α-approximate portfolio size must be $\frac{1}{4}\log_{2\alpha} d = \Omega(\log_{2\alpha} d)$. Fix d. Each client will be in their own unique group, so the number of clients $|C| = d$. Denote $\gamma = 2\alpha$, and denote N to be the highest integer that satisfies $1 + N(\gamma^{2N} + 1) = d$; then $N \geq \frac{1}{4}\log_{\gamma} d$.

Our metric space is a star graph with central vertex a_0 and leaf vertices $a_1, \ldots, a_N$ with unit distances between a_0 and a_i for all $i \in [N]$. A facility can be opened at any vertex (including the central vertex a_0) with operating cost $c = 1$.

The clients are specified as follows. While each client is in their own unique group, the group distances will be weighted, i.e., client in group $s \in [d]$ will have a weight μ_s so that the L_p norm objective for traveled distances $\tau_s, s \in [d]$ is $\left(\sum_{s\in[d]}(\mu_s\tau_s)^p\right)^{1/p}$.

- There is a single client at a_0 with weight γ^{N^2} and revenue 0.
- At a_i, there are $\gamma^{2N} + 1$ clients, each with revenue $r = \frac{1}{\gamma^{2N}+1}$. The first client has weight γ^i while the other γ^{2N} clients have weight γ^{-i} each.

Set subsidy $\delta = \frac{1}{2N}$. Note that since the sum of revenues is $r \times (\gamma^{2N} + 1) \times N = N$ and since each of the $N + 1$ facilities has an operating cost $c = 1$, we can open at most N facilities, i.e., a facility must not be open at some $a_i, i \in [0, N]$, and all clients at a_i must travel a unit distance to some other open facility.

If no facility is open at a_0, the weighted group distance vector is $\tau^{(0)} = (\gamma^{N^2}, 0, \ldots, 0)$. If a facility is not open at a_i for $i \in [N]$, the weighted group distance vector is $\tau^{(i)} = (\gamma^i, \underbrace{\gamma^{-i}, \ldots, \gamma^{-i}}_{\gamma^{2N}}, 0, \ldots, 0)$. For any $p \geq 1$, the L_p norm of these vectors is

$$\|\tau^{(i)}\|_p = \begin{cases} \gamma^{N^2} & \text{if } i = 0, \\ (\gamma^{ip} + \gamma^{2N-ip})^{1/p} & \text{if } i \in [N]. \end{cases}$$

Consider $p = N/j$ for $j \in \{1, \ldots, N\}$. Then $\|\tau^{(i)}\|_p$ is minimized at $i = j$ with value $2^{j/N} \cdot \gamma^j \leq 2 \cdot \gamma^j$. For all $i > j$, the first term dominates, and $\|\tau^{(i)}\|_p > \gamma^i \geq (2\alpha) \cdot \gamma^j = \alpha \cdot (2\gamma^j)$. For all $i < j$, the second term dominates, and $\|\tau^{(i)}\|_p > \gamma^{2N-i} \geq 2\alpha \cdot \gamma^j$. That is, the only α-approximate solution for the L_p norm objective is to not open the facility at a_j.

Thus, any α-approximate portfolio for L_p norms $\left\{\frac{N}{1}, \frac{N}{2}, \ldots, \frac{N}{N}\right\}$ must contain N distinct solutions. Since $N = \Omega(\log_{2\alpha} d)$, the result follows. □

3.6 Experiments

We now present our experiments on U.S. Census and pharmacy data in the state of Mississippi. For different values of the subsidy parameter δ, our goal is to present a portfolio of solutions to open 10 new pharmacies in Mississippi to help tackle the issue of medical deserts in an equitable way. Recall that (see Figure 3.1) we define a medical desert as a U.S. Census blockgroup[6] with over 20% below the poverty line and at a distance of over 2 miles (urban areas) or 10 miles (rural areas) from its nearest CVS/Walgreens/Walmart

[6]A U.S. Census blockgroup is a small administrative region with between 500 and 3000 people. It is the smallest unit for which U.S. Census data is publicly available.

pharmacy.[7] Focusing on these three largest pharmacy chains allows us to understand the impact of opening multiple facilities at scale.

As Figure 3.1 notes, 348 of the 2445 blockgroups in Mississippi are medical deserts, and they disproportionately affect the majority Black or African American population. Using our integer programming formulation (Equation IP) for FSFL and our portfolio algorithm for L_p norms from Section 3.4, we will give a portfolio of solutions for various values of the subsidy parameter δ. Each solution in the portfolio recommends the locations of 10 new pharmacies in Mississippi in addition to the 206 existing facilities. The total loss of these new pharmacies is bounded by a fraction δ of the total revenue of all clients.

We choose $\delta \in \{0.005, 0.01, 0.02, 0.05\}$ to allow the losses to span from 0.5% to 5%. Each blockgroup is a client, and a facility can be opened at any blockgroup location, where the location of a blockgroup is taken to be its geometric center. Further details on our modeling choices can be found at the end of this section.

We classify each blockgroup into $4 \times 2 \times 2 = 16$ groups based on (1) which of the 4 Congressional districts in Mississippi it lies in, (2) whether the blockgroup is rural or urban, and (3) whether or not 20% of the people in the blockgroup are below the poverty line. The group distance for a group is defined as the average distance of blockgroups to facilities, weighted by the population of the blockgroup. Distances of urban groups are weighted five times as much as rural groups to account for lesser access to vehicles. Given $p \geq 1$, the L_p norm objective minimizes the L_p norm of the 16-dimensional group distance vector.

We set the approximation parameter ε in Theorem 3.2 to 0.15, so that we are guaranteed a 1.15-approximate portfolio for all L_p norm objectives since we obtain exact solutions to the Equation IP.

Results. Figure 3.5 shows the portfolio for each value of $\delta \in \{0.005, 0.01, 0.02, 0.05\}$. For example, the portfolio for $\delta = 0.02$ has four solutions corresponding to $p = 1, 5.4, 13.5,$

[7] We choose a higher distance threshold for rural areas to account for better access to vehicles. This is in line with the methodology of the U.S. government to document food deserts [107].

Table 3.3: Reduction in the number of medical deserts for different solutions in the portfolio for subsidy $\delta = 0.02$ for various groups in Mississippi. A medical desert is a blockgroup with $\geq 20\%$ poverty rate and over n miles further away from the nearest pharmacy chain, where $n = 2$ miles for urban areas and $n = 10$ miles for rural areas.

Group		Existing Medical Deserts	Number of medical deserts reduced			
Urban/ Rural	District:		L_1 Norm	$L_{5.4}$ Norm	$L_{13.5}$ Norm	L_∞ Norm
Rural	1	41	0	0	0	0
	2	109	16	22	**23**	**23**
	3	98	19	19	**20**	19
	4	26	**6**	5	0	0
Urban	1	6	0	0	0	0
	2	31	**4**	0	0	**4**
	3	26	**1**	**1**	0	0
	4	11	**5**	0	0	0
Total		348	51	47	43	46

and ∞ arranged in the third column. We further expand on our results for this portfolio.

As Table 3.3 shows, *each solution in this portfolio for $\delta = 0.02$ reduces between 43 and 51 medical deserts out of 348 medical deserts*, despite adding only 10 facilities each to the existing 206 facilities. Further, 70-80% of these $\sim$50 blockgroups are African American for each solution, thus significantly reducing the disproportionate impact of medical deserts on the Black population.

The portfolio solutions are also significantly diverse across the 16 groups. As Table 3.1, Table 3.3, and Figure 3.5 show, *different solutions are optimal for different groups in terms of reduction of distances or medical deserts, with each solution performing well for different groups*. Each solution has its own strength: the L_1 norm solution reduces the most number of medical deserts, the $L_{5.4}$ norm solution leads to the smallest average distance traveled, and the L_∞ norm solution is the most equitable.

Further details on modeling choices. Since the U.S. Census blockgroup is the most granular level at which Census data is provided, we assume that all people in each Census blockgroup are located at the geometric center of the blockgroup. All our computations

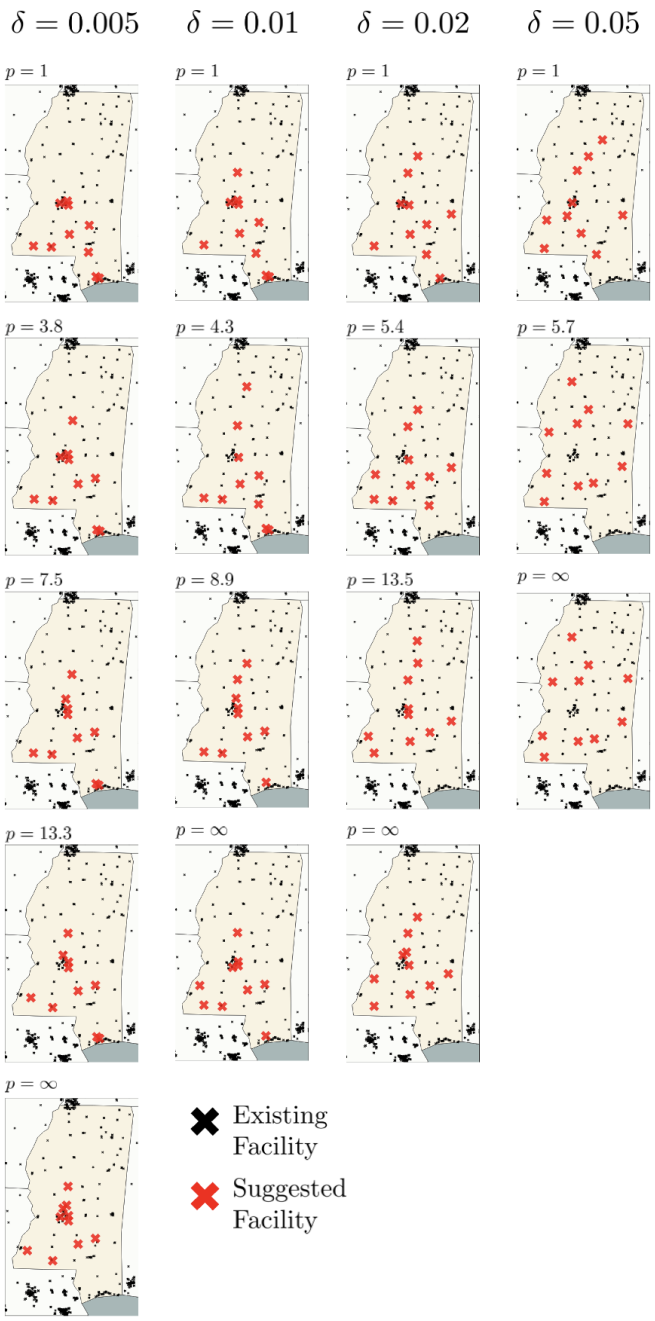


Figure 3.5: Portfolios of suggested locations for $k = 10$ new pharmacies using the FSFL model in the state of Mississippi, USA, in addition to existing CVS, Walmart, and Walgreens pharmacies. Each column shows the portfolio for a given subsidy parameter $\delta \in \{0.005, 0.01, 0.02, 0.05\}$ for approximation factor $1 + \varepsilon = 1.15$. While different solutions recommend opening facilities in different locations, all solutions significantly reduce the number of medical deserts.

assume straight-line distances. We assume that a new facility can be located at the center of any blockgroup. A facility cannot be opened outside Mississippi, although we allow people to travel to existing facilities outside Mississippi if those are closer. For simplicity, all operating costs are uniformly set to $\$2500$ while client revenues are set to $\$0.10$ for each person above the poverty line and $\$0.05$ for each person below the poverty line. These parameters can be varied in our model.

3.7 k-CLUSTERING and UNCAPACITATEDFACILITYLOCATION

In this section, we show that the k-CLUSTERING and UNCAPACITATEDFACILITYLOCATION (UFL) problems are special cases of the Fair Subsidized Facility Location (FSFL) problem.

The k-CLUSTERING problem is similar to FSFL except we are only given the metric space $(C \cup F, \texttt{dist})$, the objective g, and a bound k on the number of facilities as input and must return a solution (F', Π) with $|F'| \leq k$ that minimizes g. Examples of k-CLUSTERING include the k-median problem, where g is the sum of client distances to corresponding assigned facilities, the k-mean problem, where g is the L_2 norm of the client distances, and the k-center problem, where g is the maximum client distance.

In UFL, we are only given the metric space $(C \cup F, \texttt{dist})$, operating costs $c_f > 0$ for $f \in F$, and the distance objective function g. Given a solution (F', Π) with client distances $\tau_j, j \in C$, its objective value is the sum $g(\tau) + \sum_{f \in F'} c_f$ of operating costs of open facilities and the distance objective function value.

We reduce both these problems to FSFL:

Theorem 3.6. *k-CLUSTERING and UFL are special cases of FSFL.*

Proof. We reduce the following *budgeted facility location problem* to FSFL; it is standard to reduce both k-CLUSTERING and UFL to this budgeted problem. In this budgeted problem, we are given (i) the metric space $(C \cup F, \texttt{dist})$, (ii) operating costs $c_f > 0$ for $f \in F$,

(iii) a budget B on operating cost of open facilities, and (iv) the distance objective function g. A solution (F', Π) is feasible if $\sum_{f \in F'} c_f \leq B$ and its objective value is $g(\tau)$ where τ is the vector of client distances to assigned facilities. Clearly, k-CLUSTERING is a special case of this problem with $c_f = 1$ for all f and $B = k$. UFL can be reduced to this problem by 'guessing' the operating cost B of open facilities in the optimal solution and minimizing g while fixing this operating cost.

Given an instance of the budgeted problem, assume (after possibly scaling operating costs, budget B, and distances $\texttt{dist}$) without loss of generality that $\min_{f \in F} c_f = 1$. We construct an instance of FSFL as follows: metric space $(C \cup F, \texttt{dist})$, operating costs c, and distance objective function g stay the same. We define revenue $r_j := \frac{1}{|C|}$ for each client $j \in C$, and set $\delta = B - 1$.

First, we show that a feasible solution (F', Π) to the budgeted problem is also feasible for the FSFL instance. It is sufficient to show that it is δ-subsidized, that is, the total loss of unprofitable facilities is at most δ times the total revenue $\sum_{j \in C} r_j$ of all clients. Note that since $\min_f c_f = 1$ and $\sum_{j \in C} r_j = |C| \times \frac{1}{|C|} = 1$, we get that all open facilities are unprofitable, so that $\ell_f = c_f - \sum_{j:\Pi(j)=f} r_j$ for all $f \in F'$. Therefore, the total loss of facilities is

$$\sum_{f \in F'} \ell_f = \sum_{f \in F} \left(c_f - \sum_{j:\Pi(j)=f} r_j \right) = \left(\sum_{f \in F} c_f \right) - \sum_{j \in C} r_j \leq B - 1 = \delta = \delta \sum_{j \in C} r_j,$$

where the inequality follows since (F', Π) is feasible for the budgeted problem.

Conversely, we show that a feasible solution (F', Π) to FSFL is feasible for the budgeted problem: as before, all open facilities are unprofitable, so that

$$\begin{aligned} B - 1 = \delta = \delta \sum_{j \in C} r_j \geq \sum_{f \in F'} \ell_f &= \sum_{f \in F} \left(c_f - \sum_{j:\Pi(j)=f} r_j \right) \\ &= \left(\sum_{f \in F} c_f \right) - \sum_{j \in C} r_j = \left(\sum_{f \in F} c_f \right) - 1. \end{aligned}$$

Therefore, $\sum_{f \in F'} c_f \leq B$. Observing that the objective values stay exactly the same finishes the proof. □

3.8 Conclusion

We gave trade-offs between portfolio size and approximation quality for the classes of (1) conic combinations of given base functions and (2) functions that interpolate monotonically between the sum and max of the base functions. In particular, for the latter, we guarantee a portfolio size at most logarithmic in the number of base functions. Our algorithms reduce the problem of finding portfolios to the problem of designing approximation algorithms for fixed objectives.

As a concrete application, we proposed the Fair Subsidized Facility Location problem motivated by the crisis of pharmacy deserts. Our model is a generalization of the classical facility location problems and allows for richer clients to contribute to new facilities in underserved areas through a 'subsidy' constraint. It also allows the losses to be controlled while simultaneously being fair to different groups of people. Our approximation algorithm introduces a new combinatorial subroutine to round fractional solutions to get an integral set of facilities. However, it remains open if the dependence of the approximation ratio on $1/\delta$ is necessary.

We used our FSFL model to propose portfolios for opening new pharmacies in Mississippi, USA, with the aim of reducing medical deserts and average distances to the nearest pharmacy. Solutions in our portfolio not only significantly reduce the number of medical deserts using only 10 new facilities each, they are also diverse in their effects on people living in different areas. Thus, a policymaker can evaluate these solutions on multiple axes of interest and weigh their relative trade-offs before making an informed decision, as opposed to relying on the algorithm designer for a single good solution. Further, these options may also help constituents in different districts to mobilize political support for facilities opening closer to them.

CHAPTER 4
ORDERED NORMS, SYMMETRIC MONOTONIC NORMS, AND COVERING POLYHEDRA

4.1 Introduction

In this chapter, we study portfolios for ordered norms and symmetric monotonic norms. We will look at general bounds on portfolio sizes for these norm classes, and specifically for certain scheduling and covering problems. We begin by recalling the definitions of top-ℓ norms, ordered norms, and symmetric monotonic norms:

Top-ℓ norms [9, 37]. Recall (Definition 2.1) that the top-ℓ norm of a vector $x \in \mathbb{R}^d$ is the sum of the ℓ highest coordinates of ℓ by absolute value. Top-ℓ norms generalize the L_1 and L_∞ norms.

Ordered norms [20, 38]. Given a non-zero *weight vector* $w \in \mathbb{R}^d_{\geq 0}$ with nonincreasing weights $w_1 \geq \cdots \geq w_d \geq 0$, the ordered norm of $x \in \mathbb{R}^d_{\geq 0}$ is the weighted sum of coordinates of x with the kth highest coordinate of x weighted by the kth highest weight w_k. Ordered norms generalize top-ℓ norms (choose $w_1 = \ldots = w_\ell = 1$ and $w_{\ell+1} = \ldots = w_d = 0$, i.e., $w = \mathbf{1}_\ell$) and have a natural fairness interpretation of minimizing the cost of the most burdened individuals when x is a vector of individual costs.

Symmetric monotonic norms [8, 9, 23, 20]. A norm is symmetric monotonic if it is (i) invariant to the permutation of coordinates and (ii) nondecreasing in each coordinate. L_p norms, top-ℓ norms, and ordered norms are all symmetric monotonic norms.

This chapter is based on joint work with Swati Gupta and Mohit Singh. A preliminary version appeared in the *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA) 2025* [108]. An extended version is currently under submission.

Ordered norms are fundamental to symmetric monotonic norms in two aspects: each symmetric monotonic norm is (1) $O(\log d)$-approximated by some ordered norm [38], and (2) the supremum of some set of ordered norms [20].

For top-ℓ norms, Goel and Meyerson [9] essentially obtain a $(1+\varepsilon)$-approximate portfolio of size $O\left(\frac{\log d}{\varepsilon}\right)$ for all $\varepsilon \in (0, 1]$. We generalized this bound to all classes of functions that interpolate monotonically between L_1 and L_∞ norms in Chapter 3.

However, for ordered norms, only a general construction of $\mathrm{poly}(d^{1/\varepsilon})$-sized $(1+\varepsilon)$-approximate portfolios was known, due to Chakrabarty and Swamy [20]. No bound was known for symmetric monotonic norms. We observe that their result generalizes to symmetric monotonic norms (Lemma 4.5). It was also known that a solution that is simultaneously α-approximate for all top-ℓ norms is, in fact, simultaneously α-approximate for all symmetric monotonic norms [9]. *This property is no longer true for portfolios of size greater than 1* (e.g., see our Example 2.1, Lemma A.2, Theorem 4.3, or Theorem 4.4). In particular, we show that the approximation ratio of a portfolio for top-ℓ norms and ordered norms can differ by a factor polynomial in d. Consequently, we cannot restrict to constructing portfolios only for top-ℓ norms and need new techniques for the much larger sets of ordered norms and symmetric monotonic norms. We show that there exist sets $\mathcal{D}$ and objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ for which the portfolio size must be $d^{\Omega(1/\log\log d)}$ (i.e., nearly polynomial in d) for ordered and symmetric monotonic norms even for approximation factor as large as $O(\log d)$ (Theorem 4.3).

4.1.1 Setting

As before, we are given some domain or set $\mathcal{D}$ of feasible solutions and base objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$. Given some norm $\|\cdot\|$ on $\mathbb{R}^d$, we can seek to minimize $\|\mathbf{h}(x)\|$ over $x \in \mathcal{D}$. In this chapter, we will be concerned with the norms $\|\cdot\|$ in the class $\mathbf{C}$ of ordered norms or symmetric monotonic norms on $\mathbb{R}^d$. At various places in this chapter, we will have $\mathcal{D} \subseteq \mathbb{R}^d_{\geq 0}$ and base objective function $h_i(x) = x_i$ for all $i \in [d]$, so that $\mathbf{h}(x) = x$;

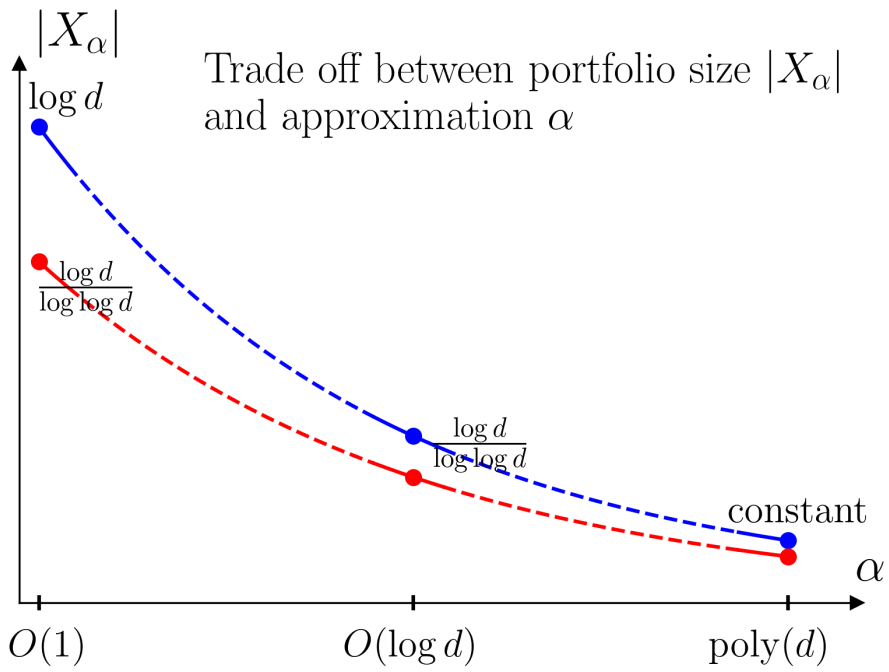


Figure 4.1: A qualitative plot to illustrate the trade-off between approximation α and the smallest portfolio size $|X_\alpha|$ for the MACHINELOADSIDENTICALJOBS problem for ordered norms. The worst-case lower bound $|X_\alpha| = \Omega\left(\frac{\log d}{\log \alpha + \log\log d}\right)$ is illustrated in red, and the upper bound $|X_\alpha| = O\left(\frac{\log d}{\log(\alpha/4)}\right)$ is illustrated in blue. The two bounds converge for $\alpha = \Omega(\log d)$.

in these cases, we will omit the base functions $h_1, \ldots, h_d$.

4.1.2 Contributions and Techniques

First, we show (Lemma 4.5) that symmetric monotonic norms – like ordered norms – always admit a portfolio of size $\text{poly}(d^{1/\varepsilon})$ for arbitrary $\mathcal{D}$ and base functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$. Further, we give sets $\mathcal{D}$ and nonnegative base functions $h_1, \ldots, h_d$ on $\mathcal{D}$ where any $O(\log d)$-approximate portfolios for symmetric monotonic norms or ordered norms have size $d^{\Omega(1/\log\log d)}$.

To obtain smaller portfolios for covering and scheduling problems, we develop a general framework called `OrderAndCount`. We obtain portfolios of size $\text{polylog}(d)$ using this framework, which is an exponential improvement over the general size bound (also see

Table 4.1). Specifically, we consider the following two settings:

Characterizing trade-off for MACHINELOADSIDENTICALJOBS. As our first result, we consider the MACHINELOADSIDENTICALJOBS (MLIJ) problem where n identical jobs must be scheduled on d unidentical machines. The goal is to minimize some norm of the vector of machine loads. This is a simple model for workload distribution among d workers (corresponding to machines) with different processing speeds, and various norms correspond to various fairness criteria for fair distribution of jobs. We prove the following result:

Theorem 4.1. *There is a polynomial-time algorithm that, given any instance of the* MACHINELOADSIDENTICALJOBS *problem with d machines and any $\alpha > 4$, finds a portfolio X of size*

$$|X| = O\left(\frac{\log d}{\log(\alpha/4)}\right)$$

that is (i) α-approximate for ordered norms and (ii) $O(\alpha \log d)$-approximate for symmetric monotonic norms. Further, for all $\alpha > 1$, there exists a family of instances of MACHINELOADSIDENTICALJOBS *for which the size of any α-approximate portfolio for ordered norms is $\Omega\left(\frac{\log d}{\log \alpha + \log\log d}\right)$.*

Table 4.1: Approximations for ordered norms and symmetric monotonic norms in this chapter, for arbitrary $\varepsilon \in (0, 1]$.

Problem or set of feasible vectors $\mathcal{D}$	Worst-case approximation factor for simultaneous approximation	Guarantees for portfolio of size > 1		
		Size	Approximation for ordered norms	Approximation for symmetric monotonic norms
MACHINELOADSIDENTICALJOBS d machines (Theorem 4.1)	$\Omega(\sqrt{d})$	$O\left(\frac{\log d}{\varepsilon}\right)$	$4 + \varepsilon$	$O(\log d)$
COVERINGPOLYHEDRON with r constraints: $\{x \in \mathbb{R}^d_{\geq 0} : Ax \geq b\}$, $A \in \mathbb{R}^{r \times d}_{\geq 0}, b \in \mathbb{R}^r_{\geq 0}$ (Theorem 4.2)	$\Omega(\sqrt{d})$	$\left(\frac{\log(d/\varepsilon)}{\varepsilon}\right)^{O(r^2)}$	$1 + \varepsilon$	$O(\log d)$

The above result characterizes the trade-off between portfolio sizes and achievable approximation factors (up to $\log \log$ factor) for MLIJ (also see Figure 4.1). To obtain this result, we use our `OrderAndCount` approach, which exploits the fact that each ordered norm, while a convex function in general, is a linear function when restricted to a region where all vectors satisfy the same order of coordinate values. That is, if vector $x \in \mathbb{R}^d$ satisfies $x_{\pi(1)} \geq x_{\pi(2)} \geq \ldots \geq x_{\pi(d)} \geq 0$ for some order π on $[d]$, the ordered norm $\|x\|_{(w)}$ is the linear function $\sum_k w_k x_{\pi(k)}$. This gives the following algorithm to obtain portfolios for ordered norms: for each order π, we can restrict to the set $\mathcal{D}_\pi$ of vectors in $\mathcal{D}$ that satisfy order π, and collect the set of extreme points of $\mathcal{D}_\pi$. The union of these extreme points is then an optimal (i.e, 1-approximate) portfolio for ordered norms. In general, this results in exponentially many solutions even when $\mathcal{D}$ is a polytope, since there are exponentially many orders π and potentially exponentially many extreme points of each $\mathcal{D}_\pi$. We show that for MLIJ, (i) it suffices to restrict to a specific order π^* (that is determined by the problem instance), and that (ii) there are at most d extreme points of $\mathcal{D}_{\pi^*}$. This already reduces the portfolio size to d. Finally, we give a rounding algorithm to show that these d extreme points can further be α-approximated by a subset of $O(\log_{\alpha/4} d)$ *integral* points, which results in the desired portfolio.

Exponential improvement in portfolios for covering. Next, we consider the COVERINGPOLYHEDRON problem, where the set of feasible solutions is a polyhedron with r *covering constraints* of the form: $a^\top x \geq b$ (for $a \in \mathbb{R}^d_{\geq 0}$, and $b \in \mathbb{R}_{\geq 0}$) together with nonnegativity $x \geq 0$. This generalizes the MLIJ problem above, and models many natural scenarios for workload distribution. We seek to minimize various norms of the vector x subject to these covering constraints.

Many problems can be modeled as the covering polyhedron, for example, a fair centralized server that must balance the workload on d machines, each with r parallel processing units [109]. This load-balancing problem also appears in the context of volunteer-

dependent non-profit organizations, such as HIV social care centers, blood donation drives, food recovery organizations [110], etc. Numerous studies have been conducted on the reasons for the attrition of volunteers, and overburdening by the amount of demands placed on them is one of the key ones [111, 112].

This can be thought of as a scheduling problem with d machines and r different kinds of jobs, where each machine is capable of running different kinds of jobs in parallel. If b_j units for the jth job type need to be scheduled, and machine $i \in [d]$ has processing speed $A_{j,i}$ for the jth type of job, then the total loads $x_i, i \in [d]$ on the machines satisfy $\sum_{i\in[d]} A_{j,i}x_i \geq b_j$. For a given fairness criterion or norm $\|\cdot\|$ on $\mathbb{R}^d$, this translates to minimizing $\|x\|$ over the covering polyhedron $\{x \in \mathbb{R}^d : Ax \geq b, x \geq 0\}$.

The challenge in extending `OrderAndCount` to such problems is twofold: (i) bounding the number of possible orders that the optimal solution $x^* = \arg\min_x \|x\|_{(w)}$ for any ordered norm might satisfy, and then (ii) selecting a subset of corresponding extreme points for each order that must be included in the portfolio. For the first challenge, we develop a novel *primal-dual counting technique* which allows us to count the number of possible orders in an appropriate dual space that is structurally much simpler (Section 4.5). For the second challenge, we show that a sparsification procedure allows us to reduce the number of extreme points for each order. Together, using `OrderAndCount`, we give poly-logarithmic sized portfolios for COVERINGPOLYHEDRON for constant r:

Theorem 4.2. *For* COVERINGPOLYHEDRON *in d dimensions and r constraints, for any $\varepsilon \in (0, 1]$, there is a portfolio X of size*

$$|X| = O\big(\log(d/\varepsilon)/\varepsilon\big)^{3r^2-2r},$$

which is (i) $(1+\varepsilon)$-approximate for ordered norms, and (ii) $O(\log d)$-approximate for symmetric monotonic norms. There exists an algorithm to find this portfolio with running time that is polynomial in d and $(\log(d)/\varepsilon)^{r^2}$.

This result shows the following trade-off between ε and portfolio size $|X_{1+\varepsilon}|$: we have that $|X_{1+\varepsilon}|^{1/\Omega(r^2)} \times \varepsilon$ remains roughly a constant. If the number of constraints r is $o\left(\frac{\sqrt{\log d}}{\log\log d}\right)$, this gives an exponential improvement over the current best bound of $\mathrm{poly}(d^{1/\varepsilon})$ [20].

4.2 Preliminaries

We give useful preliminary results in this section. Omitted proofs are included in Section B.1. Our first lemma shows that portfolios can be composed:

Lemma 4.1 (Portfolio composition)**.** *Given class* $\mathbf{C}$ *of functions over set* $\mathcal{D}$ *of feasible solutions,*

1. *If* X_1 *is an* α_1*-approximate portfolio for* $\mathbf{C}$ *over* $\mathcal{D}$ *and* X_2 *is an* α_2*-approximate portfolio for* $\mathbf{C}$ *over* X_1*, then* X_2 *is an* $\alpha_1\alpha_2$*-approximate portfolio for* $\mathbf{C}$ *over* $\mathcal{D}$*.*
2. *Suppose* $\mathcal{D} = \bigcup_{i\in[n]} \mathcal{D}_i$ *and* X_i *is an* α*-approximate portfolio for* $\mathbf{C}$ *over* $\mathcal{D}_i$ *for each* $i \in [n]$*. Then* $\bigcup_{i\in[n]} X_i$ *is an* α*-approximate portfolio for* $\mathbf{C}$ *over* $\mathcal{D}$*.*

Next, it is known that any symmetric monotonic norm $\|\cdot\|$ on $\mathbb{R}^d$ can be $O(\log d)$-approximated by an ordered norm on $\mathbb{R}^d$ [38]. Consequently, the same bound also holds for portfolio approximations:

Lemma 4.2. *For any* $\mathcal{D}$*, an* α*-approximate portfolio* $X \subseteq \mathcal{D}$ *for ordered norms is an* $O(\alpha \log d)$*-approximate portfolio for symmetric monotonic norms.*

Finally, we characterize the class of duals to ordered norms and state the corresponding Cauchy-Schwarz inequality, which will be used in our `OrderAndCount` framework. An order π on a finite set X is defined as a bijection between X and $\{1, \ldots, |X|\}$; for simplicity we denote the set of all orders on X as $\mathrm{Perm}(X)$ or as $\mathrm{Perm}(d)$ when $X = [d]$. We say that a vector $x \in \mathbb{R}^d_{\geq 0}$ *satisfies* an order $\pi \in \mathrm{Perm}(d)$ if $x_{\pi(1)} \geq \cdots \geq x_{\pi(d)}$. Recall that for a vector $x \in \mathbb{R}^d$, we denote $x^{\downarrow}$ as the vector with coordinates of x sorted in decreasing order. We also denote $\mathbf{1}_k \in \mathbb{R}^d$ as the vector with k ones followed by zeros.

Lemma 4.3 (Dual ordered norms)**.** *Given a weight vector $w \in \mathbb{R}^d$, the dual norm $\|\cdot\|^*_{(w)}$ to ordered norm $\|\cdot\|_{(w)}$ is given by*

$$\|y\|^*_{(w)} = \max_{k\in[d]} \frac{\|y\|_{(\mathbf{1}_k)}}{\|w\|_{(\mathbf{1}_k)}}.$$

Lemma 4.4 (Ordered Cauchy-Schwarz)**.** *For all $x, y \in \mathbb{R}^d_{\geq 0}$, $\|x\|_{(w)}\|y\|^*_{(w)} \geq x^\top y$. Further, equality holds if and only if*

1. *there is some order $\pi \in \mathrm{Perm}(d)$ such that x, y both satisfy π.*
2. *for each $k \in [d]$ either $x^\downarrow_k = x^\downarrow_{k+1}$ or $\frac{\|y\|_{(\mathbf{1}_k)}}{\|w\|_{(\mathbf{1}_k)}} = \|y\|^*_{(w)}$.*

4.3 General Bounds on Portfolio Size

In this section, we give general upper and lower bounds on portfolio sizes for ordered and symmetric monotonic norms. Omitted proofs are included in Section B.1.

[20] show that the best-known upper bound on the size of a $(1+\varepsilon)$-approximate portfolio for ordered norms is polynomial in $d^{1/\varepsilon}$. We generalize their bound to symmetric monotonic norms:

Lemma 4.5. *For any feasible set $\mathcal{D}$, base functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$, and $\varepsilon \in (0, 1]$, there is a $(1+\varepsilon)$-approximate portfolio of size $\mathrm{poly}(d^{1/\varepsilon})$ for symmetric monotonic norms.*

Further, we show that this bound is nearly tight for ordered norms and symmetric monotonic norms. Specifically, there exist sets $\mathcal{D}$ and base functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ where any α-approximate portfolios must have size $d^{1/\Omega(\log\log d)}$ even for approximation α as large as $O(\log d)$:

Theorem 4.3. *There exist sets $\mathcal{D}$ and base functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ such that any $O(\log d)$-approximate portfolio for ordered norms must have size $d^{\Omega(1/\log\log d)}$. The same bound holds for symmetric monotonic norms.*

As noted previously, this is another proof that α-approximate portfolios for top-ℓ norms are not necessarily α-approximate portfolios for larger classes such as ordered norms or symmetric monotonic norms. Indeed, the size of the smallest $O(1)$-approximate portfolio for top-ℓ norms for base functions $h_1, \ldots, h_d$ is $O(\log d)$. As the above result shows, the minimum portfolio size for $O(1)$-approximate portfolios for ordered or symmetric monotonic norms can be $d^{\Omega(1/\log\log d)}$, since an $O(\log d)$-approximate portfolio is alos an $O(1)$-approximate portfolio.

4.4 `OrderAndCount` for MACHINELOADSIDENTICALJOBS

In this section, we introduce the `OrderAndCount` framework and prove Theorem 4.1 for the MLIJ problem. Recall that we seek to assign n copies of a job among d machines with different processing times $p_i, i \in [d]$. This is the simplest model for workload distribution where some tasks must be distributed among individuals in a workplace: processors correspond to individuals, processing times represent their efficiencies, and balancing loads on machines corresponds to managing the workloads of the individuals. Given a norm $\|\cdot\|$ on $\mathbb{R}^d$, the goal is to schedule the jobs to minimize the norm of the vector of machine loads. We seek a portfolio of solutions (i.e., schedules) for ordered norms and symmetric monotonic norms.

To see why a single solution may be suboptimal, we observe a simple example where no solution is a simultaneous $o(\sqrt{d})$-approximation: suppose there are $n = d$ jobs and $p_1 = 1$ while $p_2 = \cdots = p_d = \sqrt{d}$. The optimal solution for the L_∞ norm (i.e., maximum load) minimization assigns one job per machine to get maximum load $\sqrt{d}$. The optimal solution for the L_1 norm (i.e., total load) minimization assigns all jobs to the most efficient machine, i.e., machine 1, for a total load of d. Therefore, any assignment with $< d/2$ jobs on machine 1 is an $\Omega(\sqrt{d})$-approximation for L_1 norm, and any assignment with $\geq d/2$ jobs on machine 1 is an $\Omega(\sqrt{d})$-approximation for L_∞ norm. This motivates us to increase the portfolio size.

In Subsection 4.4.1, we prove the upper bound on portfolio size in Theorem 4.1, guaranteeing for each $\alpha > 4$ a portfolio of size $O\left(\frac{\log d}{\log(\alpha/4)}\right)$ that is α-approximate for ordered norms and $O(\alpha \log d)$-approximate for symmetric monotonic norms. We prove the lower bound showing that any α-approximate portfolio for ordered norms must have size $\Omega\left(\frac{\log d}{\log \alpha + \log\log d}\right)$ in Subsection 4.4.2. We will also prove (Theorem 4.4) that there are instances of MLIJ with an optimal portfolio of size 2 for top-ℓ norms, but where any $O(1)$-approximate portfolio for ordered norm must have size $\Omega\left(\frac{\log d}{\log\log d}\right)$.

We start with some notation. Since all jobs are identical, we can identify a schedule by the number of jobs on each machine. If $n_i \in \mathbb{Z}_{\geq 0}$ jobs are scheduled on machine i, then $\sum_{i\in[d]} n_i = n$. The corresponding load vector is $x = x(n) = (n_1 p_1, \ldots, n_d p_d)$. Therefore, the set of feasible solutions is $\mathcal{D} = \{x \in \mathbb{R}^d_{\geq 0} : x_i = n_i p_i \ \forall\ i \in [d], n_i \in \mathbb{Z}_{\geq 0} \ \forall\ i \in [d], \sum_i n_i = n\}$. We can relabel the machine indices and assume without loss of generality that $0 < p_1 \leq \cdots \leq p_d$. That is, machine 1 is the fastest, followed by machine 2, etc.

4.4.1 Portfolio Upper Bound

Our high-level plan is as follows: we show that special instances of MLIJ that we call *doubling instances* – those where each p_i is a power of 2 – satisfy two key properties: (i) any instance of MLIJ is 2-approximated by some doubling instance (Lemma 4.6), and (ii) the optimal solution x^{OPT} for any symmetric monotonic norm to a doubling instance satisfies $x_1^{\mathrm{OPT}} \geq x_2^{\mathrm{OPT}} \geq \cdots \geq x_d^{\mathrm{OPT}}$ (Lemma 4.7), i.e., must satisfy a specific order of coordinates. These inequalities allow us to relax the integrality constraints and consider the polyhedron $\mathcal{P} = \{x : \sum_i \frac{x_i}{p_i} = n; x_1 \geq \cdots \geq x_d \geq 0\}$ if fractional solutions, where the coordinate-wise inequality constraints can be put in for doubling instances. This sets up `OrderAndCount`: there is only one possible order for vectors $x \in \mathcal{P}$, which is $x_1 \geq \cdots \geq x_d \geq 0$. Therefore, each ordered norm $\|x\|_{(w)} = w^\top x$ is a linear function over $\mathcal{P}$, and so the set of vertices of $\mathcal{P}$ form an optimal portfolio of fractional solutions for ordered norms for the doubling instance and a 2-approximate portfolio for the original

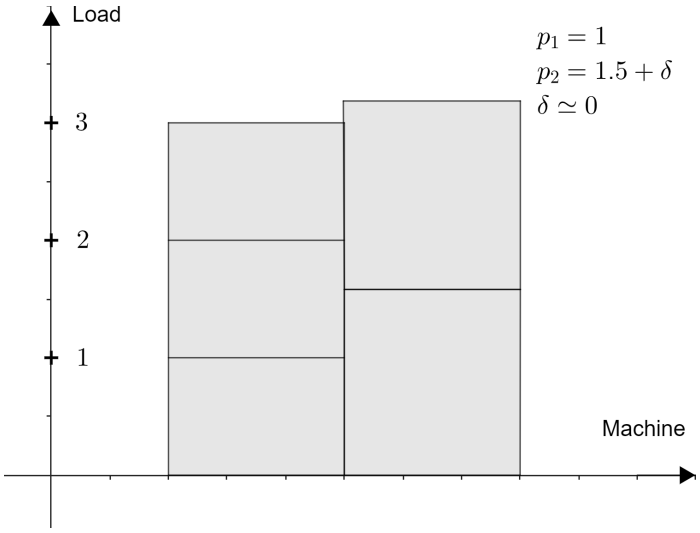


Figure 4.2: An example for makespan minimization with 2 machines and 5 jobs where $x_1^{\mathrm{OPT}} < x_2^{\mathrm{OPT}}$ for optimal load vector x^{OPT}.

instance. We show that we can restrict to $O(\log_{\alpha/4} d)$ of these vertices, losing factor $\alpha/4$. Finally, we lose another factor 2 in rounding fractional solutions to integral ones, to get an overall approximation factor α for ordered norms. By Lemma 4.2, this portfolio is $O(\alpha \log d)$-approximate for symmetric monotonic norms.

Lemma 4.6. *Given an instance of* MLIJ *with d machines and n copies of a job, there exists another instance of* MLIJ *with d machines and n jobs such that for any load vector x' for this modified instance, the corresponding load vector x for the original instance satisfies*

$$\frac{1}{\sqrt{2}} x \leq x' \leq \sqrt{2} x.$$

Proof. To construct the new instance, round each p_i to its closest power of 2, say p_i'. Then $\frac{1}{\sqrt{2}} p_i' \leq p_i \leq \sqrt{2} p_i'$. When n_i jobs are scheduled on processor i, corresponding load vectors $x = (n_1 p_1, \ldots, n_d p_d)$ and $x' = (n_1 p_1', \ldots, n_d p_d')$ are within factor $\sqrt{2}$ of each other. □

Corollary 4.1. *For ordered norms, an α-approximate portfolio for an instance of* MLIJ *can be obtained from a $\frac{\alpha}{2}$-approximate portfolio for the corresponding doubling instance.*

Here is the first main idea of `OrderAndCount`: we show next that for doubling in-

stances, optimal load vector x^{OPT} for any symmetric monotonic norm always satisfies the order $x_1^{\mathrm{OPT}} \geq \cdots \geq x_d^{\mathrm{OPT}}$. This is false if the instance is not doubling; see Figure 4.2.

Lemma 4.7. *Suppose x^{OPT} is the optimal load vector for some symmetric monotonic norm $\|\cdot\|$ for a doubling instance. Then, we can assume without loss of generality that $x_1^{\mathrm{OPT}} \geq x_2^{\mathrm{OPT}} \geq \cdots \geq x_d^{\mathrm{OPT}}$.*

Proof. Suppose $x_i^{\mathrm{OPT}} < x_{i+1}^{\mathrm{OPT}}$ for some i. Transfer one job from machine $i+1$ to machine i, to get the new load vector x defined as:

$$x_l = \begin{cases} x_l^{\mathrm{OPT}} & \text{if } l \neq i, i+1, \\ x_i^{\mathrm{OPT}} + p_i & \text{if } l = i, \\ x_{i+1}^{\mathrm{OPT}} - p_{i+1} & \text{if } l = i+1. \end{cases}$$

Since p_i divides p_{i+1} and $x_{i+1}^{\mathrm{OPT}} > x_i^{\mathrm{OPT}}$, we get that $x_{i+1}^{\mathrm{OPT}} - x_i^{\mathrm{OPT}} \geq p_i$. Therefore,

$$\max(x_i, x_{i+1}) = \max\left(x_i^{\mathrm{OPT}} + p_i, x_{i+1}^{\mathrm{OPT}} - p_{i+1}\right) \leq x_{i+1}^{\mathrm{OPT}} = \max\left(x_i^{\mathrm{OPT}}, x_{i+1}^{\mathrm{OPT}}\right).$$

Further, $x_i + x_{i+1} < x_i^{\mathrm{OPT}} + x_{i+1}^{\mathrm{OPT}}$. That is, $(x_i, x_{i+1}) \prec (x_i^{\mathrm{OPT}}, x_{i+1}^{\mathrm{OPT}})$. Since all other coordinates of x and x^{OPT} are equal, a simple inductive argument shows that $x \preceq x^{\mathrm{OPT}}$. Lemma 2.1 then implies that $\|x\| \leq \|x^{\mathrm{OPT}}\|$, finishing the proof. □

For the rest of this section, we restrict ourselves to doubling instances; we will give an $\alpha/2$-approximate portfolio of size $\leq 1 + \log_{\alpha/4} d$ for ordered norms for any doubling instance. For any weight vector w, Lemma 4.7 allows us to relax the integer program (Equation IP1) to a linear program: while not every load vector forms a feasible solution to Equation IP1, Lemma 4.7 shows that there is an optimal solution for the ordered norm that is feasible for this IP.

$$\min w^\top x \qquad \text{s.t.} \tag{IP1}$$

$$\sum_i \frac{x_i}{p_i} = n, \tag{4.1}$$

$$x_i \geq x_{i+1} \qquad \forall\, i \in [d-1], \tag{4.2}$$

$$\frac{x_i}{p_i} \in \mathbb{Z}_{\geq 0} \qquad \forall\, i \in [d], \tag{4.3}$$

$$\min w^\top x \qquad \text{s.t.} \tag{LP1}$$

$$\sum_i \frac{x_i}{p_i} = n, \tag{4.4}$$

$$x_i \geq x_{i+1} \qquad \forall\, i \in [d-1], \tag{4.5}$$

$$x \geq 0. \tag{4.6}$$

The next lemma characterizes the d vertices of the constraint polytope $\mathcal{P} := \{x : \sum_i \frac{x_i}{p_i} = n, x_1 \geq \cdots \geq x_d \geq 0\}$ of Equation LP1. We omit the straightforward proof.

Lemma 4.8. *For any weight vector w, the optimal solution x^* to Equation LP1 satisfies for some $l \in [d]$ that*

$$x_1^* = \cdots = x_l^* = \frac{n}{\sum_{i\in[l]} \frac{1}{p_i}}, \quad x_{l+1}^* = \cdots = x_d^* = 0.$$

For $l \in [d]$, denote the lth vertex as $x(l) := \frac{n}{\sum_{i\in[l]} \frac{1}{p_i}} \mathbf{1}_l$, with l non-zero entries. Call $x(l)$ *good* if

$$\frac{n}{\sum_{i\in[l]} \frac{1}{p_i}} \geq p_l, \tag{4.7}$$

i.e., if the value of each non-zero coordinate is at least the processing time corresponding to the last non-zero coordinate. Clearly, $x(1)$ is good since the number of jobs $n \geq 1$. Further, if $x(l)$ is good, then $x(l-1)$ is also good. The next lemma says that if $x(l)$ is good, then it

can be rounded to an integral load vector:

Lemma 4.9. *If $x(l)$ is good, then it can be rounded to $\hat{x}(l)$ that is feasible for Equation IP1 and $\frac{1}{2}x(l) \leq \hat{x}(l) \leq 2x(l)$.*

Proof. Denote $n_i = \frac{x(l)_i}{p_i}$ for all $i \in [d]$, then $n_{l+1} = \cdots = n_d = 0$ and $\sum_{i\in[d]} n_i = n$. Then one can assign either $\hat{n}_i = \lfloor n_i \rfloor$ or $\hat{n}_i = \lceil n_i \rceil$ jobs to machine $i \in [d]$, while ensuring that $\sum_{i\in[d]} \hat{n}_i = n$. The load on machine $i \in [d]$ in this new schedule is $\hat{x}(l)$, with $\hat{x}(l)_i = p_i\hat{n}_i$.

By definition of good vertices, and since $p_1 \leq \ldots \leq p_d$, we have $x(l)_i \geq p_l \geq p_i$ for each $i \in [l]$. Therefore, we get $n_i \geq 1$, thus implying $\frac{1}{2}n_i \leq \lfloor n_i \rfloor \leq n_i$ and $n_i \leq \lceil n_i \rceil \leq 2n_i$ for all $i \in [l]$. This implies $\frac{1}{2}n_i \leq \hat{n}_i \leq 2n_i$ for all $i \in [d]$. Since $n_i = \frac{x(l)_i}{p_i}$ and $\hat{n}_i = \frac{\hat{x}(l)_i}{p_i}$, we get the result. □

Let L be the largest index such that $x(L)$ is good. Our next lemma shows that rounding good vertices gives a 2-approximate portfolio for ordered norms:

Lemma 4.10. *$\{\hat{x}(1), \ldots, \hat{x}(L)\}$ is a 2-approximate portfolio for ordered norms for the doubling instance.*

Proof. Fix a weight vector w. Let x^{OPT} be the (integral) optimal load vector for $\|\cdot\|_{(w)}$, and let l be the largest index such that $x_l^{\mathrm{OPT}} > 0$. We will first show that there exists an index $l' \leq l$ such that (i) $x(l')$ is good, and (ii) $\|x(l')\|_{(w)} \leq \|x^{\mathrm{OPT}}\|_{(w)}$. Together with Lemma 4.9, this implies that $\|\hat{x}(l')\|_{(w)} \leq 2\|x^{\mathrm{OPT}}\|_{(w)}$, implying the result.

We note first that $x(l)$ is good: since x^{OPT} is integral and $x_l^{\mathrm{OPT}} \neq 0$, we have $x_l^{\mathrm{OPT}} \geq p_l$. From Lemma 4.7, we have $x_1^{\mathrm{OPT}} \geq \cdots \geq x_l^{\mathrm{OPT}} \geq p_l$. Since $\sum_{i\in[l]} \frac{x_i^{\mathrm{OPT}}}{p_i} = n$, we get $n \geq \sum_{i\in[l]} \frac{p_l}{p_i} = p_l \sum_{i\in[l]} \frac{1}{p_i}$. That is, $x(l)$ is good.

In particular, this implies that $x(l')$ is good for each $l' \leq l$, so it is now sufficient to show that there is some $l' \leq l$ such that $\|x(l')\|_{(w)} \leq \|x^{\mathrm{OPT}}\|_{(w)}$. Consider the following linear program:

$$\min w^\top x \qquad \text{s.t.} \qquad \text{(LP2)}$$

$$\sum_i \frac{x_i}{p_i} = n, \tag{4.8}$$

$$x_i \geq x_{i+1} \qquad \forall\, i \in [d-1], \tag{4.9}$$

$$x_{l+1} = \cdots = x_d = 0. \tag{4.10}$$

x^{OPT} is feasible for this LP by assumption. Further, by an argument similar to Lemma 4.8, we get that the vertices of the constraint polytope for this LP are $x(1), \ldots, x(l)$. Therefore, there is some $l' \leq l$ such that $\|x(l')\|_{(w)} = w^\top x(l') \leq w^\top x^{\mathrm{OPT}} = \|x^{\mathrm{OPT}}\|_{(w)}$, finishing the proof. □

We are ready to prove the upper bound in Theorem 4.1. We will convert the 2-approximate portfolios of size d for doubling instances to an $\alpha/2$-approximate portfolio of size $\sim \log_{\alpha/4} d$, which implies α-approximate portfolios of size $\sim \log_{\alpha/4} d$ for MLIJ by Corollary 4.1.

Proof of upper bound in Theorem 4.1. We claim that for all indices $l, i \in [d]$ such that $i \leq \frac{\alpha}{4} l$, we have $x(l) \preceq \frac{\alpha}{4} x(i)$. Therefore, $\|x(l)\|_{(w)} \leq \frac{\alpha}{4} \|x(i)\|_{(w)}$ for all ordered norms $\|\cdot\|_{(w)}$ from Lemma 2.1, implying that $\left\{x((\alpha/4)^j) : j \in [0, 1 + \log_{(\alpha/4)} L]\right\}$ is an $(\alpha/2)$-approximate portfolio over doubling instances.

Since $p_1 \leq \cdots \leq p_d$ and $i \leq \frac{\alpha}{4} l$, we have $\sum_{j \in [l]} \frac{1}{p_j} \geq \frac{4}{\alpha} \sum_{j \in [i]} \frac{1}{p_j}$. Therefore, for all $k \leq l$, we have

$$\sum_{j \in [k]} x(l)_j = \frac{kn}{\sum_{i \in [l]} \frac{1}{p_j}} \leq \frac{\alpha}{4} \cdot \frac{kn}{\sum_{j \in [i]} \frac{1}{p_j}} = \frac{\alpha}{4} \cdot \sum_{j \in [k]} x(i)_k.$$

Further, for $k > l$,

$$\sum_{j \in [k]} x(l)_j = \sum_{j \in [l]} x(l)_j = \frac{nl}{\sum_{j \in [l]} \frac{1}{p_j}} \leq \frac{\alpha}{4} \frac{nl}{\sum_{j \in [i]} \frac{1}{p_j}} \leq \frac{\alpha}{4} \sum_{j \in [i]} x(i)_j \leq \frac{\alpha}{4} \sum_{j \in [k]} x(i)_j.$$

Therefore, $x(l) \preceq (\alpha/4) x(i)$. This completes the proof. □

4.4.2 Portfolio Lower Bound

We prove the lower bound by giving appropriate doubling instances with d machines where any α-approximate portfolio for ordered norms must have size $\Omega\left(\frac{\log d}{\log \alpha + \log\log d}\right)$. Given d, let $S = S(d)$ be a superconstant that we specify later; assume that S is an integer that is a power of 2. Let L be the largest integer such that $1 + S^2 + \cdots + S^{2L} \leq d$, then $L = \Theta(\log_S d)$. The d machines are divided into $L+1$ classes from 0 to L: there are S^{2l} machines in the lth class and the processing time on these machines is $p_l = S^l$. The number of jobs n is S^{3L}; it is chosen so as to ensure that all vertices in the constraint polytope for Equation LP1 are good, and can be rounded to an integral solution that is only worse by a factor at most 2 (Lemma 4.9).

There are $L+1$ weight vectors for our instance. The first weight vector is $w(0) = (1, 1, \ldots, 1)$. The second weight vector is $w(1) = \left(1, \frac{1}{S^2}, \frac{1}{S^2}, \ldots, \frac{1}{S^2}\right)$. More generally, for $l \in [0, L]$,

$$w(l) = \Bigg(1, \underbrace{\frac{1}{S^2}, \ldots, \frac{1}{S^2}}_{S^2}, \underbrace{\frac{1}{S^4}, \ldots, \frac{1}{S^4}}_{S^4}, \ldots, \underbrace{\frac{1}{S^{2l-2}}, \ldots, \frac{1}{S^{2l-2}}}_{S^{2l-2}}, \underbrace{\frac{1}{S^{2l}}, \ldots, \frac{1}{S^{2l}}}_{\text{remaining}}\Bigg).$$

With some foresight, we choose S such that $\frac{S}{L} = 5\alpha$. We claim the following: for each $l \in [0, L-1]$,

1. There is a schedule $\hat{x}(l)$ for this instance with $\|\hat{x}(l)\|_{(w(l))} \leq nlS^{-l}$.

2. Any schedule y that schedules more than $n/4$ jobs on machines in classes $l+1$ to L has $\|y\|_{(w(l))} \geq \frac{nS}{4} \cdot S^{-l}$. Combined with the above and since $\alpha \leq \frac{S}{4L}$, it cannot be an α-approximation for the ordered norm corresponding to $w(l)$.

3. Any schedule y that schedules more than $n/4$ jobs on machines in classes 0 to $l-1$ has $\|y\|_{(w(l))} \geq \frac{nS}{2} \cdot S^{-l}$. Therefore, it cannot be an α-approximation for the ordered norms corresponding to $w(l)$.

4. $L = \Theta(\log_S d) = \Omega\left(\frac{\log d}{\log \alpha + \log\log d}\right)$.

Claims 1, 2, and 3 imply that any α-approximate solution for norm $w(l)$ must schedule at least $n/2$ jobs on machines in class l. Another application of claims 2 and 3 then implies that a portfolio that is α-approximate for weight vectors $\{w(0), \ldots, w(L-1)\}$ must have distinct solutions for each weight vector, and therefore has size at least L. Claim 4 then implies our theorem.

Claim 4 follows from some algebra: $L = \Theta(\log_S d) = \Theta(\log_{\alpha L} d) = \Theta\left(\frac{\log d}{\log \alpha + \log L}\right)$. If $L = \Omega(\log d)$, then we are done since the target size is anyway $\Theta\left(\frac{\log d}{\log \alpha + \log\log d}\right) = O(\log d)$ for all constant α. Otherwise, $\log L = O(\log\log d)$ and so $L = \Theta\left(\frac{\log d}{\log \alpha + \log L}\right) = \Omega\left(\frac{\log d}{\log \alpha + \log\log d}\right)$.

We move to claim 1. As alluded to before, $n = S^{3L}$ has been chosen so that each vertex $x(l)$ of the constraint polytope is good (see Equation 4.7), since

$$\frac{n}{1 \cdot \frac{1}{1} + S^2 \cdot \frac{1}{S} + \cdots + S^{2L} \cdot \frac{1}{S^L}} \geq \frac{n}{2S^L} \geq S^L = p_L.$$

With this in hand, it is sufficient to give a fractional solution $x(l)$ with $\|x(l)\|_{(w(l))} = \Theta(nlS^{-l})$, since Lemma 4.9 then implies the existence of an integral solution $\hat{x}(l)$ with norm value at most twice. Consider $x(l) = (a, \ldots, a, 0, \ldots, 0)$ where the first $1 + S^2 + \cdots + S^{2l}$ coordinates are non-zero and equal to a; all other coordinates are 0. Since a total of n jobs must be scheduled (Equation 4.8),

$$n = a\left(1 \cdot \frac{1}{1} + S^2 \cdot \frac{1}{S} + \cdots + S^{2l} \cdot \frac{1}{S^l}\right) \geq aS^l,$$

so that $a \leq \frac{n}{S^l}$. Therefore,

$$\|x(l)\|_{(w(l))} = a \times \text{ sum of first } (1 + S^2 + \cdots + S^{2l}) \text{ coordinates of } w(l) = a \cdot l \leq nlS^{-l}.$$

We move to claim 2. Let y schedule more than $n/4$ jobs on machines in classes $l+1$ to

L. Irrespective of how these $n/4$ jobs are distributed, they contribute a total load of at least $(n/4) \times S^{l+1}$. Since all coordinates of $w(l)$ are at least $\frac{1}{S^{2l}}$, the contribution of these jobs to $\|y\|_{(w(l))}$ is at least

$$\frac{1}{S^{2l}} \times \frac{n}{4} S^{l+1} = \frac{nS}{4} \cdot S^{-l}.$$

Finally, we prove claim 3. Consider the restricted instance with only machines from classes $0, \ldots, l-1$ and $n/4$ jobs. Let x be the optimal fractional solution for this instance for L_∞ norm; it is easy to see that x must have equal loads on machines, so that from Equation 4.8:

$$n = \|x\|_\infty \left(1 \cdot \frac{1}{1} + S^2 \cdot \frac{1}{S} + \cdots + S^{2l-2} \cdot \frac{1}{S^{l-1}}\right) \leq 2\|x\|_\infty S^{l-1},$$

implying $\|x\|_\infty \geq \frac{nS^{-l+1}}{2}$. Therefore, any integral optimal solution $\hat{x}$ to this restricted instance must also satisfy

$$\|\hat{x}\|_\infty \geq \|x\|_\infty \geq \frac{nS^{-l+1}}{2}.$$

Since y is a solution to the larger original instance, we have $\|y\|_\infty \geq \|\hat{x}\|_\infty$. Finally, since $w(l) = 1$ by construction, we get $\|y\|_{(w(l))} \geq \|y\|_\infty$. Together, we get $\|y\|_{(w(l))} \geq \frac{nS}{2} \cdot S^{-l}$. This completes the proof of claim 3 and of Theorem 4.1.

Portfolios for different classes of norms. Recall Lemma 2.2: if x^* is a simultaneous α-approximation for each top-ℓ norm, then it is a simultaneous α-approximation for all symmetric monotonic norms. One might naturally wonder if this is true for portfolios: is an α-approximate portfolio for top-ℓ norms also an α-approximate portfolio for all symmetric monotonic norms? We show that not only is this false but that the gap between portfolio sizes for top-ℓ norms and ordered norms can be unbounded, by constructing such instances for MLIJ.

Theorem 4.4. *There exist instances of* MLIJ *on d machines for which*

1. *there is an $O(1)$-approximate portfolio X of size 2 for top-ℓ norms, and*

2. *any $O(1)$-approximate portfolio X' for ordered norms has size $\Omega\left(\frac{\log d}{\log\log d}\right)$.*

Proof. From Theorem 4.1, there exist doubling instances of MLIJ with d machines where any $O(1)$-approximate portfolio for ordered norms must have size $\Omega\left(\frac{\log d}{\log\log d}\right)$. We will show here that *all* doubling instances of MLIJ admit $O(1)$-approximate portfolio of size 2 for all top-ℓ norms. Together, this implies the result.

Recall Lemma 4.9, Lemma 4.10: $X' = \{\hat{x}(1), \ldots, \hat{x}(L)\}$ is an $O(1)$-approximate portfolio for all ordered norms where $\frac{1}{2}x(l) \leq \hat{x}(l) \leq 2x(l)$ for all $l \in [L]$. Therefore, $\|\hat{x}(l)\|_{(\mathbf{1}_\ell)}$ is within factor 2 of $\|x(l)\|_{(\mathbf{1}_\ell)}$ for all $\ell \in [d]$. Further for all $\ell \in [d]$,

$$\|x(l)\|_{(\mathbf{1}_\ell)} = \begin{cases} \frac{ln}{\sum_{i\in[l]}\frac{1}{p_i}} & \text{if } l \leq \ell, \\ \frac{\ell n}{\sum_{i\in[l]}\frac{1}{p_i}} & \text{if } l > \ell. \end{cases}$$

Fix ℓ. Since $p_i \leq p_{i+1}$ for all i, $\frac{ln}{\sum_{i\in[l]}\frac{1}{p_i}}$ is non-increasing in l. Further, $\frac{\ell n}{\sum_{i\in[l]}\frac{1}{p_i}}$ is decreasing in l. Therefore, the smallest among $\|x(l)\|_{(\mathbf{1}_\ell)}, l \in [L]$ is either $\|x(1)\|_{(\mathbf{1}_\ell)}$ or $\|x(L)\|_{(\mathbf{1}_\ell)}$. This implies

$$\begin{aligned} \min\{\|\hat{x}(1)\|_{(\mathbf{1}_\ell)}, \|\hat{x}(L)\|_{(\mathbf{1}_\ell)}\} &\leq 2\min\{\|x(1)\|_{(\mathbf{1}_\ell)}, \|x(L)\|_{(\mathbf{1}_\ell)}\} \\ &\leq 2\min\{\|x(1)\|_{(\mathbf{1}_\ell)}, \|x(2)\|_{(\mathbf{1}_\ell)}, \ldots, \|x(L)\|_{(\mathbf{1}_\ell)}\} \\ &\leq 4\min\{\|\hat{x}(1)\|_{(\mathbf{1}_\ell)}, \|\hat{x}(2)\|_{(\mathbf{1}_\ell)}, \ldots, \|\hat{x}(L)\|_{(\mathbf{1}_\ell)}\}. \end{aligned}$$

Since $\{\hat{x}(1), \ldots, \hat{x}(L)\}$ is an $O(1)$-approximate portfolio for all ordered norms, this implies that $\{\hat{x}(1), \hat{x}(L)\}$ is an $O(1)$-approximate portfolio for all top-ℓ norms. □

Example 4.1. *One can also show that portfolios for ordered norms are not portfolios for L_p norms: consider an instance of* MLIJ *with $p_i = \sqrt{i}$ for each machine $i \in [d]$. Denote $\rho(l) = \sum_{i\in[l]}\frac{1}{p_i} = \sum_{i\in[l]}\frac{1}{\sqrt{i}}$; also denote the dth Harmonic number $H_d = \sum_{i\in[d]}\frac{1}{i} = \Theta(\log d)$. Then for each $l \in [d]$, $x(l) = \frac{n}{\rho(l)}\big(\underbrace{1, \ldots, 1}_{l}, 0, \ldots, 0\big)$. Recall (Lemma 4.9, Lemma 4.10) that there exists an $L \in [d]$ such that (1) $\frac{1}{2}x(l) \leq \hat{x}(l) \leq 2x(l)$ for all $l \in [L]$*

and (2) $X' = \{\hat{x}(1), \ldots, \hat{x}(L)\}$ is an $O(1)$-approximate portfolio for ordered norms. We claim that each $x \in X'$ is an $\Omega(\sqrt{H_d})$-approximation for the L_2 norm objective.

For each $l \in [L]$, $\rho(l) \leq 1 + 2\int_1^l \frac{2}{\sqrt{x}} \leq 4\sqrt{l}$, so that

$$2\|\hat{x}(l)\|_2 \geq \|x(l)\|_2 = \frac{n}{\rho(l)} \cdot \sqrt{l} \geq \frac{n}{4}.$$

Consider the following assignment: assign $n_i = \frac{n}{iH_d}$ jobs to machine $i \in [d]$ (we choose n large enough so that each n_i is integral). Then this is a valid assignment since $\sum_{i\in[d]} n_i = n$ by definition of H_d. The machine loads for this assignment are $x_i = n_i p_i = \frac{n}{H_d\sqrt{i}}$. The L_2 norm of x is

$$\|x\|_2 = \frac{n}{H_d}\sqrt{\sum_{i\in[d]} \frac{1}{i}} = \frac{n}{\sqrt{H_d}}.$$

Therefore, each $\hat{x}(l)$ is an $\Omega(\sqrt{H_d})$-approximation for the L_2 norm.

4.5 `OrderAndCount` for COVERINGPOLYHEDRON

In this section, we use `OrderAndCount` to prove Theorem 4.2 to obtain portfolios for COVERINGPOLYHEDRON. A d-dimensional covering polyhedron is defined as $\mathcal{P} = \{x \in \mathbb{R}^d : Ax \geq b, x \geq 0\}$ where $A \in \mathbb{R}^{r\times d}_{\geq 0}$ is the constraint matrix with r constraints and $b \in \mathbb{R}^r_{\geq 0}$. As alluded to before, such polyhedra model workload management in settings with r splittable jobs to be distributed among d machines, each of which can run all r jobs concurrently. We give an algorithm that given $\mathcal{P}$ and any constant $\varepsilon \in (0, 1]$, obtains a portfolio of size $O\left(\left(\frac{\log(d/\varepsilon)}{\varepsilon}\right)^{3r^2-2r}\right)$ that is (i) $(1+\varepsilon)$-approximate for ordered norms and (ii) $O(\log d)$-approximate for symmetric monotonic norms.

We focus on the result for ordered norms since the result for symmetric monotonic norms follows from Lemma 4.2. Assume that $b = \mathbf{1}_r = (1, \ldots, 1)^\top$, without loss of generality by rescaling rows of A if necessary (and removing rows with $b = 0$ since they are feasible anyway for all $x \geq 0$).

For any order or permutation π on $[d]$, define the restriced polytope $\mathcal{P}_\pi := \mathcal{P} \cap \{x \in$

$\mathbb{R}^d : x_{\pi(1)} \geq \cdots \geq x_{\pi(d)} \geq 0\}$. Our high-level plan is the same: any ordered norm $\|\cdot\|_{(w)}$ is a linear function on each $\mathcal{P}_\pi$. Therefore, the minimum norm point $x(w) := \arg\min_{x \in \mathcal{P}} \|x\|_{(w)}$ must be one of the vertices of some $\mathcal{P}_\pi$. Denote by X the union of sets of vertices across all orders π; then X is an optimal portfolio for ordered norms. However, two main issues potentially blow up the size $|X|$:

1. Each $\mathcal{P}_\pi$ can have too many vertices. For each vertex of $\mathcal{P}_\pi$, d out of $r+d$ constraints $Ax \geq \mathbf{1}_r, x_{\pi(1)} \geq \cdots \geq x_{\pi(d)} \geq 0$ must be tight. Therefore, $\mathcal{P}_\pi$ may have $\binom{d+r}{d} \sim d^r$ vertices.

2. There are $d!$ orders $\pi \in \text{Perm}(d)$. Since we are taking a union over all such orders, we get the following rough bound on the portfolio size $|X|$:

$$|X| \leq \binom{\text{number of vertices}}{\text{in each } \mathcal{P}_\pi} \times \binom{\text{number of orders}}{\pi} \sim d^r \times d!. \tag{4.11}$$

Broadly, we first use a *sparsification* idea (Subsection 4.5.1) to reduce the effective dimension to $\left(\frac{\log(d/\varepsilon)}{\varepsilon}\right)^r$ from d, losing approximation factor $1+\varepsilon$. This is done by counting the number of unique columns of A up to factor $1+\varepsilon$. Sparsification also gives an upper bound on the number of vertices in the restricted region $\mathcal{P}_\pi$ corresponding to each order. There are still too many orders to sum over, and this is where the *primal-dual counting technique* comes in (Subsection 4.5.2). It allows us to restrict to a small number of permutations π by counting in a suitable dual space to our primal problem:

$$\min_{x \geq 0} \|x\|_{(w)} \quad \text{s.t. } Ax \geq b. \tag{Primal}$$

$$\min \|\lambda^\top A\|^*_{(w)} \quad \text{s.t. } \lambda \in \mathbf{\Delta}_r. \tag{Dual}$$

The advantage with the 'dual' is that the underlying polytope – the probability simplex $\mathbf{\Delta}_r := \{z \in \mathbb{R}^r : z \geq 0, \sum_{i \in [r]} z_i = 1\}$ in r dimensions – is easier to handle. Additionally, it is in r dimensions instead of d. The key ingredient connecting the primal and the dual

Algorithm 3 `SparsifyPolyhedron`($\mathcal{P}$)

input: covering polyhedron $\mathcal{P} = \{x \in \mathbb{R}^d : Ax \geq \mathbf{1}_r, x \geq 0\}$, error parameter $\varepsilon \in (0, 1]$

output: another covering polyhedron $\widetilde{\mathcal{P}} = \{x \in \mathbb{R}^d : \widetilde{A}x \geq \mathbf{1}_r, x \geq 0\}$

define $\mu = \frac{3d^2}{\varepsilon}$ and initialize $\widetilde{A} = \mathbf{0}_{r \times d}$
for $i = 1$ to r **do**
  define $a_i^* = \max_{j \in [d]} A_{i,j}$ and $B(i) = \left\{ j \in [d] : A_{i,j} < \frac{a_i^*}{\mu} \right\}$
  for $j \in [d]$ **do**
    if $j \in B(i)$ **then**
      set $\widetilde{A}_{i,j} = 0$
    else
      let $l \in [0, \lfloor \log_{(1+\varepsilon/2)} \mu \rfloor]$ be the unique integer such that

$$\frac{a_i^*}{\mu} \left(1 + \frac{\varepsilon}{2}\right)^l \leq A_{i,j} < \frac{a_i^*}{\mu} \left(1 + \frac{\varepsilon}{2}\right)^{l+1}$$

      set $\widetilde{A}_{i,j} = \frac{a_i^*}{\mu} \left(1 + \frac{\varepsilon}{2}\right)^l$
return $\widetilde{A}$, $\widetilde{P} = \{x \in \mathbb{R}^d : \widetilde{A}x \geq \mathbf{1}_r, x \geq 0\}$

will be the Cauchy-Schwarz inequality for ordered norms (Lemma 4.4).

4.5.1 Sparsification

Denote $N = O\big(\frac{\log(d/\varepsilon)}{\varepsilon}\big)$. We give a sparsification procedure (Algorithm 3) that reduces the number of distinct columns in A to N^r. For each row of matrix A, this sparsification (1) removes 'small' entries in the row and (2) restricts the number of unique entries in the row to N. Since there are r rows, the number of distinct columns after sparsification is N^r.

Lemma 4.11. *The columns of matrix $\widetilde{A} \in \mathbb{R}_{\geq 0}^{r \times d}$ output by Algorithm* `Sparsify-Polyhedron` *take one of N^r values, i.e., $[d]$ can be partitioned into $S_1, \ldots, S_{N^r}$ such that for any $j, j' \in S_l$, the jth and j'th columns of $\widetilde{A}$ are the same.*

Proof. Fix row $i \in [r]$. By construction, each entry in the ith row of $\widetilde{A}$ is in the set $\{0\} \cup \left\{ \frac{a_i^*}{\mu} \left(1 + \frac{\varepsilon}{2}\right)^l : l \in [0, \lfloor \log_{(1+\varepsilon/2)} \mu \rfloor] \right\}$, where $\mu = \frac{3d^2}{\varepsilon}$. These are $O(\log_{(1+\varepsilon/2)} \mu) = O(\log_{(1+\varepsilon/2)}(d^2/\varepsilon)) = O\left(\frac{\log(d/\varepsilon)}{\varepsilon}\right) = N$ distinct numbers. Since each column has r entries, one from each row, we get a total of N^r possible values for a column. □

Sparsification only loses a factor $(1+\varepsilon)$ in the approximation (proof deferred to Section B.2):

Lemma 4.12. $\widetilde{\mathcal{P}} = \{x : \widetilde{A}x \geq \mathbf{1}_r, x \geq 0\}$ *output by Algorithm* `SparsifyPolyhedron` *is a* $(1+\varepsilon)$*-approximate portfolio for symmetric monotonic norms over* $\mathcal{P}$*.*

These lemmas allow us to work with $\widetilde{\mathcal{P}} = \{x : \widetilde{A}x \geq \mathbf{1}_r, x \geq 0\}$ instead of $\mathcal{P}$. $\widetilde{\mathcal{P}}$ has the nice property that columns of $\widetilde{A}$ take at most N^r distinct values. We will give an optimal portfolio for ordered norms over $\widetilde{\mathcal{P}}$ of size $O(N^{3r^2-2r})$. Using Lemma 4.1 to compose portfolios, this is sufficient to prove Theorem 4.2. Hereafter, we will only work with the sparsified matrix $\widetilde{A}$ and polyhedron $\widetilde{\mathcal{P}}$. For ease of notation, we drop the symbol $\widetilde{A}$ and assume that the original matrix A and corresponding polyhedron $\mathcal{P}$ are already given to us in the sparsified form.

Let $S_1, \ldots, S_{N^r}$ denote the partition of $[d]$ based on the value of columns of A, i.e., for each $l \in [N^r]$ and $j, j' \in S_l$, jth and j'th columns of A are the same. Further, define $\mathcal{Q} = \{x \in \mathbb{R}^d_{\geq 0} : x_j = x_{j'} \; \forall \; j, j' \in S_l, \; \forall \; l \in [N^r]\}$, i.e., the set of all non-negative vectors in $\mathbb{R}^d$ that attain the same value for all coordinates $j \in S_l$, for all $l \in [N^r]$. Define $\mathcal{P}^{=} = \mathcal{P} \cap \mathcal{Q}$. Recall that for weight vector w, we define $x(w) := \arg\min_{x \in \mathcal{P}} \|x\|_{(w)}$. Our next lemma shows that $x(w) \in \mathcal{P}^{=}$:

Lemma 4.13. *Given a weight vector* w*, we can assume without loss of generality that for all* $l \in [N^r]$ *and* $j, j' \in S_l$*,* $x(w)_j = x(w)_{j'}$*. That is,* $\mathcal{P}^{=}$ *is an optimal portfolio for symmetric monotonic norms over* $\mathcal{P}$*.*

Proof. Suppose $x(w)_j \neq x(w)_{j'}$, say $x(w)_j > x(w)_{j'}$. Then consider $\overline{x} \in \mathbb{R}^d$ such that $\overline{x}_k = x(w)_k$ for all $k \neq j, j'$, and $\overline{x}_j = \overline{x}_{j'} = \frac{x(w)_j + x(w)_{j'}}{2}$. Then $\overline{x} \preceq x(w)$ and so Lemma Lemma 2.1 gives $\|\overline{x}\|_{(w)} \leq \|x(w)\|_{(w)}$.

Further, clearly $\overline{x} \geq 0$ since $x(w) \geq 0$. Since the jth and j'th columns of A are equal, $A\overline{x} = Ax(w) \geq \mathbf{1}_r$, or that $\overline{x} \in \mathcal{P}$. □

We define *reduced orders* next, which are simply orders in the smaller space $\mathbb{R}^{N^r}$:

Definition 4.1 (Reduced orders)**.** *An order ρ on $[N^r]$ is called a* reduced order. *For $x \in \mathcal{Q}$, define vector $z(x) \in \mathbb{R}^{N^r}$ as follows: for $l \in [N^r]$, $z(x)_l := x_j$ where $j \in S_l$. $x \in \mathcal{Q}$ is said to* satisfy *reduced order ρ if $z_{\rho(1)} \geq \cdots \geq z_{\rho(N^r)} \geq 0$. Given a reduced order ρ, define polyhedron*

$$\mathcal{P}_\rho^= = \{x \in \mathcal{P} \cap \mathcal{Q} : x \textit{ satisfies reduced order } \rho\}.$$

At this point, a natural first attempt at bounding the portfolio size is to count the number of ordered norms in the space of 'reduced' vectors $\{z(x) : x \in \mathcal{P}^=\} \subseteq \mathbb{R}^{N^r}$. After all, [20]'s result shows that there are at most $\text{poly}(N^{r/\varepsilon})$ ordered norms in $\mathbb{R}^{N^r}$ up to a $(1+\varepsilon)$-approximation. However, this approach fails because ordered norms on $\mathbb{R}^d$ cannot be translated appropriately into an ordered norm on the smaller space $\mathbb{R}^{N^r}$.

For example, consider the covering polyhedron $\mathcal{P} = \{x \in \mathbb{R}^3_{\geq 0} : x_1 \geq 2, x_2 + x_3 \geq 4, 2x_1 + x_2 + x_3 \geq 10\}$. The point $(3,2,2) \in \mathcal{P}$ is the (unique) minimizer of the L_1 norm, which corresponds to weight vector $w = (1,1,1)$. The constraint polytope for $\mathcal{P}$ has two unique columns, and the corresponding 'reduced covering polyhedron' is $\mathcal{P}' = \{z \in \mathbb{R}^2 : z_1 \geq 2, z_2 \geq 2, z_1 + z_2 \geq 5\}$. A point $(a,b,b) \in \mathcal{P}$ corresponds to the point $(a,b) \in \mathcal{P}'$. However, by a majorization argument, the point $(5/2, 5/2) \in \mathcal{P}'$ minimizes *all ordered norms* on $\mathcal{P}'$, but the corresponding point $(5/2, 5/2, 5/2) \in \mathcal{P}$ with L_1 norm 7.5 is suboptimal for the L_1 norm.

Therefore, it is not sufficient to count ordered norms in $\mathbb{R}^{N^r}$, and we need an alternate approach that we describe next. Suppose that we are given some reduced order ρ. Then for $x \in \mathcal{P}_\rho^=$, $\|x\|_{(w)}$ is a linear function of x. Therefore, given a weight vector w, if $x(w)$ satisfies reduced order ρ, then $x(w)$ is one of the vertices of polyhedron $\mathcal{P}_\rho^=$. With this observation, the rest of the proof is organized as follows:

- For each reduced order ρ, $\mathcal{P}_\rho^=$ has at most $N^{r^2} + 1$ vertices (Lemma 4.14).
- Consider the set Π of reduced orders such that for any weight vector w, $x(w)$ satisfies some reduced order $\rho \in \Pi$, i.e, $\Pi = \{\text{reduced order } \rho : \exists\, w \text{ where } x(w) \text{ satisfies } \rho\}$.

Then we will show that $|\Pi| \leq N^{2r(r-1)}$ (Lemma 4.15).

Together, these observations mean that $X := \bigcup_{\rho \in \Pi} \left(\text{vertices of } \mathcal{P}_\rho^=\right)$ is an optimal portfolio for ordered norms over $\mathcal{P}_=$. By Lemma 4.13, $\mathcal{P}_=$ is an optimal portfolio for ordered norms over $\mathcal{P}$. Therefore, Lemma 4.1 implies that X is an optimal portfolio for ordered norms over $\mathcal{P}$. Further,

$$\begin{aligned}|X| &= \left| \bigcup_{\rho \in \Pi} \left(\text{vertices of } \mathcal{P}_\rho^=\right) \right| \leq \sum_{\rho \in \Pi} \left|\left(\text{vertices of } \mathcal{P}_\rho^=\right)\right| \\ &\leq \sum_{\rho \in \Pi} (N^{r^2} + 1) = |\Pi|(N^{r^2} + 1) \leq N^{2r(r-1)}(N^{r^2} + 1) = O(N^{3r^2 - 2r}).\end{aligned}$$

This implies Theorem 4.2. We prove Lemma 4.14 next and defer Lemma 4.15 to the next section.

Lemma 4.14. *For each reduced order ρ, $\mathcal{P}_\rho^=$ has at most $N^{r^2} + 1$ vertices*

Proof. For simplicity, assume (after possibly relabeling indices) that $\rho(l) = l$ for all $l \in [N^r]$, and that $S_1 = \{1, \ldots, |S_1|\}, S_2 = \{1+|S_1|, \ldots, |S_1|+|S_2|\}$ etc. Then the polyhedron $\mathcal{P}_\rho^=$ is the set of all x such that $A_i^\top x \geq 1$ for all rows $i \in [r]$ and

$$x_1 = \cdots = x_{|S_1|} \geq x_{|S_1|+1} = \cdots = x_{|S_1|+|S_2|} \geq \cdots \geq x_{d-|S_{N^r}|+1} = \cdots = x_d \geq 0.$$

Any vertex on $\mathcal{P}_\rho^=$ corresponds to a set of d (linearly independent) inequalities. The constraints of the polytope have $d - N^r$ equalities and $N^r + r$ inequalities. Therefore, each vertex corresponds to some N^r of the $N^r + r$ inequalities being tight. The number of such choices is $\binom{N^r+r}{N^r}$. Then,

$$\binom{N^r + r}{N^r} = \binom{N^r + r}{r} \leq \left(1 + \frac{N^r}{r}\right)^r.$$

For $r = 1$, this is at most $1 + N^r$. For $r \geq 2$, $\left(1 + \frac{N^r}{r}\right)^r \leq (N^r)^r = N^{r^2}$. □

4.5.2 Primal-Dual Counting

In this section, we study the set Π of reduced orders such that for any weight vector w, $x(w)$ satisfies some reduced order $\rho \in \Pi$, i.e, $\Pi = \{\text{reduced order } \rho : \exists\, w \text{ s.t. } x(w) \text{ satisfies } \rho\}$. We will prove the following:

Lemma 4.15. *The number of possible reduced orders* $|\Pi| \leq N^{2r(r-1)}$.

The main idea is to count reduced orders not on $x(w)$, but in a dual space. We write the following modified primal and dual, and denote $\lambda(w) = \arg\min_{\lambda \in \mathbf{\Delta}_r} \|\lambda^\top A\|^*_{(w)}$:

$$\min \|x\|_{(w)} \quad \text{s.t.} \quad Ax \geq \mathbf{1}_r, x \in \mathcal{Q}. \tag{Primal'}$$

$$\min \|A^\top \lambda\|^*_{(w)} \quad \text{s.t.} \quad \lambda \in \mathbf{\Delta}_r \tag{Dual}$$

Note that $(A^\top \lambda)_j$ is simply the dot product of the jth column of A with λ. Further, recall that for all $j, j' \in S_l$ and for any $l \in [N^r]$, the jth and j'th columns of A are equal. Therefore, we have $(A^\top \lambda)_j = (A^\top \lambda)_{j'}$ for any λ. By definition, this means that $A^\top \lambda \in \mathcal{Q}$ for all $\lambda \geq 0$.

The next lemma establishes the crucial connection between reduced orders in Equation Primal' and Equation Dual. It uses Lemma 4.4 (Ordered Cauchy-Schwarz) along with a Lagrangian function; we defer its proof to Section B.2.

Lemma 4.16. *Given a weight vector* w, $\|x(w)\|_{(w)} \|A^\top \lambda(w)\|^*_{(w)} = 1$. *Further, there is a reduced order* ρ *such that both* $x(w), A^\top \lambda(w)$ *satisfy* ρ.

As a consequence of this lemma, it is sufficient to count reduced orders in the dual:

$$\begin{aligned}
\Pi &= \{\text{reduced order } \rho : \exists\, w \text{ where } x(w) \text{ satisfies } \rho\} \\
&= \{\text{reduced order } \rho : \exists\, w \text{ where } A^\top \lambda(w) \text{ satisfies } \rho\} \\
&\subseteq \{\text{reduced order } \rho : \exists\, \lambda \in \mathbf{\Delta}_r \text{ where } A^\top \lambda \text{ satisfies } \rho\}.
\end{aligned}$$

Denote $\Pi^* = \{\text{reduced order } \rho : \exists\, \lambda \in \mathbf{\Delta}_r \text{ where } A^\top \lambda \text{ satisfies } \rho\}$. We will show that $|\Pi^*| \leq N^{2r(r-1)}$. From the above, this is sufficient to prove Lemma 4.15. Our final lemma is a geometric counting inequality.[1]

Lemma 4.17. *T hyperplanes partition $\mathbf{\Delta}_r$ into at most $T^{r-1} + 1$ regions.*

Proof. The result is trivially true for $r = 1$ since $\mathbf{\Delta}_1$ is a point. For $r = 2$, $\mathbf{\Delta}_2$ is a line segment, and T 'hyperplanes' partition it into $\leq T+1$ regions. For $r \geq 3$, we use induction on T. 1 hyperplane clearly divides any convex body into at most $2 \leq 1^{r-1} + 1$ regions. Suppose $T > 1$. Let the Tth hyperplane be $\mathcal{H}$. By the induction hypothesis, the first $T-1$ hyperplanes divide $\mathbf{\Delta}_r$ into at most $(T-1)^{r-1} + 1$ regions. If $\mathbf{\Delta}_r \subseteq \mathcal{H}$, then $\mathcal{H}$ does not add any new regions, and we are done.

Otherwise, the number of new regions $\mathcal{H}$ adds is the number of regions that the first $T-1$ hyperplanes partition $\mathbf{\Delta}_r \cap \mathcal{H}$ into. But $\mathbf{\Delta}_r \cap \mathcal{H}$ can be affinely transformed into $\mathbf{\Delta}_{r-1}$ in this case, and so the number of new regions is at most $(T-1)^{r-2} + 1$. Therefore, by the induction hypothesis, the total number of regions with T hyperplanes is at most

$$((T-1)^{r-1} + 1) + ((T-1)^{r-2} + 1) \leq T^{r-1} + 1 \quad \forall\, T \geq 1, r \geq 3. \qquad \square$$

We are ready to finish the proof of Lemma 4.15. Partition $\mathbf{\Delta}_r$ into regions $\{R_\rho : \rho \in \Pi^*\}$, where $R_\rho := \{\lambda \in \mathbf{\Delta}_r : A^\top \lambda \text{ satisfies } \rho\}$. The size $|\Pi^*|$ is exactly the number of such regions. Pick $j, j' \in [d]$ such that j, j' belong to different sets $S_l, S_{l'}$. Then these regions are separated by hyperplanes of the form $\{\lambda : (A^\top \lambda)_j = (A^\top \lambda)_{j'}\}$, i.e., different reduced orders exist on different sides of these hyperplanes. There are $\binom{N^r}{2}$ such hyperplanes, each corresponding to a pair of sets $S_l, S_{l'}$. By the above lemma, these partition $\mathbf{\Delta}_r$ into at most

$$\binom{N^r}{2}^{r-1} + 1 = \left(\frac{N^r(N^r-1)}{2}\right)^{r-1} + 1 \leq N^{2r(r-1)}.$$

[1] This result also follows from [113]'s (stronger) bound on the number of regions induced by T hyperplanes in an r-dimensional Euclidean space. For completeness, we provide a (shorter) proof here.

regions. Thus, $|\Pi| \leq |\Pi^*| = |\{R_\rho : \rho \in \Pi^*\}| \leq N^{2r(r-1)}$. This finishes the proof of Lemma 4.15, and therefore the proof of Theorem 4.2.

Finally, we remark that this also gives an algorithm with runtime $\mathrm{poly}(N^{r^2}, d)$: tracing back, find the set Π^* using the above hyperplane argument, and then simply output the union of vertices of $\mathcal{P}_\rho^=$ for all $\rho \in \Pi^*$.

4.6 Conclusion

Motivated by fairness concerns in workload distribution and placement of critical facilities, we studied portfolios for scheduling with identical jobs (MLIJ) and for covering polyhedra. We characterized the trade-off between portfolio size and the approximation factors for the problem of scheduling identical jobs on unidentical machines. Then, we extended the portfolio size upper bound to covering polyhedra. We state two open questions here:

General COVERINGPOLYHEDRON: For covering polyhedra in dimension d, we improved portfolio sizes from the general bound of $\mathrm{poly}(d)$ when the number of constraints $r = o(\sqrt{\log d}/(\log\log d))$. We conjecture that this is tight up to polylogarithmic factors, i.e, that there exist covering polyhedra in dimension d with $O(\log d)$ constraints such that any $O(\log d)$-approximate portfolios for symmetric monotonic norms must have polynomial size.

Scheduling with unidentical jobs: We gave $O(1)$-approximate portfolios of size $O(\log d)$ for MLIJ, i.e., machine load minimization on d machines with identical jobs. It is unclear if there exists a similar-sized portfolio for the more general problem of machine-load minimization with *unidentical* jobs.

CHAPTER 5

SIMULTANEOUS APPROXIMATIONS

We next turn our attention to portfolios of size-1, i.e., simultaneous approximations [8, 9] for symmetric monotonic norms, and show stronger approximation guarantees for specific problems. We develop an `IterativeOrdering` framework that unifies simultaneous approximation algorithms for many combinatorial optimization problems. The key improvements we obtain using our framework for symmetric monotonic norms (summarized in Table 5.1) are:

- COMPLETIONTIMES: For minimizing symmetric monotonic norms of the *completion times* of jobs (e.g., jobs on the cloud computing servers [114]) in a scheduling problem, we show the existence of simultaneous 4-approximation and give a polynomial-time simultaneous 8-approximation. These are the first constant-factor results for this problem, to the best of our knowledge. We also give an instance (see Section 5.3) where no simultaneous 1.13-approximation exists. Note the contrast with the previously discussed problem of minimizing *machine loads*, where a size-1 portfolio may not even be $o(\sqrt{d})$-approximate for all symmetric monotonic norms for d machines, even for identical jobs (i.e, problem MLIJ; see Theorem 4.1).

- ORDEREDSETCOVER: For minimizing symmetric monotonic norms of covering time of n elements of a ground set, we show the existence of a simultaneous 4-approximation. Previously, a polynomial time $O(\log n)$-approximation was known [23], which up to constants is the best possible if P $\neq$ NP [115]. This result highlights the difference between existence and polynomial time computable simultaneous ap-

This chapter is based on joint work with Swati Gupta and Mohit Singh. A preliminary version (alongside Chapter 4) appeared in the *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA) 2025* [108]. An extended version is currently under submission.

proximation.

- ORDEREDTSP: For minimizing symmetric monotonic norms over the time each vertex of a given graph is visited in a Hamiltonian tour, we show the existence of a 5.83-approximation. The previously-known lower bound on the existence of a simultaneous approximation is 1.78 [27]. Therefore, ours bridges the gap in the existence of simultaneous approximations for ORDEREDTSP. The best polynomial-time approximation remains the 8-approximation of Farhadi *et al.* [27].

- k-CLUSTERING: For finding k facilities that minimize symmetric monotonic norms of client distances to open facilities, we give for any $\varepsilon \in (0, 1]$ a polynomial-time *bicriteria approximation* that (a) has objective value within factor $3+\varepsilon$ of the optimal for any symmetric monotonic norm, and (b) opens at most $O\left(\frac{\log n}{\varepsilon}\right) \cdot k$ facilities. This improves upon the previous bicriteria approximation of [9] that has objective value bound $6 + \varepsilon$ with the same bound on the number of facilities.

In many such problems, one seeks to meet the demands of some 'clients' (jobs in scheduling, ground set elements in set cover, vertices in TSP etc) and the goal is to minimize some norm of the vector of times each element is 'satisfied'. For example, in scheduling problems, a job is satisfied when it is completed, in set cover problems, an element is satisfied when it is covered, and in routing problems, a vertex is satisfied when it is visited, etc. We recursively solve the problem, by dividing it into smaller subproblems and stitching the subproblem solution together to get an approximation. The guarantees on satisfaction times are preserved *pointwise*[1], leading to simultaneous approximation guarantees for all symmetric monotonic norms. This generalizes the approach of many previous papers, e.g., [26, 23, 27]. Besides the traditionally studied notion of polynomial-time computable simultaneous approximations, we also providing novel guarantees on the *existence* of certain simultaneous approximations.

[1]That is, if $\tilde{x}$ is the (sorted) approximate vector and x^* is the (sorted) optimal vector of cover times for some norm, then we show coordinate-wise bounds such as $\tilde{x}_i \leq \alpha x^*_i \ \ \forall\, i \in [d]$.

Table 5.1: A summary of simultaneous approximations for symmetric monotonic norms, obtained using the `IterativeOrdering` framework. Here, a bicriteria (α, β)-approximation to a k-CLUSTERING problem opens at most βk facilities, while being within factor α of the optimal solution that opens $\leq k$ facilities (see Section 5.4), for any symmetric montonic norm. γ is a parameter for composable problems (see Section 5.1).

Domain/set $\mathcal{D}$ of feasible solutions		Existence simultaneous approximation	Polynomial-time simultaneous approximation	Reference
γ-COMPOSABLE problem	**This work**	$(\sqrt{\gamma}+1)^2$	-	Theorem 5.1
COMPLETIONTIMES ($\gamma = 1$)	**This work**	4	8	Theorem 5.1
ORDEREDTSP ($\gamma = 2$)	Previous work	16		[23]
	Previous work	8		[27]
	This work	$3 + 2\sqrt{2} \simeq 5.83$	$6 + 4\sqrt{2} \simeq 11.66$	Theorem 5.1
ORDEREDSETCOVER (on ground set of n elements) ($\gamma = 1$)	Previous work	$O(\log n)$		[23]
	This work	4	-	Theorem 5.1
k-CLUSTERING (on n points, bicriteria approximations)	Previous work	$(3+\varepsilon, O((\log n) + 1/\varepsilon))$	$(9+\varepsilon, O((\log n) + 1/\varepsilon))$	[8]
	Previous work	$\left(1+\varepsilon, O\left(\frac{\log n}{\varepsilon}\right)\right)$	$\left(6+\varepsilon, O\left(\frac{\log n}{\varepsilon}\right)\right)$	[9]
	This work	$\left(1+\varepsilon, O\left(\frac{\log n}{\varepsilon}\right)\right)$	$\left(3+\varepsilon, O\left(\frac{\log n}{\varepsilon}\right)\right)$	Theorem 5.2

5.1 `IterativeOrdering` Framework

This section presents our `IterativeOrdering` framework to obtain simultaneous approximations for various combinatorial problems described above. As we will show, all of these problems (1) involve a set of clients and a set of objects that *satisfy* clients, and (2) seek an order on the objects that minimizes the satisfaction time of clients. This is formalized in Definition 5.1. Additionally, such problems are often *composable*, in the sense that orders on different subsets of objects can be combined into a single order on the union of the subsets; this is formalized in Definition 5.2.

Various norms of the vector of satisfaction times correspond to different fairness objectives and lead to different combinatorial optimization problems. We are interested in global guarantees, i.e., simultaneous approximations for all symmetric monotonic norms of this vector. *A priori*, it is unclear whether a given problem even admits good simultaneous approximations, as we showed in Chapter 2, Chapter 3, and Chapter 4. As [23] note, previous works [26] contain similar algorithmic ideas to obtain polynomial-time simultaneous approximations for such problems. We go a step further and formalize the underlying algorithm as `IterativeOrdering`, unifying previous approaches and obtaining new algorithms. As we show in Theorem 5.1, applying it to COMPLETIONTIMES gives the first constant-factor simultaneous approximations for this problem. Applying it to ORDEREDTSP and ORDEREDSETCOVER proves the *existence* of better-than state-of-the-art simultaneous $O(1)$-approximations. Similar ideas apply to k-CLUSTERING problems; we present improved simultaneous approximation to k-CLUSTERING in Section 5.4.

We begin by formally defining the combinatorial problems considered in this section:

- COMPLETIONTIMES. The input consists of n jobs, d machines, and processing times $p_{i,j} > 0$ for each job $j \in [n]$ on machine $i \in [d]$. The output is an assignment of jobs to machines, and an order on the jobs assigned to each machine. Given a norm

$\|\cdot\|$ on $\mathbb{R}^n$, the objective is to minimize the norm of the *completion times* of jobs.[2] Special cases include average completion time minimization (for the L_1 norm) [13], and makespan minimization (for the L_∞ norm) [116].

- ORDEREDSETCOVER. The input consists of a ground set of n elements and m subsets $S_1, \ldots, S_m$ of the ground set. The output is an order on the subsets; each output induces a vector of cover times of elements in the ground set, defined for an element as the position of the first set in the order containing it. Given a norm $\|\cdot\|$ on $\mathbb{R}^n$, the objective is to minimize the norm of cover times. Special cases include classical Set Cover (for the L_∞ norm) [55], and Min-Sum Set Cover or MSSC (for the L_1 norm) [56].

- ORDEREDVERTEXCOVER. This is a special case of ORDEREDSETCOVER where the ground set corresponds to edges of an undirected graph and the subsets correspond to vertices of the graph. Special cases include classical Vertex Cover (for the L_∞ norm), and Min-Sum Vertex Cover or MSVC (for the L_1 norm) [56].

- ORDEREDTSP. The input consists of a metric space on n points or vertices V and a starting vertex $v_0 \in V$. The output is a Hamiltonian tour of the vertices starting at v_0; each tour induces a vector of visit times of the vertices, defined for a vertex as its distance from v_0 along the tour. Given a norm $\|\cdot\|$ on $\mathbb{R}^n$, the objective is to minimize the norm of visit times. Special cases include the Traveling Salesman Problem or TSP (for the L_∞ norm) [57], the Traveling Repairman Problem (for the L_1 norm) [58], and the Traveling Firefighter Problem (for the L_2 norm) [27].

[2]Note that this is different from minimizing norms of machine loads that we considered in MLIJ. The two problems have different fairness interpretations: COMPLETIONTIMES captures fairness for jobs while MLIJ captures fairness for machines.

5.1.1 OrderedSatisfaction Problems

Next, we formally define OrderedSatisfaction problems that capture common structure among the above-mentioned problems.

Definition 5.1. *An* OrderedSatisfaction *problem is specified by*

1. *A set of clients* C.
2. *A set* $\mathcal{X}$ *of* objects. *Each object* $x \in \mathcal{X}$ *is associated with a subset* $C(x)$ *of clients that it* satisfies.
3. *Each collection* $X \subseteq \mathcal{X}$ *of objects is called a* satisfier, *and is said to satisfy the clients in the union* $C(X) := \bigcup_{x \in X} C(x)$.
4. *For each satisfier* $X \subseteq \mathcal{X}$ *and an order* $\pi \in \mathrm{Perm}(X)$ *on* X, *there is an associated time vector* $t(X, \pi) \in \mathbb{R}_{\geq 0}^{X}$ *that must satisfy the following downward closure property: given any time* $T \in \mathbb{R}_{\geq 0}$ *define another satisfier* $X_T := \{x \in T : t(X, \pi)_x \leq T\} \subseteq X$ *with corresponding order* π_T *on* X_T *induced from* π. *Then we must have for all* $x \in X_T$ *that*

$$t(X_T, \pi_T)_x \leq t(X, \pi)_x. \tag{5.1}$$

For each satisfier $X \subseteq \mathcal{X}$ *and order* π *on* X, *also define the* satisfaction time vector $s(X, \pi) \in \mathbb{R}_{\geq 0}^{C(X)}$ *as follows: for each client* $e \in C(X)$, *let* $x \in X$ *be the first object in the order* π *that satisfies* e, *i.e.,* $x = \arg\min_{y \in X : e \in C(y)} \pi(y)$. *Then the satisfaction time* $s(X, \pi)_e$ *of client* e *is defined as*

$$s(X, \pi)_e = t(X, \pi)_x. \tag{5.2}$$

The goal is to output a satisfier $X \subseteq \mathcal{X}$ that satisfies all clients (i.e., $C(X) = C$) and an order π on X. The L_1 norm or the min-sum objective is to minimize the total satisfaction time $\sum_{e \in C} s(X, \pi)_e$ of clients and the L_∞ norm or min-max objective is to minimize the

maximum satisfaction time $\max_{e \in C} s(X,\pi)_e$ of clients across all (X,π). More generally, given a symmetric monotonic norm $\|\cdot\|$ on $\mathbb{R}^C$, the corresponding objective is to minimize $\|s(X,\pi)\|$. We seek simultaneous approximations with guarantees for all symmetric monotonic norms.

Lemma 5.1. COMPLETIONTIMES, ORDEREDSETCOVER, ORDEREDVERTEXCOVER, *and* ORDEREDTSP *are* ORDEREDSATISFACTION *problems.*

We give the proof for COMPLETIONTIMES here, deferring the proof for the other three problems to Section 5.2.

For COMPLETIONTIMES, choose the set of clients $C = [n]$ as the set of jobs. Choose the set of objects to be $\mathcal{X} = [n] \times [d] = \{(j,i) : j \in [n], i \in [d]\}$. The object (j,i) represents the assignment of job j to machine i; and we define $C(j,i) = \{j\}$, i.e., assigning job j to machine i satisfies job j. A satisfier $X \subseteq \mathcal{X}$ corresponds to a *partial assignment*, where some jobs may be unassigned or assigned to multiple machines. Given machine $i \in [d]$, let $J_i(X)$ be the set of jobs assigned to machine i in partial assignment X, i.e $J_i(X) = \{j \in [n] : (j,i) \in X\}$. Then any order π on X induces an order on $J_i(X)$.

Given $(j,i) \in X$ and an order π on X, time $t(X,\pi)_{(j,i)}$ is defined naturally as the completion time of job j on machine i, or more formally, as

$$t(X,\pi)_{(j,i)} := \sum_{\substack{j' \in J_i(X): \\ \pi(j',i) \leq \pi(j,i)}} p_{j',i}, \tag{5.3}$$

Similarly, the satisfaction time of a job j is the least time across machines when it is completed: $s(X,\pi)_j = \min_{i:\, j \in J_i(X)} t(X,\pi)_{(j,i)}$. It is easy to see that the vectors satisfy downward closure (Equation 5.1) with equality: X_T is simply the partial assignment for all jobs that finish under time T.

The goal is to find a schedule (with jobs possibly assigned to multiple machines), i.e., a pair (X,π) such that $C(X) = [n]$.

5.1.2 γ-COMPOSABLE Problems

Given an ORDEREDSATISFACTION problem, consider the following process of composing subproblems: given satisfiers $X_1, \ldots, X_k \subseteq \mathcal{X}$ with corresponding orders $\pi_1, \ldots, \pi_k$ on them, consider the satisfier $\bigcup_{j\in[k]} X_j$ with a *composed order* (denoted $\bigoplus_{j\in[k]} \pi_j$) where every object $x \in X_1$ is ordered first according to π_1, then every object $x \in X_2 \setminus X_1$ is ordered according to π_2, and so on. For example, in COMPLETIONTIMES, this process corresponds to composing partial assignments one after the other, scheduling the jobs in the first partial assignment, then those in the next partial assignment, and so on.

In many ORDEREDSATISFACTION problems, including COMPLETIONTIMES, such compositions suitably maintain the satisfaction times of the clients. To formalize this, define the 'cost' of a satisfier X and order π as $c(X, \pi) := \max_{x\in X} t(X, \pi)_x$. For example, for COMPLETIONTIMES, $c(X, \pi)$ is the makespan of the corresponding partial assignment.

Definition 5.2. *Given $\gamma \geq 1$, an* ORDEREDSATISFACTION *problem is called* γ-COMPOSABLE *if for all satisfiers $X_1, \ldots, X_k \subseteq \mathcal{X}$ and corresponding orders $\pi_1, \ldots, \pi_k$, the time vector $t(X, \pi)$ for the composition $X := \bigcup_{j\in[k]} X_j$ and $\pi := \bigoplus_{j\in[k]} \pi_j$ satisfies the following: for each $j \in [k]$ and each object $x \in X_j \setminus \left(\bigcup_{l\in[j-1]} X_l\right)$, we must have*

$$t(X, \pi)_x \leq \gamma \left(\sum_{l\in[j-1]} c(X_l, \pi_l) \right) + t(X_j, \pi_j)_x. \tag{5.4}$$

For example, we show that COMPLETIONTIMES is 1-composable: indeed, if partial assignments corresponding to $(X_1, \pi_1), (X_2, \pi_2), \ldots, (X_k, \pi_k)$ are put one after the other to form a composed assignment (X, π), then all jobs j scheduled in (X_1, π_1) finish by their completion time in partial assignment (X_1, π_1), all jobs j scheduled in (X_2, π_2) finish by time (makespan of $(X_1, \pi_1) +$ completion time of j in (X_2, π_2)), and so on.

We show in Section 5.2 that ORDEREDSETCOVER and ORDEREDVERTEXCOVER are both 1-composable and ORDEREDTSP is 2-composable.

Lemma 5.2. COMPLETIONTIMES, ORDEREDSETCOVER, *and* ORDEREDVERTEXCOVER *are* 1*-composable and* ORDEREDTSP *is* 2*-composable.*

The next lemma follows from various definitions; we include its proof in Section 5.2.

Lemma 5.3. *Suppose we are given satisfier $X \subseteq \mathcal{X}$, order π on X, and $T > 0$ for an* ORDEREDSATISFACTION *problem. Define $X_T = \{x \in X : t(X,\pi)_x \leq T\}$, and let the restriction of π to T be denoted π_T. Then*

1. $c(X_T, \pi_T) \leq T$
2. *The number of clients $|C(X_T)|$ satisfied by X_T is at least the number of clients (X, π) satisfies within time T, i.e.*

$$|C(X_T)| \geq |\{e \in C(X) : s(X,\pi)_e \leq T\}|.$$

5.1.3 Algorithm `IterativeOrdering`

In this subsection, we give simultaneous approximation algorithm `IterativeOrdering` for γ-COMPOSABLE problems. We show the existence of a simultaneous $(\sqrt{\gamma}+1)^2$-approximation, leading to various approximations for different combinatorial optimization problems.

Formally, given an approximation ratio $\alpha \geq 1$, a simultaneous α-approximation for an ORDEREDSATISFACTION problem is a satisfier $X \subseteq \mathcal{X}$ and an order π on X such that $C(X) = C$ and for any other X', π' with $C(X') = C$, and for any symmetric monotonic norm $\|\cdot\|$ on $\mathbb{R}^C$, the corresponding satisfaction times of clients satisfy

$$\|s(X,\pi)\| \leq \alpha \|s(X',\pi')\|.$$

We need one last piece of the framework to state algorithm `IterativeOrdering`. Given a γ-COMPOSABLE problem and some *budget* B, consider the satisfier $X' \subseteq \mathcal{X}$ and

Algorithm 4 `IterativeOrdering`(β)

input: A γ-COMPOSABLE problem and parameter $\beta \geq 1$
output: A satisfier $X \subseteq \mathcal{X}$ and order π on X such that $C(X) = C$

set $\theta = \sqrt{\gamma} + 1$
$j \leftarrow 0$
while $\bigcup_{l \in [0,j-1]} C(X_l) \neq C$ **do**
    set budget $B = \theta^j$
    find (β, B)-satisfier X_j and corresponding order π_j
    increase counter $j \leftarrow j + 1$
define satisfier $X = \bigcup_{i \in [0,j]} X_i$ and composed order $\pi \leftarrow \oplus_{i \in [0,j]} \pi_i$
return X and π

order π' on X' that satisfies as many clients $|C(X')|$ as possible under the cost constraint $c(X', \pi') \leq B$. Consider the following relaxation: given $\beta \geq 1$, we call another satisfier X and order π on X a (β, B)-satisfier if $c(X, \pi) \leq \beta B$ and $|C(X)| \geq |C(X')|$, i.e., (X, π) has cost within factor β of the budget B and satisfies at least as many clients as (X', π').

Of course, (X', π') (corresponding to $\beta = 1$) can always be found using an exhaustive search for any (finite) problem, but this search may take time exponential in the input size. For example, for COMPLETIONTIMES, this search for (X', π') for a given B amounts to searching over all possible partial assignments with makespan $\leq B$. As we show later, this is still useful in obtaining our results for the existence of simultaneous approximations. For many problems, choosing a larger β allows finding a (β, B)-satisfier in *polynomial-time*, e.g., $\beta = 2$ for ORDEREDTSP and COMPLETIONTIMES (see Section 5.2). This difference accounts for the gap between our approximations for existence and polynomial-time algorithms.

Algorithm `IterativeOrdering` is inspired by [26]'s algorithm for the Traveling Repairman Problem (TRP), which was subsequently also used for ORDEREDSETCOVER, ORDEREDVERTEXCOVER by [23], who also mention its applicability to similar covering problems. It takes as input a γ-composable problem with $\beta \geq 1$, and constructs a simultaneous $\beta(\sqrt{\gamma} + 1)^2$-approximation to the problem. Choosing $\beta = 1$ gives the existence results while choosing appropriate $\beta > 1$ gives polynomial-time results. We assume by

re-scaling all costs that the minimum non-zero cost $c(X, \pi)$ across satisfiers $X \subseteq \mathcal{X}$ and orders π on X is 1.

Lemma 5.4. *Given a* γ-COMPOSABLE *problem and* $\beta \geq 1$, `IterativeOrdering` *gives a simultaneous* $\left(\beta(\sqrt{\gamma}+1)^2\right)$*-approximation.*

Proof. Suppose there were k total iterations in `IterativeOrdering`; then the output satisfier is $X = \bigcup_{j\in[0,k]} X_j$ and corresponding order is $\pi = \bigoplus_{j\in[0,k]} \pi_j$.

Fix symmetric monotonic norm $\|\cdot\|$. Let the optimal solution for this norm be (X^*, π^*). We will show that for all $T > 0$, if (X^*, π^*) satisfies i clients within time T, then (X, π) satisfies $\geq i$ clients within time $\beta(\sqrt{\gamma}+1)^2 T$. Given corresponding satisfaction time vectors $s(X, \pi)$, $s(X^*, \pi^*) \in \mathbb{R}^C$; this is equivalent to saying that for any $i \in \{1, \ldots, |C|\}$, the ith smallest entry of $s(X, \pi)$ is at most $\beta(\sqrt{\gamma}+1)^2$ times the ith smallest entry of $s(X^*, \pi^*)$. Since $\|\cdot\|$ is symmetric and monotone, this implies that (X, π) is a $\beta(\sqrt{\gamma}+1)^2$-approximation.

Given $T > 0$, define $X_T^* := \{x \in X^* : t(X^*, \pi^*)_x \leq T\}$, and let π_T^* be the restriction of π^* to X_T^*. Then, by Lemma 5.3,

$$c(X_T^*, \pi_T^*) \leq T. \tag{5.5}$$

Also by the same lemma,

$$|C(X_T^*)| \geq |\{e \in C(X^*) : s(X^*, \pi^*)_e \leq T\}| := i. \tag{5.6}$$

Let $j \in \mathbb{Z}_{\geq 0}$ be the unique integer such that $T \in (\theta^{j-1}, \theta^j]$. Then, by definition of a (β, B)-satisfier, in iteration j of the algorithm, we get (X_j, π_j) such that (a) $c(X_j, \pi_j) \leq \beta\theta^j$, and (b) $|C(X_j)| \geq |C(X_T^*)| \geq i$.

For all clients $e \in C(X_j)$, γ-composability implies that the satisfaction time $s(X, \pi)_e \leq \gamma\left(\sum_{l\in[0,j-1]} c(X_l, \pi_l)\right) + c(X_j, \pi_j)$. Since the cost $c(X_l, \pi_l) \leq \beta\theta^l$ for all $l \in [0, k]$, we

have

$$s(X,\pi)_e \leq \gamma\left(\sum_{l\in[0,j-1]}\beta\theta^l\right)+\beta\theta^j \leq \beta\left(\gamma\frac{\theta^j}{\theta-1}+\theta^j\right)=\theta^{j-1}\times\beta\theta\left(\frac{\gamma}{\theta-1}+1\right).$$

Therefore, (X,π) satisfies at least $|C(X_j)|\geq i$ clients within time $\beta\theta\left(\frac{\gamma}{\theta-1}+1\right)\times\theta^{j-1}<\beta\theta\left(\frac{\gamma}{\theta-1}+1\right)\times T$. Since $\theta=\sqrt{\gamma}+1$, we have $\theta\left(\frac{\gamma}{\theta-1}+1\right)=(\sqrt{\gamma}+1)^2$. □

This leads to the following results proving the existence of various simultaneous approximations, and a polynomial-time 8-approximation for COMPLETIONTIMES:

Theorem 5.1. *1. For any γ-*COMPOSABLE *problem, there always exists a simultaneous $(\sqrt{\gamma}+1)^2$-approximation.*

2. For ORDEREDSETCOVER*,* ORDEREDVERTEXCOVER*, and* COMPLETIONTIMES*, there always exists a simultaneous 4-approximation.*

3. For ORDEREDTSP*, there always exists a simultaneous $(3+2\sqrt{2})$-approximation.*

4. For COMPLETIONTIMES*, a simultaneous 8-approximation can be found in polynomial-time.*

Part 1 of the theorem follows by choosing $\beta = 1$ in Lemma 5.4 and parts 2 and 3 follow from our observations in Lemma 5.2 that ORDEREDSETCOVER, ORDEREDVERTEXCOVER, and COMPLETIONTIMES are 1-composable while ORDEREDTSP is 2-composable.

The proof of the last part involves giving a subroutine for COMPLETIONTIMES that outputs a $(2,B)$-satisfier for each budget $B>0$. This is equivalent to asking the following: given a time limit B, schedule as many given jobs as possible on the machines. We show that [106]'s 2-approximation for makespan minimization generalizes to this setting (proof of the lemma in Section 5.2):

Lemma 5.5. *Given processing times $p_{i,j}$ for jobs $j \in [n]$ on machines $i \in [d]$, and a time budget B, there exists a polynomial-time algorithm to find a partial schedule that (1) finishes within time $2B$, (2) schedules at least as many jobs as any partial schedule that finishes within time B.*

5.2 Omitted Proofs from Section 5.1

We complete the proof of various lemmas in Section 5.1 on `IterativeOrdering`.

Proof of Lemma 5.1. The proof for COMPLETIONTIMES was supplied in the previous section. Further, ORDEREDVERTEXCOVER is a special case of ORDEREDSETCOVER. Therefore, it suffices to complete the proof for ORDEREDSETCOVER and ORDEREDTSP.

- ORDEREDSETCOVER. Given the ground set $E = \{e_1, \ldots, e_n\}$ and subsets $S_1, \ldots, S_m \subseteq E$, choose the set of clients C as the ground set E, the set of objects $\mathcal{X}$ as the set $\{S_1, \ldots, S_m\}$ of subsets, with $C(S_i) = S_i$ for all i. Given any satisfier $X \subseteq \mathcal{X}$ and order π on X, define the time $t(X, \pi)_{S_i}$ for $S_i \in X$ to be the position $\pi(S_i)$ of S_i in π. Then the satisfaction time of an element $e \in C(X) = \bigcup_{S \in X} S$ is precisely the cover time of the element.

Further, given a $T > 0$, X_T is simply the first T subsets in X according to order π, and clearly, $t(X_T, \pi_T)_{S_i} = t(X, \pi)_{S_i} = \pi(S_i)$ for all such subsets $S_i \in X_T$. This proves downward closure.

- ORDEREDTSP. Given a metric on vertices V and starting vertex v_0, choose the set of clients C as V, and the set of objects $\mathcal{X}$ as V also, with $C(v) = \{v\}$ for all $v \in \mathcal{X}$. Given any satisfier $X \subseteq \mathcal{X}$, any order π on X corresponds to a path consisting of the vertices of X. We define the time $t(X, \pi)_v$ as follows: if π does not start at v_0 or if $v_0 \notin X$, then $t(X, \pi)_v = \infty$ for all $v \in X$. This is to disallow paths that do not start at the starting vertex v_0. If π starts at v_0, then define $t(X, \pi)_v$ to be the length of the path from v_0 to v. Since $C(v) = \{v\}$ for all $v \in V$, the satisfaction time of a vertex $v \in X$ is the same as $t(X, \pi)_v$.

We prove downward closure next: given a $T > 0$ and (X, π), if π does not start at v_0, then $X_T = \emptyset$, and downward closure holds trivially. Otherwise, X_T is precisely the set of vertices within distance T of the starting vertex v_0 along path π, and $t(X_T, \pi_T)_v = t(X, \pi)_v$ for all $v \in X_T$. □

Proof of Lemma 5.2. As before, it suffices to complete the proof for ORDEREDSETCOVER and ORDEREDTSP.

- ORDEREDSETCOVER. Given satisfier X of subsets of the ground set and order π on X, the cost $c(X, \pi) = \max_{S_i \in X} \pi(S_i) = |X|$ is simply the size of X. Consider satisfiers $X_1, \ldots, X_k$, corresponding orders $\pi_1, \ldots, \pi_k$, and some $S \in X_j \setminus (X_1 \cup \cdots \cup X_{j-1})$. For the composed satisfier $X = \bigcup_{l \in [k]} X_l$, and the composed order $\pi = \bigoplus_{l \in [k]} \pi_l$, we have that $t(X, \pi)_S = \pi(S)$ is the position of S in the order when all subsets in X_1 are ordered first, all subsets in $X_2 \setminus X_1$ are ordered next, and so on. Therefore,

$$t(X, \pi)_S \leq |X_1| + \cdots + |X_{j-1}| + \pi_j(S) = c(X_1, \pi_1) + \cdots + c(X_{j-1}, \pi_{j-1}) + t(X_j, \pi_j)_S.$$

ORDEREDTSP. Given satisfier $X \subseteq V$ and path π on X, the cost $c(X, \pi) = \infty$ if the path does not start at v_0 and $c(X, \pi)$ is the length of the path otherwise. Composing paths $\pi_1, \ldots, \pi_k$ on vertex sets $X_1, \ldots, X_k$ respectively that each starts at v_0 amounts to the following: start at v_0, complete path π_1, and return to v_0, the complete path π_2 and return to v_0 again, and so on, shortcutting any vertices visited a second time.

Then, given a vertex v visited in path π_j, the length of the path from v_0 to v in this composed path is at most $2(\text{length}(\pi_1) + \cdots + \text{length}(\pi_{j-1})) + \text{length from } v_0 \text{ to } v \text{ in } \pi_j$, which is precisely $2\left(\sum_{l \in [j-1]} c(X_l, \pi_l)\right) + t(X_j, \pi_j)_v$. □

Proof of Lemma 5.3. For part 1, by definition, $c(X_T, \pi_T) = \max_{x \in X_T} t(X_T, \pi_T)_x$. By downward closure, $t(X_T, \pi_T)_x \leq t(X, \pi)_x$ for all $x \in X_T$. However, X_T was defined as $\{x \in X : t(X, \pi)_x \leq T\}$, and thus $c(X_T, \pi_T) \leq T$.

Part 2: for each client e satisfied within time T by (X, π), by definition of satisfaction

time there is some object $x \in X$ with $t(X,\pi)_x \leq T$. Therefore, $x \in X_T$ and so $e \in C(X_T)$, i.e., $|C(X_T)|$ is at least the number of clients satisfied by (X,π) within time t. □

Proof of Lemma 5.5. Given processing times p and budget $B \geq 0$, consider the following linear programming relaxation of the problem:

$$\max \sum_{i,j} x_{i,j} \qquad \text{s.t.} \qquad \text{(LP-PS)}$$

$$\sum_j p_{i,j} x_{i,j} \leq B \qquad \forall\, i \in [d], \qquad (5.7)$$

$$\sum_i x_{i,j} \leq 1 \qquad \forall\, j \in [n], \qquad (5.8)$$

$$x_{i,j} = 0 \qquad \text{if } p_{i,j} > B\ \forall\, i,j, \qquad (5.9)$$

$$x \geq 0.$$

Variable $x_{i,j}$ indicates whether or not job j has been assigned to machine i. The objective is to maximize the number of jobs scheduled under the constraint that the makespan is at most B. However, to ensure that the optimal solution does not schedule a cheap job multiple times, we include (Equation 5.8). Further, job j should not be scheduled on machine i if $p_{i,j}$ exceeds the makespan B (Equation 5.9). The optimal solution OPT to the partial scheduling problem clearly satisfies these constraints, and therefore $\mathrm{OPT} \leq \sum_{i,j} x^*_{i,j}$ for the (fractional) optimal solution x^* to the LP.

We will round x^* to an integral solution x with makespan $\leq 2B$ and $\sum_{i,j} x_{i,j} \geq \sum_{i,j} x^*_{i,j}$ implying that x schedules at least as many jobs as OPT, completing the proof.

Let $k_i = \lceil \sum_j x^*_{i,j} \rceil$ for all i. We will construct an undirected bipartite graph G with $n + k_1 + \cdots + k_d$ vertices: n vertices correspond to jobs and k_i vertices correspond to machine i for all i.

Fix machine i. Let $J_i = \{j : x^*_{i,j} > 0\}$ be the set of jobs (fractionally) assigned to i under x^*, and relabel them so that $J_i = \{1, 2, \ldots, l\}$; assume without loss of generality that

$p_{i,1} \geq \cdots \geq p_{i,l}$. Let $v_1, \ldots, v_{k_i}$ be the vertices corresponding to machine i. Start assigning weights $x^*_{i,1}, x^*_{i,2}, \ldots$ to edges $v_1 1, v_1 2, \ldots$, until we reach a job a such that $x^*_{i,1} + x^*_{i,2} + \cdots + x^*_{i,a} > 1$. Assign weight $1 - \sum_{b \leq a-1} x^*_{i,b}$, i.e., just enough weight that makes the total weight of edges incident to v_1 exactly 1. The remaining weight for job a, $\sum_{b \leq a} x^*_{i,b} - 1$ goes to edge $v_2 a$. Continue this process with job $a+1$ on vertex v_2, and so on. Since $\sum_{j \in J_i} x^*_{i,j} \leq k_i$, weight $x^*_{i,l}$ is assigned to edge $v_{k_1} l$. Notice that for each of $v_1, \ldots, v_{k_i}$, the sum of weights of edges incident on it is at most 1. Do this for all vertices to get graph G, and denote the weights in G by w.

By construction, the sum of weights of edges incident on a vertex is at most 1 if it corresponds to a machine. From Equation 5.8 and the construction, the sum of weights of edges incident on vertices corresponding to jobs is also at most 1. Therefore, w forms a fractional matching on G. Further, the sum of all edge weights, $\|w\|_1$, is $\sum_{i,j} x^*_{i,j}$. Since G is bipartite, this fractional matching can be rounded to an integral matching y at least as large as w, i.e., $\|y\|_1 \geq \|w\|_1$. Obtain integral solution x by assigning jobs to machines according to matching y, i.e., if job j is adjacent to a vertex corresponding to machine i, assign $x_{i,j} = 1$; assign $x_{i,j} = 0$ in all other cases. Then we have that $\sum_{i,j} x_{i,j} = \|y\|_1 \geq \|w\|_1 = \sum_{i,j} x^*_{i,j}$.

It remains to argue that the makespan to each machine is at most $2B$. Fix machine i. Suppose jobs $j_1, \ldots, j_{k_i}$ are adjacent to vertices $v_1, \ldots, v_{k_i}$ respectively in matching y. Then, since jobs were sorted in decreasing order, the processing time p_{i,j_2} is upper bounded by the processing time of jobs adjacent to v_1 in G:

$$p_{i,j_2} = p_{i,j_2} \sum_{j : w(v_1 j) > 0} w(v_1, j) \leq \sum_{j : w(v_1 j) > 0} p_{i,j} w(v_1, j).$$

Similarly, for each $b \in [2, j_{k_i}]$, we get $p_{i,j_b} \leq \sum_{j : w(v_{b-1,j}) > 0} p_{i,j} w(v_{b-1}, j)$. Adding these,

$$\sum_{a \in [2, k_i]} p_{i,j_a} \leq \sum_{a \in [2,k_i]} \sum_{j : w(v_{b-1,j}) > 0} p_{i,j} w(v_{b-1}, j) \leq \sum_{j \in J_i} p_{i,j} x^*_{i,j} \leq B.$$

Since $p_{i,j_1} \leq B$ by Equation 5.7, we get the total makespan on machine i under x is

$$p_{i,j_1} + \sum_{a \in [2,k_i]} p_{i,j_a} \leq 2B. \qquad \square$$

5.3 Lower Bounds for Simultaneous Approximations

We give two lower bounds on best-possible simultaneous approximations here, for ORDEREDVERTEXCOVER and COMPLETIONTIMES, respectively.

Observation 5.1. *There exists an instance of* ORDEREDVERTEXCOVER *where no solution is better than* $9/8$*-simultaneous approximate for the* L_1 *and* L_∞ *norms (i.e., Min-Sum Vertex Cover and classical Vertex Cover).*

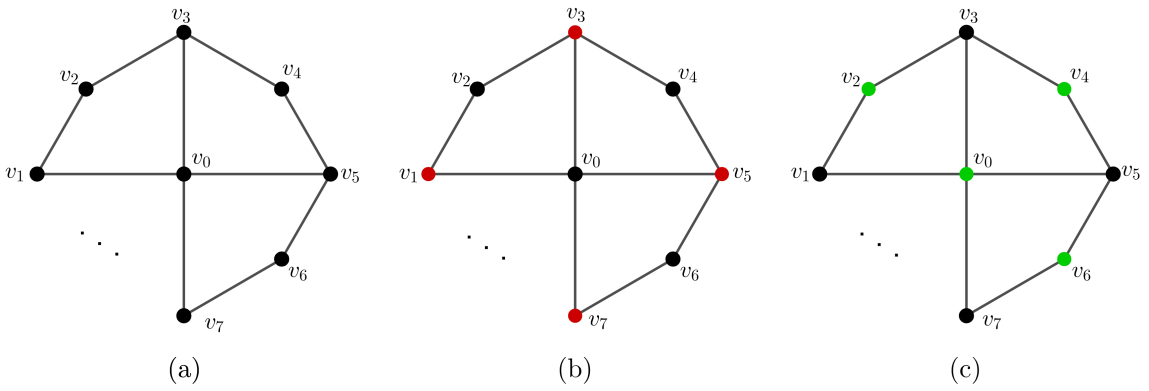


Figure 5.1: The vertex cover instance used in proof of Observation 5.1.

Proof. Consider the following instance: the graph as $2n+1$ vertices $v_0, \ldots, v_{2n}$ with vertices $v_1, \ldots, v_{2n}$ forming cycle and vertex v_0 connected to each of $v_1, v_3, \ldots, v_{2n-1}$ (Figure 5.1(a)).

The smallest vertex cover is $\{v_1, v_3, \ldots, v_{2n-1}\}$ (Figure 5.1(b)), and it is the only vertex cover of size n. Therefore, any other vertex cover is at best a $\frac{n+1}{n}$-approximation. When $n = 8$, this is $9/8$.

We show that this vertex cover is a $9/8$-approximation for MSVC when $n = 8$. Irrespective of the order of the vertices in this vertex cover, exactly 3 edges are covered by each

time step. Therefore, the total cover time of the edges is $3 \times (1 + \ldots + n) = \frac{3}{2}n(n+1)$. When $n = 8$, this is 108.

However, if we instead use the cover $(v_0, v_1, v_3, \ldots, v_{2n-1})$ (Figure 5.1(c)) in this order, n edges are covered at the first step, and 2 edges are covered in each subsequent step, resulting in total cover time of $n + 2(2 + \ldots + (n+1)) = n(n+4)$. When $n = 8$, this is $96 = \frac{8}{9} \times 108$. □

Next, we show a similar bound for COMPLETIONTIMES:

Observation 5.2. *There exists an instance of* COMPLETIONTIMES *where no solution is better than* 1.13*-simultaneous approximate for the* L_1 *and* L_∞ *norms (i.e., average completion time minimization and makespan minimization).*

Proof. Consider an instance with two machines (labeled A, B) and three jobs. Let $\mu, \delta \in [0, 1)$ be parameters we fix later. Jobs $1, 2$ both have processing time 1 on machine A and processing time $1 + \delta$ on machine B. Job 3 has processing time $1 + \mu$ on machine A and 2 on machine B.

Consider solutions where jobs $1, 2$ are on different machines. Then, the optimal solution (for both makespan minimization and average completion time minimization) is to place job 3 is on machine A. The makespan for this solution is $2 + \mu$, and the total completion time is $1 + (2 + \mu) + (1 + \delta) = 4 + \mu + \delta$:

$$MS_1 = 2 + \mu, \quad CT_1 = 4 + \mu + \delta.$$

Suppose jobs $1, 2$ are both on machine A now. The optimal solution (for both makespan and total completion time) is to place job 3 is on machine B. The makespan and total completion time are respectively

$$MS_2 = 2, \quad CT_2 = 5.$$

Suppose jobs $1, 2$ are both on machine B. The optimal solution is to place job 3 is on machine A. The makespan and total completion time are:

$$MS_3 = 2(1+\delta), \quad CT_3 = 3(1+\delta) + (1+\mu) = 4 + \mu + 3\delta.$$

Therefore, when $\mu + \delta \leq 1$, the second solution has optimal makespan 2 and the first solution has the optimal average completion time. The simultaneous approximation ratio of the first solution is $\frac{2+\mu}{2}$. The simultaneous approximation ratio of the second solution is $\frac{5}{4+\mu+\delta}$. The simultaneous approximation ratio of the third solution is $\max\left(1+\delta, \frac{4+\mu+3\delta}{4+\mu+\delta}\right)$. The best possible simultaneous approximation ratio then is

$$\min\left(\frac{2+\mu}{2}, \frac{5}{4+\mu+\delta}, \max\left(1+\delta, \frac{4+\mu+3\delta}{4+\mu+\delta}\right)\right).$$

Maximizing this over all μ, δ such that $0 \leq \mu, \delta$ and $\mu + \delta \leq 1$, we get the value $\frac{\sqrt{61}-1}{6} > 1.13$ at $(\mu, \delta) = \left(\frac{\sqrt{61}-7}{3}, \frac{\sqrt{61}-7}{6}\right)$. □

5.4 k-CLUSTERING and UNCAPACITATEDFACILITYLOCATION

In this section, we consider k-CLUSTERING and UNCAPACITATEDFACILITYLOCATION that we formally defined in Section 2.2 and studied in Chapter 3. Recall that we are given a metric space $(C \cup F, \texttt{dist})$ on $|C \cup F| = n$ points with clients C and potential open facilities F, and are required to choose a subset $F' \subseteq F$ of them. The induced distance vector $x^{F'} \in \mathbb{R}^C$ is defined as the vector of distances between point j and its nearest open facility, i.e., $x^{F'}_j = \min_{f \in F'} \texttt{dist}(j, f)$ for all $j \in X$. Given a norm $\|\cdot\|$ on $\mathbb{R}^C$, k-CLUSTERING seeks to open a set F' of at most k facilities to minimize $\|x^{F'}\|$, while UNCAPACITATEDFACILITYLOCATION allows any number of facilities to open but penalizes the number of open facilities through the combined objective function $|F'| + \|x^{F'}\|$.[3]

[3]Note that we allowed for facility opening costs in the definition of UNCAPACITATEDFACILITYLOCATION in Section 2.2 and Chapter 3; here we restrict to the uniform cost case, where each facility has cost 1 to open.

For k-CLUSTERING, we consider more general *bicriteria* (α, β)-approximations with objective value within factor α of the optimum but that violate the bound on the number of open facilities by a factor β. [23] show that any solution to k-CLUSTERING that is simultaneously $O(1)$-approximate for symmetric monotonic norms must open at least $\Omega(k \log n)$ facilities, i.e., violate the size bound by factor $\beta = \Omega(\log n)$.

Fix any $\varepsilon \in (0, 1]$. Using ideas similar to `IterativeOrdering`, we give the algorithm `IterativeClustering` that finds a solution with at most $O\left(\frac{k \log n}{\varepsilon}\right)$ open facilities that is simultaneously $(1+\varepsilon)$-approximate for all symmetric monotonic norms, matching the result of [9]. In polynomial time, `IterativeClustering` finds a solution that is $(3 + \varepsilon)$-approximate, improving the previous $(6 + \varepsilon)$-approximation of [9].

We remark that – as pointed out to us by a reviewer from STOC 2024 – carefully combining the rounding techniques from [106] and the linear program for top-ℓ norm minimization from [20] matches our polynomial-time bound, and gives an even better $(2 + \varepsilon)$-approximation if facilities are also allowed to open in client set C. Our algorithm generalizes to the setting when $C \not\subseteq F$ and is a natural extension of the `IterativeOrdering` framework that emphasizes common structure across different combinatorial problems.

We also show that the above result for k-CLUSTERING leads to an $O(\log n)$-approximate portfolio of size $O(\log n)$ for UNCAPACITATEDFACILITYLOCATION, the first such result for symmetric monotonic norms to our knowledge.

5.4.1 k-CLUSTERING

We prove the following result:

Theorem 5.2. *For k-CLUSTERING, Algorithm `IterativeClustering` gives*

1. *a simultaneous bicriteria $\left(1 + \varepsilon, O\left(\frac{\log n}{\varepsilon}\right)\right)$-approximation in finite time, and*
2. *a simultaneous bicriteria $\left(3 + \varepsilon, O\left(\frac{\log n}{\varepsilon}\right)\right)$-approximation can in polynomial-time.*

Algorithm 5 `PartialClustering`$((C \cup F, \texttt{dist}), k, R, \alpha)$

input A metric space $(C \cup F, \texttt{dist})$, integer $k \geq 1$, radius $R \geq 0$, parameter $\alpha \geq 1$
output A set $G \subseteq F$ of k facilities that contains at least as many clients within distance αR as contained by any other set $G' \subseteq F$ of k facilities within distance R, i.e.,

$$|B(G, \alpha R)| \geq \max_{G' \in \binom{F}{k}} |B(G', R)| .$$

Algorithm 6 `IterativeClustering`$((C \cup F, \texttt{dist}), k, \varepsilon, \alpha)$

input A metric space $(C \cup F, \texttt{dist})$ on n points, integer $k \geq 1$, parameter $\varepsilon > 0$, parameter $\alpha \geq 1$
output A set $F' \subseteq F$ of $O\left(\frac{k \log n}{\varepsilon}\right)$ facilities

$F' \leftarrow \emptyset$
$R_0 = \frac{D\varepsilon}{n}$, where D is the k-center optimum for $(C \cup F, \texttt{dist})$
for $l = 0, 1, \ldots, \log_{1+\varepsilon}(n/\varepsilon)$ **do**
    $R \leftarrow R_0(1+\varepsilon)^l$
    $F_l \leftarrow$ `PartialClustering`$((C \cup F, \texttt{dist}), k, R, \alpha)$
    $F' \leftarrow F' \cup F_l$
return F'

Broadly, `IterativeClustering` iteratively combines solutions that each contain k facilities. Each of these solutions corresponds to a radius R, and subroutine `Partial-Clustering` attempts to get the set of k facilities that covers the largest number of points within radius R. As with `IterativeOrdering`, radius R increases exponentially across iterations.

For polynomial-time computations, `PartialClustering` cannot be solved exactly since it generalizes the k-center problem. To get efficient algorithms, we allow it to output k facilities that cover as many points within radius αR as those covered by any k facilities within radius R. As [8] note, [117] give an approximation algorithm for `Partial-Clustering` for $\alpha = 3$, which we state in a modified form:

Lemma 5.6 (Theorem 3.1, [117])**.** *There exists a polynomial-time algorithm that given metric $(C \cup F, \texttt{dist})$, integer $k \geq 1$, and radius R, outputs k facilities that cover at least as many points within radius $3R$ as those covered by any set of k facilities within radius R. That is, subroutine* `PartialClustering` *runs in polynomial-time for $\alpha = 3$.*

We give some notation: given nonempty $F' \subseteq F$ and some radius $R \geq 0$, we denote by $B(F'; R)$ the set of all clients within distance R of F', i.e., $B(F'; R) = \{j \in C : \exists\, f \in F' \text{ with } \texttt{dist}(j, f) \leq R\}$. We say that a set of facilities F' *covers* p points within radius R if $|B(F'; R)| \geq p$.

Let D denote the k-center optimum for $(C \cup F, \texttt{dist})$. By definition, there are k facilities that can cover all of C within radius D. Therefore, the largest radius we need to consider is D. What is the smallest radius we need to consider? Since all of our objective norms are monotonic and symmetric, points covered within very small radii do not contribute a significant amount to the norm value. Therefore, we can start at a large enough radius, which has been set to $\frac{D\varepsilon}{n}$ with some foresight. We will first prove the following claim:

Claim 5.1. *For parameter $\alpha \geq 1$,* `IterativeClustering` *gives a simultaneous bicriteria $\left(\alpha(1+2\varepsilon), O\left(\frac{\log n}{\varepsilon}\right)\right)$-approximation for symmetric monotonic norms.*

Proof. We first show that the number of facilities output by the algorithm is $O\left(\frac{k \log n}{\varepsilon}\right)$. The number of iterations in the for loop is $\log_{(1+\varepsilon)}\left(\frac{n}{\varepsilon}\right) = O\left(\frac{\log n}{\varepsilon} + \frac{\log(1/\varepsilon)}{\varepsilon}\right)$. When $\varepsilon > \frac{1}{n}$, this expression is $O\left(\frac{\log n}{\varepsilon}\right)$. Since each iteration adds at most k facilities to C, we are done in this case. When $\varepsilon \leq \frac{1}{n}$, then $\frac{k \log n}{\varepsilon} \geq n$, that is, all facilities can be opened anyway.

Fix any symmetric monotonic norm $\|\cdot\|$ on $\mathbb{R}^C$, and let OPT denote the optimal solution for this norm and $x^{\mathrm{OPT}} \in \mathbb{R}^C$ denote the corresponding distance vector. Let the distance vector for facilities C output by the algorithm be x. We need to show that $\|x\| \leq \alpha(1 + 2\varepsilon)\|x^{\mathrm{OPT}}\|$.

By definition, $(x^{\mathrm{OPT}})^{\uparrow}_1 \leq (x^{\mathrm{OPT}})^{\uparrow}_2 \leq \cdots \leq (x^{\mathrm{OPT}})^{\uparrow}_{|C|}$. Let j^* be the smallest index such that $(x^{\mathrm{OPT}})^{\uparrow}_{j^*} > R_0 = \frac{D\varepsilon}{n}$. Since $\|\cdot\|$ is symmetric, we have $\|x^{\uparrow}\| = \|x\|$ and $\|(x^{\mathrm{OPT}})^{\uparrow}\| = \|x^{\mathrm{OPT}}\|$. Our twofold strategy is to show that:

1. for all $j \geq j^*$,

$$(x)^{\uparrow}_j \leq \alpha(1+\varepsilon)(x^{\mathrm{OPT}})^{\uparrow}_j, \tag{5.10}$$

2. the contribution of $x^{\uparrow}_1, \ldots, x^{\uparrow}_{j^*-1}$ to $\|x\|$ is small; specifically,

$$\left\| \left(x^{\uparrow}_1, \ldots, x^{\uparrow}_{j^*-1}, 0, \ldots, 0 \right) \right\| \leq \alpha\varepsilon \|x^{\mathrm{OPT}}\|. \tag{5.11}$$

Consider the first part. We have $R_0(1+\varepsilon)^{\log_{1+\varepsilon}(n/\varepsilon)} = R_0 \frac{n}{\varepsilon} = D$. That is, in the final iteration of the for loop, $R = D$. Therefore, by definition of D and `PartialClustering`, F_l in this iteration covers all of X within radius αD. That is, $\|x\|_\infty \leq \alpha D$ since $F_l \subseteq F'$.

fix some $j \geq j^*$, and let $l \geq 0$ be the smallest integer such that $(x^{\mathrm{OPT}})^{\uparrow}_j \leq R_0(1+\varepsilon)^l$. If $l \geq 1 + \log_{1+\varepsilon}(n/\varepsilon)$, then $(x^{\mathrm{OPT}})^{\uparrow}_j > R_0(1+\varepsilon)^{l-1} = D$. Since $\|x\|_\infty \leq \alpha D$, Equation 5.10 holds in this case.

Otherwise, $l \leq \log_{1+\varepsilon}(n/\varepsilon)$. The k facilities in OPT cover at least j points within radius $R = R_0(1+\varepsilon)^l$. By definition of `PartialClustering`, in iteration l of the for loop, F_l covers at least j points within radius αR. Since $F_l \subseteq F'$, F' also covers at least j points within radius αR, so that $x^{\uparrow}_j \leq \alpha R = R_0(1+\varepsilon)^l$. By definition of l, $(x^{\mathrm{OPT}})^{\uparrow} > R_0(1+\varepsilon)^{l-1}$, and so

$$x^{\uparrow}_j \leq \alpha R_0 (1+\varepsilon)^l \leq \alpha(1+\varepsilon)(x^{\mathrm{OPT}})^{\uparrow}_j.$$

We move to Equation 5.11. By definition of j^*, OPT covers at least $j^* - 1$ points within radius R_0. In iteration 0, by definition of `PartialClustering`, F_0 (and therefore F') covers at least $(j^* - 1)$ points within radius αR_0. That is, $x^{\uparrow}_{j^*-1} \leq \alpha R_0$.

Denote $(1, 0, \ldots, 0) = \mathbf{e}$. Since $\|\cdot\|$ is monotonic and D is the k center optimum, $\|x^{\mathrm{OPT}}\| \geq \left(\|x^{\mathrm{OPT}}\|_\infty, 0, \ldots, 0\right) \|\mathbf{e}\| \geq D\|\mathbf{e}\|$. Therefore,

$$\begin{aligned}
\left\| \left(x^{\uparrow}_1, \ldots, x^{\uparrow}_{j^*-1}, 0, \ldots, 0 \right) \right\| &\leq \sum_{j \in [j^*-1]} x^{\uparrow}_j \|\mathbf{e}\| && \text{(triangle inequality)} \\
&\leq \sum_{j \in [j^*-1]} \alpha R_0 \|\mathbf{e}\| && (x^{\uparrow}_{j^*-1} \leq \alpha R)
\end{aligned}$$

$$< n\alpha \frac{D\varepsilon}{n} \|\mathbf{e}\| \qquad (j^* \le n)$$
$$\le \alpha\varepsilon \|x^{\mathrm{OPT}}\|. \qquad (\|x^{\mathrm{OPT}}\| \ge D\|\mathbf{e}\|)$$

Together, Equation 5.10 and Equation 5.11 imply that

$$\begin{aligned}\|x\| &\le \left\| \left(x_1^{\uparrow}, \ldots, x_{j^*-1}^{\uparrow}, 0, \ldots, 0 \right) \right\| + \left\| \left(0, \ldots, 0, x_{j^*}^{\uparrow}, \ldots, x_n^{\uparrow} \right) \right\| \\ &\le \alpha\varepsilon \|x^{\mathrm{OPT}}\| + \alpha(1+\varepsilon) \left\| \left(0, \ldots, 0, (x^{\mathrm{OPT}})_{j^*}^{\uparrow}, \ldots, (x^{\mathrm{OPT}})_n^{\uparrow} \right) \right\| \\ &\le \alpha\varepsilon \|x^{\mathrm{OPT}}\| + \alpha(1+\varepsilon)\|x^{\mathrm{OPT}}\| = \alpha(1+2\varepsilon)\|x^{\mathrm{OPT}}\|.\end{aligned}$$

The first inequality is the triangle inequality, the second follows from Equation 5.10 and Equation 5.11, and the last inequality follows since $\|\cdot\|$ is symmetric monotonic. □

With this result in hand, our main theorem is simple to derive: we choose $\alpha = 1$ in the claim with $\varepsilon/2$ as the parameter for the existence result. We choose $\alpha = 3$ in the claim with $\varepsilon/6$ as the parameter for the polynomial-time result; Lemma 5.6 guarantees that the algorithm is polynomial-time.

5.4.2 UncapacitatedFacilityLocation

First, we note that a single solution cannot be better than $\Omega(\sqrt{n})$-approximate for even the L_1 and L_∞ norms: suppose the metric is a star metric with n, with $C = F$ so that $|C| = |F| = n$. The distance from the center to each leaf is $\sqrt{n}$. Then the optimal L_1 solution is to open each facility, and the cost of this solution is $n + 1$. The optimal L_∞ solution is to open just one facility at the center, the cost of this solution is $1 + \sqrt{n}$. Now, any solution that opens fewer than $n/2$ facilities has cost $\ge n/2 + (n/2)\sqrt{n} = \Omega(n\sqrt{n})$ for the L_1 norm and therefore is an $\Omega(\sqrt{n})$-approximation. Any solution that opens $\ge n/2$ facilities is an $\Omega(\sqrt{n})$-approximation for the L_∞ norm. A similar example was noted for k-clustering in [9].

This motivates us to seek larger portfolios and get a smaller approximation. The main

theorem of this section gives an $O(\log n)$-approximate portfolio of size $O(\log n)$ for UNCAPACITATEDFACILITYLOCATION:

Theorem 5.3. *There exists a polynomial-time algorithm that given any instance of* UNCAPACITATEDFACILITYLOCATION *on n points, outputs an $O(\log n)$-approximate portfolio of size $O(\log n)$ for symmetric monotonic norms.*

Proof. Assume without loss of generality that the number of points n is a power of 2. Choose solutions corresponding to $k = 2^0, 2^1, 2^2, \ldots, 2^{\log_2 n}$ with $\varepsilon = 1$ in Theorem 5.2 part 2. There are $O(\log n)$ of these, and the theorem asserts that they can be found in polynomial time. We claim that these form an $O(\log n)$-approximate portfolio for symmetric monotonic norms.

Fix a symmetric monotonic norm $\|\cdot\|$, and suppose the optimal solution OPT for this norm opens $k^* \in [n]$ facilities. Let l be the unique integer such that $2^{l-1} < k^* \leq 2^l$, i.e., $l = \lceil \log_2 k^* \rceil$. We show that the solution corresponding to $k = 2^l$ in our portfolio is an $O(\log n)$-approximation for $\|\cdot\|$. Add arbitrary $2^l - k^*$ facilities to OPT; this only decreases the induced distance vector x^{OPT}. For this new set of facilities, Theorem 5.2 guarantees that $\|x\| \leq 4\|x^{\mathrm{OPT}}\|$. Therefore, the objective value of the portfolio solution is

$$O(\log n) \cdot 2^l + \|x\| = O(\log n) \left(k^* + \|x^{\mathrm{OPT}}\|\right) = O(\log n) \cdot \mathrm{OPT}. \qquad \square$$

5.5 Conclusion

We presented a unified framework for simultaneous approximations for combinatorial optimization problems, and gave several new and improved approximation algorithms using our framework. Several questions remain open:

Determining best-possible simultaneous approximations. For ORDEREDTSP, it is unlikely that our simultaneous 5.83-approximation is the best-possible, since the only known lower bound on this number is 1.78 [27]. It would be interesting to close this gap in either

direction. Similarly, it is unclear if our simultaneous 4-approximation for COMPLETION-TIMES or for ORDEREDSETCOVER is tight.

Gap between computability and existence. For simultaneous approximations, there is also a gap between existence bounds and polynomial-time bounds (see Table Table 5.1). For ORDEREDSETCOVER, this gap (factor 4 vs $O(\log n)$, respectively) is explained by complexity theoretic conjectures; however, it is unclear why this gap exists for other problems, such as ORDEREDTSP (factor 5.83 vs 8 respectively) and COMPLETIONTIMES (factor 4 vs 8 respectively).

CHAPTER 6

MAXIMIZATION PORTFOLIOS, p-MEANS, AND REINFORCEMENT LEARNING

6.1 Introduction

In this chapter, we study portfolios for p-mean functions $M_p(z) := \left(\frac{1}{d}\sum_{i\in[d]} z_i^p\right)^{1/p}$ for each $p \leq 1$ and $z \in \mathbb{R}^d_{>0}$. In particular, we study a reinforcement learning (RL) setting where a deployed policy impacts multiple stakeholders in different ways. Each stakeholder is associated with a unique reward function, and the goal is to train a policy that adequately aggregates their preferences.

This setting is often modeled using Multi-objective reinforcement learning (MORL) and arises in many RL applications, such as fair resource allocation in healthcare [40], cloud computing [119, 120], and communication networks [121, 122]. Recently, with the rise of large language models (LLMs), Reinforcement learning with human feedback (RLHF) techniques that reflect the preferences of heterogeneous individuals have also been explored [123, 124, 125].

Preference aggregation in such scenarios is often achieved by choosing a social welfare function, which takes the utilities of multiple stakeholders as input and outputs a scalar value representing the overall welfare [126, 42, 40, 41, 124, 125, 123]. However, selecting the appropriate social welfare function is a nontrivial task. Different functions encode distinct fairness criteria, and the resulting policies can lead to vastly different outcomes for stakeholders depending on the choice of the social welfare function.

We focus on p-means, which are a widely used class of social welfare functions in al-

This chapter is based on joint work with Cheol Woo Kim, Shresth Verma, Madeleine Pollack, Lingkai Kong, Milind Tambe, and Swati Gupta. A version of this chapter appeared in the *Proceedings of the Forty-Second International Conference on Machine Learning (ICML) 2025* [118]. Cheol Woo Kim is an equal contribution lead co-author.

gorithmic fairness and social choice theory. Each choice of p represents a distinct notion of fairness, and the p-means unify commonly used welfare functions such as egalitarian welfare ($p = -\infty$), Nash welfare ($p = 0$), and utilitarian welfare ($p = 1$), providing a smooth transition between these principles. Notably, this is known to be the only class of social welfare functions that satisfy several key axioms of social welfare, such as monotonicity, symmetry, and independence of scale [127, 128, 129, 130].

In practice, the right choice of p is often unclear in advance, and the decision-maker must understand how policies vary with p to make informed choices about which p (and thus which policy) to adopt. Small changes in p can sometimes lead to dramatically different policies, and selecting a policy optimized for an arbitrary p can lead to poor outcomes under a different p value. Despite these challenges, much of the existing work assumes a fixed social welfare function – and hence a fixed value of p – is given [131].

To address this challenge, we apply our portfolio framework that covers the entire spectrum of fairness criteria represented by $p \leq 1$. Our main algorithm, p-`MeanPortfolio`, sequentially selects finite p values starting from $-\infty$ to 1. These values are chosen so that the optimal policies at these points sufficiently cover the entire range of $p \leq 1$ for a given approximation factor α. We also propose a computationally efficient heuristic algorithm, which adaptively selects the next p value from the intervals formed by previously chosen p values.

6.1.1 Contributions

In this chapter, we explore the concept of α-approximate portfolios for preference aggregation in MORL. We summarize our contributions as follows:

1. We propose Algorithm p-`MeanPortfolio` (Algorithm 7) to compute a finite portfolio of policies that is α-approximate for the p-means objectives for any feasible set $\mathcal{D}$ and positive base objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$ value of $p \leq 1$. As with previous results in this thesis, we provide theoretical guarantees on the portfolio size. Additionally, we

also provide guarantees on the number of oracle calls our algorithm makes, i.e., the number of times we need to find the optimal $x \in \mathcal{D}$ for a fixed p-mean. This bound is crucial in the MORL setting, where each oracle call amounts to solving an MDP and can be expensive.

2. We also introduce a lightweight heuristic `BudgetConstrainedPortfolio` (Algorithm 9) that reduces number of oracle calls compared to p-`MeanPortfolio` while maintaining high-quality portfolio generation.

3. We evaluate our approach on three different domains in MORL, spanning synthetic to real-world problems. Our results show that a small portfolio can achieve near-optimal performance across all $p \leq 1$. Moreover, the heuristic `BudgetConstrainedPort-folio` constructs portfolios that closely match those produced by p-`MeanPort-Folio`, while significantly reducing the computational cost.

The literature in RL with multiple stakeholders has primarily focused on optimizing policies for a given choice of composite objective. Our portfolio approach captures the trade-offs within the actionable policy space, rather than relying on modeling choices on how these objectives should be formulated.

6.1.2 Example

An illustrative example we consider in this work is a public health application, where the staff of a healthcare organization aims to train an intervention policy for multiple beneficiaries under a limited budget [40]. This setting can be captured by Restless multi-armed bandits (RMABs) [132], which model sequential resource allocation problems under budget constraints. In this context, each beneficiary is associated with a state variable representing their level of engagement with the healthcare program, and the overall state of the problem is the concatenation of the states of all beneficiaries. At each time step, a policy determines which beneficiary to intervene on, subject to the budget constraint. Depending

on the reward function used to train a policy, different socio-demographic groups among the beneficiaries can be prioritized. For example, one reward function might encode the preference to prioritize older populations, while another reward function might encode the preference to prioritize low-income populations. The decision-maker (e.g., healthcare organization staff) must train a policy that balances these conflicting objectives. See Figure 1.2 for a demonstration of a portfolio generated using the proposed method. The portfolio consists of three policies, each affecting stakeholders differently based on age and education levels. This perspective helps staff understand the trade-offs between operational choices. For details on the experiment, refer to Section 6.4.

6.2 Preliminaries

In this section, we provide preliminaries on p-mean functions and MORL.

6.2.1 Setting

We are given a feasible set or domain $\mathcal{D}$ with positive base objective functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$. Throughout this chapter, the base functions can be *randomized*, so that each $h_i(x)$ is a random variable that depends on $i \in [d]$ and $x \in \mathcal{D}$. This additional flexibility will allow us to deal with the MORL setting described in the following subsection. Given an approximation factor $\alpha \in (0, 1)$, we seek an α-approximate portfolio $X \subseteq \mathcal{D}$ for p-mean objectives, i.e., for each $p \leq 1$, the set X must contain some x such that $\mathbb{E}[M_p(\mathbf{h}(x))] \geq \alpha \max_{x' \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x'))]$.

$\mathbf{h}(x) = (h_1(x), \ldots, h_d(x))$ denotes the d-dimensional (random) vector of base function values, and $M_p(z) = \left(\frac{1}{d}\sum_{i\in[d]} z_i^p\right)^{1/p}$ is the p-mean function (see Definition 2.3) for each $z \in \mathbb{R}^d_{>0}$. Recall that $M_1(z) = \frac{1}{d}\|z\|_1$ is the arithmetic mean, $M_{-\infty}(z) = \min_i z_i$ is the minimum, and $M_0(z) = \left(\prod_{i\in[d]} z_i\right)^{1/d}$ is the geometric mean.

We only assume *oracle access* to $\mathcal{D}$ and base functions $h_1, \ldots, h_d$ for the p-mean functions, i.e., for each $p \in [-\infty, 1]$, we assume an oracle allows us to query $x =$

$\arg\max_{x' \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]$. Next, we define the *condition number* of the base functions in terms of their bounds:

Assumption 6.1 (Bounded Function Values)**.** *There exist strictly positive scalars U, L such that $L \leq h_i(x) \leq U$ for all $i \in [d]$ and $x \in \mathcal{D}$ with probability 1. We call $\kappa := U/L$ the condition number of the base functions.*

We will use the following monotonicity property on p-mean functions:

Lemma 6.1 (Theorem 1, Chapter 3.3.1 [133])**.** *For all strictly positive vectors $z \in \mathbb{R}^d_{>0}$, and for all $p, q \leq 1$ with $p < q$, we have $M_p(z) \leq M_q(z)$.*

6.2.2 Multi-Objective Reinforcement Learning

We consider a multi-objective Markov Decision Process (MDP), defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \mathbf{R} = (R_i)_{i \in [d]})$, where $\mathcal{S}$ denotes a finite state space, $\mathcal{A}$ denotes a finite action space, and $P : \mathcal{S} \times \mathcal{A} \to \mathbf{\Delta}_{\mathcal{S}}$ is the transition probability. Additionally, we assume that we start in a fixed initial state $s_1 \in \mathcal{S}$ and H is the time horizon. In the specific MORL context we consider, there are d distinct stakeholders, each associated with a reward function $R_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_{>0}$ for $i \in [d]$. A policy $\pi : \mathcal{S} \times [H] \to \mathbf{\Delta}_{\mathcal{A}}$ defines a distribution over actions at a given time $h \in [H]$. We use $\tau = (s_1, a_1, \ldots, s_H)$ to denote a trajectory, which is the sequence of states and actions from time 1 to H. Throughout, we assume that all reward functions are bounded:

Assumption 6.2 (Bounded Reward)**.** *There exist strictly positive scalars U, L such that $L \leq R_i(\cdot) \leq U$ for all $i \in [d]$. We call $\kappa := U/L$ the condition number of rewards.*

The d stakeholders could represent different entities depending on the application; for example, different constituent demographics of a democracy, stakeholders in a company, or simply just abstract differences in values or preferences.

6.2.3 p-Means as Aggregation Functions

The generalized p-means provides a method to aggregate heterogeneous preferences by assigning a scalar value to any policy π. We focus on two aggregation rules, denoted by $\ell \in \{1, 2\}$, that have been previously proposed in the literature. Each combination of an aggregation rule $\ell \in \{1, 2\}$ and a parameter p defines a specific aggregation function, denoted as $v^{(\ell)}(\cdot, p)$. Given a trajectory $\tau = (s_1, a_1, \ldots, s_H, a_H)$, let $G_i(\tau) = \frac{1}{H}\sum_{h=1}^{H} R_i(s_h, a_h), i \in [d]$, represent the average reward over the trajectory, and $\mathbf{G}(\tau) \in \mathbb{R}^d$ the vector with $G_i(\tau)$ as its ith entry. The aggregation functions are defined as follows:

$$v^{(1)}(\pi, p) = \mathbb{E}_{\tau\sim\pi}\left[M_p\big(\mathbf{G}(\tau)\big)\right], \tag{6.1}$$

$$v^{(2)}(\pi, p) = M_p\big(\mathbb{E}_{\tau\sim\pi}[\mathbf{G}(\tau)]\big). \tag{6.2}$$

In the MORL literature, $v^{(1)}(\cdot, p)$ is referred to as *expected scalarized returns* (ESR), while $v^{(2)}(\cdot, p)$ is known as *scalarized expected returns* (SER). In general, ESR is preferred when the expected total reward from a single execution is the primary focus, whereas SER is more suitable when policies are executed multiple times and rewards accumulate over iterations. Computing the optimal policy for $v^{(\ell)}(\cdot, p)$ requires specialized algorithms different from standard RL methods, which are not directly applicable. For example, [41] and [134] developed algorithms for $v^{(1)}(\cdot, p)$ and $v^{(2)}(\cdot, p)$, respectively. More detailed comparisons between these scalarization techniques and their solution algorithms are available in [134], [135], [136], [134], and [41].

While these aggregation rules are often studied independently in MORL, our work provides a unified framework applicable to both methods. Both $v^{(1)}, v^{(2)}$ are special cases of our general setting, with the feasible set $\mathcal{D} = \Pi$. For $v^{(1)}$, the base function $h_i(\pi) = G_i(\tau)$, where the trajectory τ is sampled from policy $\pi \in \Pi$. That is, the ith base function for a policy π quantifies the average reward for stakeholder $i \in [d]$ on the random trajectory

$\tau \sim \pi$. Then $\mathbb{E}[M_p(\mathbf{h}(\pi))] = v^{(1)}(\pi, p)$. Further, if $L \leq R_i(\cdot) \leq U$ for all i, then $L \leq G_i(\tau) \leq R$ for all i, τ, and therefore, if the rewards have condition number $\frac{U}{L} \leq \kappa$, then the base functions also have condition number $\leq \kappa$. For $v^{(2)}$, the base function $h_i(\pi) = \mathbb{E}_{\tau\sim\pi}[G_i(\tau)]$, i.e., the expected reward on trajectories sampled from policy π. Note that the base functions are deterministic in this case. As before, if the rewards have condition number $\frac{U}{L} \leq \kappa$, then the base functions also have condition number $\leq \kappa$.

Oracle Access. For a fixed p, aggregation function $v^{(\ell)}(\cdot, p), \ell \in \{1, 2\}$, and a given set of policies Π, the associated welfare maximizing policy can be obtained by solving the following problem:

$$\arg\max_{\pi\in\Pi} v^{(\ell)}(\pi, p). \tag{6.3}$$

If Π is defined as the set of all possible policies in the MDP, as in the standard RL setting, each oracle call can be computationally expensive. Moreover, achieving exact optimality for (Equation 6.3) may not always be feasible due to limitations in RL algorithms. This means that when measuring the approximation factor of a policy π with respect to an aggregation function $v^{(\ell)}(\cdot, p)$, the maximum value $v_*^{(\ell)}(p)$ might not be attainable or computable. In this case, we approximate $v_*^{(\ell)}(p)$ as the value achieved by applying the given RL algorithm. This approach measures the approximation factor relative to what is achievable using the algorithm at hand.

In practice, Π may be a small set of pre-trained policies, particularly in scenarios where training new policies is infeasible. In such cases, (Equation 6.3) can be solved efficiently by enumerating over the policies $\pi \in \Pi$ and using Monte Carlo simulations to estimate the value $v^{(\ell)}(\pi, p)$.

The notion of oracle access $\max_{x\in\mathcal{D}} M_p(\mathbf{h}(x))$ for base functions reduces to Equation 6.3 in the case of MORL.

6.2.4 Portfolios for Aggregation Rules

We state the formal definition of portfolios for these aggregation functions. This is consistent with Definition 1.2. For brevity, we denote the maximum value function $v_*^{(\ell)} : [-\infty, 1] \to \mathbb{R}$ defined as $v_*^{(\ell)}(p) := \max_{\pi \in \Pi} v^{(\ell)}(\pi, p)$. When ℓ and Π are clear from context, we denote the optimal policy for $v^{(\ell)}(\cdot, p)$ as π_p.

Definition 6.1 (MORL Portfolio)**.** *Given an MORL setting with aggregation rule $\ell \in \{1, 2\}$ and approximation factor $\alpha \in (0, 1)$, a set of policies $\Pi' \subseteq \Pi$ is called an α-approximate portfolio for aggregation functions $v^{(\ell)}$ if for each $p \leq 1$, there exists a policy $\pi' \in \Pi'$ that is an α-approximation to $v^{(\ell)}(\cdot, p)$, that is,*

$$v^{(\ell)}(\pi', p) \geq \alpha \max_{\pi \in \Pi} v^{(\ell)}(\pi, p).$$

6.3 Portfolio Algorithms

In this section, we present our main algorithmic and theoretical results.

6.3.1 Algorithm p-`MeanPortfolio`

Algorithm 7 p–`MeanPortfolio`

input: (i) domain $\mathcal{D}$, (ii) base objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$ (iii) an oracle that given a $p \in [-\infty, 1]$, returns the optimum p-mean vector $\arg\max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]$, and (iv) desired approximation factor $\alpha \in (0, 1)$

output: α-approximate portfolio $X \subseteq \mathcal{D}$

initialize $p_0 = -\frac{\ln d}{\ln(1/\alpha)}$ and $t = 0$
while $p_t < 1$ **do**
    add $x^{(t)} := \arg\max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]$ to X
    $p_{t+1} = \texttt{LineSearch}(p_t, \alpha)$
    $t \leftarrow t + 1$
return $X = \{x^{(0)}, \ldots, x^{(t-1)}\}$

We introduce p-`MeanPortfolio` (Algorithm 7), which constructs an α-approximate portfolio $X \subseteq \mathcal{D}$ for all $p \leq 1$. In Theorem 6.1, we establish formal bounds on the portfolio

Algorithm 8 `LineSearch`

input: (i) some $p \in (-\infty, 1)$, and (ii) desired approximation factor $\alpha \in (0, 1)$
output: $b > p$ such that $x_p := \arg\max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]$ is an α-approximation for $\mathbb{E}[M_q(\mathbf{h}(\cdot))]$ for all $q \in [p, b]$

$a \leftarrow p$ and $b \leftarrow 1$
initialize $x = \arg\max_{x' \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x'))]$
while $\mathbb{E}[M_a(\mathbf{h}(x))] < \alpha \ \max_{x' \in \mathcal{D}} \mathbb{E}[M_b(\mathbf{h}(x'))]$ **do**
    $q \leftarrow \frac{a+b}{2}$
    if $\mathbb{E}[M_a(\mathbf{h}(x))] \geq \sqrt{\alpha} \ \max_{x' \in \mathcal{D}} \mathbb{E}[M_q(\mathbf{h}(x'))]$ **then**
        $a \leftarrow q$
    else
        $b \leftarrow q$
return b

size $|X|$ and the number of oracle calls to Equation 6.3. The full proof is provided in Appendix C.

Theorem 6.1. *Given feasible set $\mathcal{D}$, (possibly random) base objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$ with condition number κ, and desired approximation factor $\alpha \in (0, 1)$, Algorithm 7 (p-*`MeanPortfolio`*) returns an α-approximate portfolio of policies X for the class $\mathbf{C}$ of all p-mean functions for $p \leq 1$. Further,*

1. *The portfolio size*

$$|X| = O\left(\frac{\ln \kappa}{\ln(1/\alpha)}\right),$$

2. *The number of oracle calls by the algorithm is upper-bounded by*

$$\widetilde{O}\left(\frac{(\ln \kappa)^2 \ln \ln d}{\ln(1/\alpha)}\right),$$

where $\widetilde{O}$ hides all lower order terms.

As an immediate corollary, we have:

Corollary 6.1. *Given an MDP $\mathcal{M}$ with d reward functions with condition number κ, set Π of feasible policies, an aggregation rule $\ell \in \{1, 2\}$, and a desired approximation factor $\alpha \in$*

$(0, 1)$*, Algorithm 7 (*p-`MeanPortfolio`*) returns an* α*-approximate portfolio of policies* Π' *for the set of aggregation functions* $\{v^{(\ell)}(\cdot, p) : p \leq 1\}$*. Further,*

1. *The portfolio size*

$$|\Pi'| = O\left(\frac{\ln \kappa}{\ln(1/\alpha)}\right),$$

2. *The number of oracle calls by the algorithm is upper-bounded by*

$$\widetilde{O}\left(\frac{(\ln \kappa)^2 \ln \ln d}{\ln(1/\alpha)}\right),$$

where $\widetilde{O}$ *hides all lower order terms.*

We note that the logarithmic dependence of the portfolio size on κ is necessary; in particular, there exist instance families with condition number κ where $O(1)$-approximate portfolios must have size $\Omega(\ln \kappa)$. Note the contrast with minimization portfolios for L_p norms (Chapter 3), where the portfolio size is independent of κ.

Here, we explain the high-level ideas behind the algorithm. The algorithm iteratively chooses an increasing sequence of p values $p_0 < p_1 < \ldots < p_K = 1$ using a line search subroutine (Algorithm 8). It ensures that $x^{(t)}$ is α-approximate for all $p \in [p_t, p_{t+1}]$ for all $t \in [K-1]$, and that $x^{(0)}$ is also α-approximate for all $p \in [-\infty, p_0]$.

Line search. The line search subroutine works as follows: given a $p \leq 1$, it seeks to find some $b^* \geq p$ such that x_p is an α-approximation for all $q \in [p, b^*]$. To achieve this, it maintains lower and upper bounds a, b on b^* with $p \leq a < b \leq 1$ and iteratively refines these bounds.

At each iteration, the algorithm checks whether $\mathbb{E}[M_a(x_p)] \geq \alpha \max_{x \in \mathcal{D}} \mathbb{E}[M_b(x)]$ (line 3). If this condition holds, then by the monotonicity property of p-means (Lemma 6.1), we must have:

$$\mathbb{E}[M_b(\mathbf{h}(x_p))] \geq \mathbb{E}[M_a(\mathbf{h}(x_p))] \geq \alpha \max_{x \in \mathcal{D}} \mathbb{E}[M_b(\mathbf{h}(x))].$$

Then, this holds for any $q \in [a, b]$, implying that π_p is α-approximate across this interval.

Therefore, this procedure can safely output $b^* = b$. This establishes the correctness of the algorithm, as the line search successfully identifies the next p value if it terminates.

Bounds a, b are updated in each iteration as follows (lines 4-8): we query the value $\arg\max_{x \in \mathcal{D}} \mathbb{E}[M_q(x)]$ for $q = \frac{a+b}{2}$. As before, if $\mathbb{E}[M_a(\mathbf{h}(x_p))]$ exceeds α times the maximum $\max_{x \in \mathcal{D}} \mathbb{E}[M_q(\mathbf{h}(x))]$, then we know that x_p must be an α-approximation for any value in $[a, q]$, and the lower bound a can be tightened as $a \leftarrow q$. Otherwise, the upper bound can be tightened as $b \leftarrow q$.

However, as we show in the formal proof in Appendix C, we can in fact ensure faster convergence by slightly modifying this step. Instead of checking whether $\mathbb{E}[M_a(\mathbf{h}(x_p))] \geq \alpha \times \max_{x \in \mathcal{D}} \mathbb{E}[M_q(\mathbf{h}(x))]$, we check the stronger condition $\mathbb{E}[M_a(\mathbf{h}(x_p))] \geq \sqrt{\alpha} \times \max_{x \in \mathcal{D}} \mathbb{E}[M_q(\mathbf{h}(x))]$. Since $\sqrt{\alpha} \geq \alpha$, this stricter condition does not affect correctness but accelerates convergence.

Oracle complexity. We briefly sketch how we bound the oracle complexity (i.e., number of oracle calls (Equation 6.3)) by bounding the number of oracle calls in each run of `LineSearch`. As discussed, in each iteration of the line search algorithm, the distance $b - a$ is cut by half since either b or a are updated to $\frac{a+b}{2}$. As we show in Lemma C.7, this implies that after j iterations of `LineSearch`, the ratio $\frac{\max_{x \in \mathcal{D}} \mathbb{E}[M_b(\mathbf{h}(x))]}{\max_{x \in \mathcal{D}} \mathbb{E}[M_a(\mathbf{h}(x))]}$ is upper bounded by $\psi(\kappa, d, \alpha) \times 2^{-j}$, where ψ is some function of condition number κ, dimension d, and approximation factor α. By deriving a lower bound on this ratio, we derive an upper bound on the total number of iterations j.

6.3.2 Heuristic under a Budget Constraint

In p-`MeanPortfolio`, we provide guarantees on the approximation factor α. However, achieving these guarantees may require a large number of oracle calls to solve Equation 6.3 across different values of p, which can be impractical when computational resources are severely limited.

One way to reduce oracle calls is to select a smaller α. However, while Corollary 6.1

provides an upper bound on the number of calls required, the exact number is difficult to determine in advance, making it challenging to balance computational cost with portfolio quality. Alternatively, a budget on the total number of oracle calls can be imposed along α, but this introduces its own trade-offs: if α is too large, the algorithm may exhaust calls prematurely, leaving parts of the p range unexplored; if α is too small, it may progress to $p = 1$ too quickly, missing opportunities to refine the portfolio.

To address this limitation, we propose the heuristic `BudgetConstrainedPort-folio` (Algorithm 9), designed for scenarios with a strict budget K on oracle calls. Unlike p-`MeanPortfolio`, which selects p values in a monotonically increasing order from $-\infty$ to 1, this algorithm dynamically refines the search space based on observed approximation performance. It greedily targets regions where the approximation factor is likely to be weakest, ensuring efficient use of the limited oracle budget.

First, we describe the ideal version of this greedy approach. After t oracle calls, let X_t denote our current portfolio. The objective at each step is to identify the worst-case p value where the approximation quality of X_t is the lowest by solving the following problem:

$$\mathcal{Q}(X_t) = \min_{p \leq 1} \frac{\max_{x \in X_t} \mathbb{E}[M_p(\mathbf{h}(x))]}{\max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]}. \tag{6.4}$$

However, solving this optimization problem exactly is computationally expensive (or potentially even infeasible), particularly since evaluating the objective for any new p would require an additional oracle call. Instead, `BudgetConstrainedPortfolio` approximates this worst-case p using only the information gathered from previous iterations, inspired by the theoretical principles of p-`MeanPortfolio`.

After iteration t, the previously selected p values partition the interval $[\infty, 1]$ into at most $t + 1$ disjoint intervals:

$$-\infty < p_{m(1)} < p_{m(2)} < \cdots < p_{m(t)} \leq 1.$$

Rather than solving Equation 6.4 exactly to pinpoint the worst-case p, we instead aim to identify the interval where the approximation quality is the weakest.

First, we rewrite Equation 6.4 by decomposing the minimization over the intervals:

$$\mathcal{Q}(X_t) \approx \min_{l\in[t-1]} \left[\min_{p\in[p_{m(l)},p_{m(l+1)}]} \frac{\max_{x\in X_t} \mathbb{E}[M_p(\mathbf{h}(x))]}{\max_{x\in\mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]} \right].$$

The only approximation introduced in this decomposition arises from neglecting the intervals $[-\infty, p_{m(1)}]$ and $[p_{m(t)}, 1]$. To cover the first interval, we can initialize the algorithm with a sufficiently small p_1, guaranteeing that the approximation factor remains well-controlled in this region (Corollary 6.1). The second interval is covered by explicitly setting $p_2 = 1$.

For each interval $[p_{m(l)}, p_{m(l+1)}]$, we compute its interval approximation factor, denoted as $u(l)$, using the following sequence of approximations:

$$\begin{aligned}
&\min_{p\in[p_{m(l)},p_{m(l+1)}]} \frac{\max_{x\in X_t} \mathbb{E}[M_p(\mathbf{h}(x))]}{\max_{x\in\mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]} \\
&\approx \min_{p\in[p_{m(l)},p_{m(l+1)}]} \frac{\mathbb{E}[M_p(\mathbf{h}(x_{p(m(l))}))]}{\max_{x\in\mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]} \\
&\approx \frac{\mathbb{E}[M_{p_{m(l+1)}}(\mathbf{h}(x_{p(m(l))}))]}{\max_{x\in\mathcal{D}} \mathbb{E}[M_{p(m(l+1))}(\mathbf{h}(x))]} := u(l).
\end{aligned}$$

The first approximation follows from the assumption that each interval is well-covered by the policy trained at its left endpoint, similar to the approach used in p-`MeanPortfolio`. The second approximation is justified under the assumption that $\frac{\mathbb{E}[M_p(\mathbf{h}(x_{p(m(l))}))]}{\max_{x\in\mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]}$ is a monotonically decreasing function of p in the interval $[p_{m(l)}, p_{m(l+1)}]$. Note that this assumption holds exactly when the interval is sufficiently small, as this function is continuous in p (Lemma C.7) and attains its maximum value ($= 1$) at $p = p_{m(l)}$.

To select the next p value, we identify the interval with the worst (smallest) approxima-

Algorithm 9 `BudgetConstrainedPortfolio`

input: (i) domain $\mathcal{D}$, (ii) base objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{>0}$ (iii) an oracle that given a $p \in [-\infty, 1]$, returns the optimum p-mean vector $\arg\max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]$, and (iv) desired approximation factor $\alpha \in (0, 1)$ (v) initial p_0

output: a portfolio X

initialize $X \leftarrow \emptyset$
for $t = 1$ **to** K **do**
  if $t = 1$ **then**
    $p_t \leftarrow p_0$
  else if $t = 2$ **then**
    $p_t \leftarrow 1$
  else
    compute $u(l)$ for $[p_{m(l)}, p_{m(l+1)}], \forall l \in [t-2]$
    $p_t \leftarrow \frac{p_{m(l^*)} + p_{m(l^*+1)}}{2}$, $l^* = \arg\min_{l \in [t-2]} u(l)$
  Add policy $x_{p_t} := \arg\max_{x \in \mathcal{D}} \mathbb{E}[M_{p_t}(\mathbf{h}(x)]$ to X
return X

tion factor and choose its midpoint:

$$p_{t+1} = \frac{p_{m(l^*)} + p_{m(l^*+1)}}{2}, \quad l^* = \arg\min_{l \in [t-1]} u(l).$$

6.4 Numerical Experiments

In this section, we present the results of our numerical experiments. Additional details of the experimental setups and the environments are provided in Section D.3.[1]

6.4.1 Experimental Setups

For a given α and MDP environment, we first compute a portfolio using p-`MeanPort-folio` and compare it against two baseline approaches. Suppose p-`MeanPortfolio` generates a portfolio of size K. The first baseline selects K values of p uniformly at random from $[p_0, 1]$ and computes their optimal policies, where p_0 matches that of p-`MeanPortfolio`. The second baseline randomly samples K policies from Π. Since both

[1]The code for our experiments can be found at https://github.com/jaimoondra/approximation-portfolios-for-rl/.

baselines involve randomness, we generate 10 independent baseline portfolios for each setting and report the average performance across these runs. Finally, we run `BudgetConstr-ainedPortFolio` with budget K.

To evaluate each portfolio $\Pi' = X$, we compute its approximation factor $\mathcal{Q}(\Pi')$ as defined in Equation 6.4. Since computing this value exactly is infeasible, we approximate it using a grid search over p, referring to it as the actual approximation. We also compare the number of oracle calls made by p-`MeanPortfolio` and `BudgetConstrainedPort-folio`. This procedure is repeated for varying α values.

6.4.2 Environments

We conduct experiments across three domains, ranging from synthetic to real-world settings, as described below.

Taxi Environment. We consider a synthetic setting based on the works of [137, 41]. The taxi environment consists of a grid world where a taxi driver serves passengers across $d = 4$ different source-destination pairs. When the agent drops off a passenger at their destination, it earns a reward corresponding to that route. However, since the taxi can only carry one passenger at a time, it must decide which route to prioritize. A fair agent should serve all source-destination pairs equitably, avoiding the neglect of more challenging routes. We train a p-mean maximizing policy using Welfare Q-Learning [41]. Finally, we experiment on $v^{(1)}$ and define Π as the set of all possible policies in the MDP.

Resource Allocation after Natural Disaster. In this synthetic setting, we have a set of $d = 12$ clusters of neighborhoods impacted by a natural disaster. Each cluster is characterized by average household income (high, middle, low), proximity to critical infrastructure (near, far), and population density (high, low), along with distinct post-disaster resource needs. Over a time horizon, a decision-maker must decide how to allocate a limited number of resources to these 12 clusters. The reward function for each cluster is the average of the fraction of unmet need and the fraction of total aid allocated to that cluster. We conduct

experiments using $v^{(2)}$, with Π defined as a finite set of pre-trained policies.

Healthcare Intervention.[2] We consider a real-world healthcare intervention problem, modeled as an RMAB problem described in Subsection 6.1.2 [21]. ARMMAN [22], an NGO based in India, runs large-scale maternal and child mobile health programs. One of their initiatives delivers critical health information via weekly automated voice messages. To enhance engagement, a limited number of beneficiaries receive direct calls from health workers each week, with call assignments determined by a policy. The problem involves $d = 59$ reward functions, each prioritizing different socio-demographic groups among the beneficiaries. We conduct experiments using $v^{(2)}$, where Π consists of a finite set of pre-trained policies. All experiments are strictly secondary analyses and adhere to ethics board approvals; for further discussion, refer to our impact statement.

6.4.3 Results

Table 6.1 presents comparisons of the approximation quality of p-`MeanPortfolio` with random p sampling and random policy sampling baselines, while Table 6.2 reports the number of oracle calls made by p-`MeanPortfolio` and `BudgetConstrained-Portfolio`. Note that by design, the number of calls for `BudgetConstrainedPort-folio` is always the same as the portfolio size.

Size. The portfolios computed using Algorithm p-`MeanPortfolio` remain small while achieving a very high approximation factor. Across all three environments, the portfolio size never exceeds 10, yet still attains an approximation factor close to 1. This suggests that a small portfolio is sufficient to effectively cover the entire spectrum of p values.

Approximation Quality. The proposed algorithms achieve significantly better approximation quality than both benchmark methods. In particular, p-`MeanPortfolio` generally attains the highest approximation quality, followed closely by algorithm `Budget-ConstrainedPortfolio` in most cases.

[2]The ARMMAN data used to train the reward models was handled and processed at Harvard. Only the model-generated reward functions and corresponding reward vectors were used in this thesis.

However, `BudgetConstrainedPortfolio` and random p sampling occasionally outperform p-`MeanPortfolio`. This is because p-`MeanPortfolio` selects policies solely to meet the input approximation factor α, but has no incentive or mechanism to surpass this approximation quality.

In contrast, `BudgetConstrainedPortfolio` fully utilizes the input budget K, and strategically selects boundary points first (a small initial p followed by $p = 1$). When K is very small (1 or 2), this initial heuristic strategy can provide more effective coverage across $p \leq 1$, explaining its occasional superior performance. Likewise, in rare cases where $K = 1$, a randomly chosen p may happen to yield better coverage than the single policy selected by p-`MeanPortfolio`.

Computational Efficiency. As noted earlier, p-`MeanPortfolio` requires a large number of oracle calls to produce a high-quality portfolio. In contrast, `BudgetConstr-ainedPortfolio` achieves comparable quality while using significantly fewer oracle calls. This suggests that when computational resources are limited, `BudgetConstr-ainedPortfolio` serves as a strong alternative with good empirical performance.

Diversity. As we have shown in Figure 1.2 in Subsection 6.1.2, portfolios can simplify policy deployment decisions by presenting a small yet diverse set of policies. We also provide the entire p values chosen by p-`MeanPortfolio` in Section D.2, Table D.2. We observe that the p values are quite diverse, which corroborates the algorithm's objective to cover the entire p range only with these selected values. Figure D.1 and Figure D.2 further illustrate how the outcomes of the optimal policies for different p values in the portfolio lead to varying impacts for the stakeholders. Importantly, these are not arbitrary policies but optimal policies for specific p values, with approximation guarantees extending to other p values as well.

[3]Here, we observe a drop in the actual approximation factor of p-`MeanPortfolio` when the portfolio size is 8. Although this portfolio is computed using $\alpha = 0.9$, its achieved approximation is slightly lower. This discrepancy is likely due to the inherent randomness in the RL algorithm and the challenges of computing exact optimal policies in RL settings, as discussed in Section 6.3.

Table 6.1: A comparison of actual approximation ratios across various portfolio sizes for p-MeanPortfolio, random policy sampling, and random p sampling. p-MeanPortfolio consistently outperforms the other two methods across all portfolio sizes and experiments.

Portfolio Size	p-Mean-Portfolio	Random Policy Sampling	Random p Sampling
Resource Allocation after Natural Disaster			
1	0.706	0.534	**0.832**
2	**0.904**	0.568	0.890
3	**0.999**	0.591	0.892
4	**1.000**	0.609	0.888
Healthcare Intervention			
1	**0.924**	0.479	0.913
2	**0.982**	0.545	0.941
3	**0.982**	0.589	0.929
4	**0.982**	0.625	0.947
5	**0.993**	0.635	0.957
6	**0.999**	0.647	0.962
7	**1.000**	0.667	0.953
Taxi Environment			
1	**0.66**	0.24	**0.66**
2	**0.61**	0.31	**0.61**
3	**0.97**	0.54	0.65
8	**0.87**	0.71	0.67
10	**0.99**	0.60	0.65

6.5 Conclusion

In this paper, we studied the concept of an α-approximate portfolio in MORL for generalized p-means. We proposed an algorithm to compute α-approximate portfolios and established theoretical guarantees on the trade-offs among α, portfolio size, and the number of oracle calls. Additionally, we developed an efficient heuristic algorithm. Our numerical experiments demonstrated that the proposed methods successfully compute compact portfolios that effectively capture the entire spectrum of p values, empowering decision-makers to make informed decisions.

Table 6.2: A comparison of the number of oracle calls and actual approximation ratios for p-`MeanPortfolio` and `BudgetConstrainedPortfolio` across different portfolio sizes. While p-`MeanPortfolio` often achieves slightly better approximation, it requires significantly more oracle calls.[3]

Portfolio Size	p-`MeanPortfolio`		`BudgetConstrained-Portfolio`	
	Oracle Calls	Actual Approximation	Oracle Calls	Actual Approximation
Resource Allocation after Natural Disaster				
1	1	0.706	**1**	**0.885**
2	7	0.904	**2**	**0.921**
3	18	**0.999**	**3**	0.921
4	50	**1.000**	**4**	0.921
Healthcare Intervention				
1	2	0.924	**1**	**0.938**
2	7	**0.982**	**2**	0.938
3	11	**0.982**	**3**	0.938
4	19	**0.982**	**4**	0.938
5	23	**0.993**	**5**	0.986
6	46	**0.999**	**6**	0.986
7	61	**1.000**	**7**	0.993
Taxi Environment				
1	17	**0.66**	**1**	**0.66**
2	30	**0.61**	**2**	**0.61**
3	44	**0.97**	**3**	0.92
8	118	0.87	**8**	**0.90**
10	144	**0.99**	**10**	0.92

CHAPTER 7
PORTFOLIO OF ALGORITHMS: ONLINE MIRROR DESCENT WITH MULTIPLE MIRROR MAPS

7.1 Introduction

In this chapter, we study portfolios of algorithms for online learning. Specifically, we consider the OCO [138, 139, 140] setup: given a convex body $\mathcal{K} \in \mathbb{R}^d$, at each time $t \in [T]$, a player must *play* a point $x^{(t)} \in \mathcal{K}$. An adversary reveals the convex loss function $f^{(t)} : \mathcal{K} \to \mathbb{R}$ once $x^{(t)}$ is played, incurring *loss* $f^{(t)}(x^{(t)})$ for the player. The goal of the player is to minimize *regret*, which benchmarks the performance of the online algorithm with a clairvoyant oracle that is constrained to make a fixed decision across all time steps. That is, define the total loss of the player as $\sum_{t\in[T]} f^{(t)}(x^{(t)})$; then the *regret* of the player is defined as the difference

$$\texttt{regret}(T) = \sum_{t\in[T]} f^{(t)}(x^{(t)}) - \min_{x\in\mathcal{K}} \sum_{t\in[T]} f^{(t)}(x). \tag{7.1}$$

Surprisingly, under mild conditions on losses $f^{(t)}$, it is possible to incur $o(T)$ regret even in this very general setting. In Online Projected Gradient Descent (OPGD), the player plays an arbitrary point $x^{(1)} \in \mathcal{K}$ at $t = 1$. When loss $f^{(t)}(x^{(t)})$ is incurred, the player moves to $y^{(t+1)} = x^{(t)} - \eta\nabla f^{(t)}(x^{(t)})$, where η is a parameter we fix later. Since $y^{(t+1)}$ may not be in $\mathcal{K}$, $x^{(t+1)}$ is obtained by *projecting* $y^{(t+1)}$ on $\mathcal{K}$ under Euclidean distance, i.e., $x^{(t+1)} = \arg\min_{x\in\mathcal{K}} \frac{1}{2}\|x - y^{(t+1)}\|_2^2$.

For a bounded convex body $\mathcal{K}$ with Euclidean diameter $D = \max_{x,z\in\mathcal{K}} \|x - z\|_2$, and for convex loss functions $f^{(1)}, \ldots, f^{(T)}$, OPGD incurs regret at most $O(DG\sqrt{T})$, where $G = \max_t \max_{x\in\mathcal{K}} \|\nabla f^{(t)}(x^{(t)})\|_2$ is an upper bound on the gradients of loss functions.

This chapter is based on joint work with Swati Gupta and Mohit Singh.

This bound is asymptotically optimal, unless more structural assumptions are made.

Another standard algorithm is the Online Exponentiated Gradient (OEG), which applies when the convex body $\mathcal{K}$ lies in the probability simplex $\mathbf{\Delta}_d := \{x \in \mathbb{R}^d_{\geq 0} : \sum_{i\in[d]} x_i = 1\}$. As before, the player plays an arbitrary point $x^{(1)} \in \mathcal{K}$ at $t = 1$ in OEG. When loss $f^{(t)}(x^{(t)})$ is incurred, the player moves to $y^{(t+1)} = x^{(t)} \exp(-\eta \nabla f^{(t)}(x^{(t)}))$, where η is a parameter we fix later. Since $y^{(t+1)}$ may not be in $\mathcal{K}$, $x^{(t+1)}$ is obtained by projecting $y^{(t+1)}$ on $\mathcal{K}$ under *KL divergence*: $x^{(t+1)} = \arg\min_{x\in\mathcal{K}} \sum_{i\in[d]} x_i \ln \frac{x_i}{y_i^{(t+1)}}$. In particular, when $\mathcal{K} = \mathbf{\Delta}_d$ is the probability simplex, then $x^{(t+1)} = \frac{y^{(t+1)}}{\|y^{(t+1)}\|_1}$. In this case, OEG typically yields better regret than OPGD.

As one might expect, for specific convex bodies $\mathcal{K}$ and loss functions, it is possible to achieve better regret bounds by choosing an algorithm more tailored to their geometry. Online Mirror Descent (OMD) (Algorithm 11) is a natural generalization of OPGD and OEG, where the projection step can use any *mirror map* or *Bregman divergence* (defined shortly); OPGD corresponds to the case when this mirror map is the squared L_2 norm $\frac{1}{2}\|x\|_2^2$, and OEG corresponds to the case when this mirror map is the negative entropy $\sum_{i\in[d]} x_i \ln x_i$. Given this choice of the mirror map in OMD, it is natural to ask if it can be adjusted to exploit the structure of the loss functions and the convex body. In this work, we ask the following questions:

Can we obtain a portfolio of diverse mirror maps for OMD that can tune to the problem geometry? Can the optimum mirror map among this portfolio be learnt during runtime?

7.1.1 Contributions

Portfolio of block norms. First, we propose mirror maps corresponding to *block norms* [141] (see Subsection 7.2.2) of different sizes as our portfolio. A block norm of a vector is obtained by partitioning the coordinates into various *blocks* and adding the L_2 norms of each block. Different block norms interact with the geometry of the polytope and the loss functions differently, making a portfolio of block norms useful when the problem geometry

is unknown in advance. Specifically, when the convex body $\mathcal{K}$ lies in dimension d, for each size $n \in \{2^0, 2^1, \ldots, 2^{\log_2 d} = d\}$, we consider the block norm obtained by partitioning the d coordinates into n blocks uniformly at random. The corresponding $1 + \log_2 d$ mirror maps form our portfolio of algorithms.

The case of $n = 1$ block recovers the OPGD algorithm, and the case of $n = d$ blocks roughly corresponds to OEG. Thus, this portfolio allows for interpolation between these two standard algorithms [141].

Regret improvements. Next, we show that there exist OCO settings where using an intermediate block norm[1], i.e., $n \notin \{1, d\}$ leads to asymptotically improved regret rates as compared to either of the standard algorithms: OPGD and OEG. Specifically, we show that

- (Theorem 7.2) When the gradients of loss functions are *sparse*, we characterize the regret using block norms in terms of the diameter of the convex body $\mathcal{K}$ as a function of the number n of blocks.
- (Theorem 7.3) There exists a family of sparse loss functions over the probability simplex $\boldsymbol{\Delta}_d := \{x \in \mathbb{R}^d : x \geq 0, \sum_{i \in [d]} x_i = 1\}$ in dimension d where using an appropriate block norm improves the regret by factor $O\left(\frac{\sqrt{\ln d}}{\ln \ln d}\right)$ over either OPGD or OEG. We verify this through numerical simulations.
- (Theorem 7.4) There exists a family of polytopes in dimension d and sparse loss functions where using an appropriate block norm improves the regret by a polynomial in d factor over either $n = 1$ (OPGD) or $n = d$ blocks.

Automatically choosing a mirror map. Finally, we discuss strategies to learn the optimum mirror map from the portfolio of block norm mirror maps. Even though each mirror

[1]We use 'using the nth block norm' as a proxy for 'using the mirror map corresponding to the nth Block norm.'

map enjoys a convergence (via a decrease in a tailored potential function that combines the distance from the optimum and the drop in the function value), mixing two mirror maps arbitrarily does not even lead to sublinear regret. In particular, we construct examples of instances and iteration-specific sparse losses where alternating between OPGD and OEG can lead to $\Omega(T)$ regret (Theorem 7.5).

This motivates other strategies for mixing mirror maps. Next, we extend the previous work on learning hyperparameters of algorithms [45, 46, 47]. Specifically, we give a Multiplicative Weights update algorithm (Algorithm 12) that automatically chooses the appropriate mirror map and step size from among a portfolio of mirror maps. We apply this specifically to block norm mirror maps, and show that this algorithm achieves regret within factor $O(\sqrt{\ln \ln d})$ of the optimal regret of any block norm, under mild technical assumptions (Theorem 7.7).

7.1.2 Related Work

The study of OCO and its canonical algorithms – Online Projected Gradient Descent (OPGD) and Online Mirror Descent (OMD) – dates back to [138] and the early developments in convex online learning (see [139, 140]). For most standard convex sets, these two mirror maps – Euclidean ($h(x) = \frac{1}{2}\|x\|_2^2$) and entropic ($h(x) = \sum_i x_i \log x_i$, or an equivalent proxy) – have long been viewed as the two 'universally optimal' geometries: Euclidean for the hypercube and entropic for the probability simplex. Both achieve minimax-optimal regret of order $O(f(d)\sqrt{T})$, with dimension dependence $f(d)$ varying by the setting.

It is known that other mirror maps can achieve better asymptotically better regret rates in certain OCO settings. For example, [43] show that using the appropriate L_p norm mirror map achieves regret $O(\sqrt{(\ln S)T})$ for sparse loss functions over the probability simplex, improving over both the OPGD bound $O(\sqrt{ST})$ and OEG bound $O\sqrt{(\ln d)T}$ when $S = \ln d$. However, to the best of our knowledge, we are the first to show polynomial in dimension improvement in regret over both OPGD and OEG (or an equivalent proxy). Fur-

ther, unlike [43], whose improvements hold in disjoint sparse/dense regimes, we construct a single online sequence where both Euclidean (OPGD) and entropic (OEG) mirror maps are simultaneously suboptimal. We construct a natural convex body, $P = \mathrm{conv}(\mathbf{\Delta}_d, A\mathbf{1})$, and an oblivious linear-loss sequence for which OPGD (Euclidean) and the entropy proxy (mirror map for dth block norm) both suffer regret scaling as $\widetilde{\Omega}(d^{1/6}\sqrt{T})$, while OMD with a block-norm mirror map tailored to the sparsity structure achieves $\widetilde{O}(\sqrt{T})$.

Individual block norms were first considered in (offline) Mirror Descent in Ben-Tal and Nemirovsky's optimization framework [141] as alternatives to standard algorithms such as PGD. Block norms have been considered as alternatives to OMD for OCO [142]; however, no polynomial asymptotic improvements in regrets using block norms have been shown, to the best of our knowledge.

Other algorithms such as AdaGrad [143] achieve better regret rates in regimes different from ours, e.g., the upper bound for AdaGrad does not yield regret improvements for the probability simplex OCO setting described in Subsection 7.3.2 or the other OCO setting in Subsection 7.3.3.

The problem of finding the optimal mirror map for a given OCO setting has also been considered: [144] give an instance-dependent construction that obtains the optimal mirror map as the solution to an optimization problem. Naturally, it requires *a priori* knowledge of the loss functions. Further, it is not known if this optimal mirror map can be computed efficiently. Given these limitations, it is natural to consider a large, diverse set of mirror maps and choose from among those online, as we propose.

Parameter-free algorithms for online convex optimization, such as AdaHedge [45], MetaGrad [46], and AdaFTRL [47] use multiplicative-weights updates to aggregate over learning rates while keeping the underlying mirror map fixed, typically quadratic or entropic. In contrast, our method performs an explicit MW aggregation across multiple *distinct* mirror maps, enabling adaptation not only over step sizes but also over geometries. Other approaches [145] dynamically update the mirror map as more data is revealed.

7.1.3 Outline

In Section 7.2, we present preliminaries on OCO, OMD, and block norms. Then, in Section 7.3, we characterize regret for block norms for loss functions with sparse gradients. We also give two examples showing that OMD with block norms can achieve asymptotically better regret than either OPGD or OEG. In Section 7.4, we discuss the problem of combining different mirror maps. We first show that simple strategies such as using two mirror maps at alternate steps can lead to suboptimal $\Omega(T)$ regret. Then, we give a multiplicative weight update-based algorithm that automatically chooses the best mirror map from among a suite or *portfolio* of mirror maps, without losing a significant amount in regret. We then apply this framework to the portfolio of block norms. In Section 7.5, we verify our results through numerical simulations. Finally, we conclude in Section 7.6.

7.1.4 Techniques

Throughout this chapter, we will be concerned with proving upper and lower bounds for OMD with specific mirror maps. The upper bounds are standard, and depend on computing the diameter D of the polytope in the Bregman divergence and the dual gradient G of the loss functions (see Subsection 7.2.1). The regret is upper bounded by $O(DG\sqrt{T})$.

We now describe the template that all our lower bounds in this work follow. That is, given a specific mirror map h, we show a lower bound on the regret achieved by the corresponding OMD algorithm as described next. In each setting, we are given the starting point $x^{(1)}$ in the convex body and compute the optimal point x^*. Our loss functions are usually defined such that being 'far away' from x^* at any iteration incurs a large regret. Therefore, it is sufficient to show that a large number of iterates are 'far away' from x^*. Since $x^{(1)} \neq x^*$ in our examples, proving that the iterate $x^{(t)}$ and $x^{(1)}$ are 'close' is sufficient to prove that $x^{(t)}$ and x^* are 'far away'. Therefore, to show regret lower bounds, we upper bound the appropriate distance between $x^{(t)}$ and $x^{(1)}$.

7.2 Preliminaries

In this section, we give preliminaries on Online Mirror Descent (OMD) [138] and block norms [141].

7.2.1 Online Mirror Descent

Algorithm 10 `MirrorDescentStep`$(x, \mathcal{K}, h, \nabla f, \eta)$

input: point x in convex body $\mathcal{K}$, distance generating function h, loss function gradient ∇f, and step size η

$\theta = \nabla h(x)$ ▷ Map to dual space

$\theta' = \theta - \eta \nabla f(x)$ ▷ Move in direction of negative gradient

$y = (\nabla h)^{-1}(\theta')$ ▷ Map back to primal space

return $x' = \Pi_{\mathcal{K}}^{(h)}(y) := \arg\min_{z \in \mathcal{K}} B_h(z, y)$ ▷ Project on $\mathcal{K}$

Algorithm 11 `OnlineMirrorDescent` [138]

input: (i) an online convex optimization setup with a closed, bounded convex body $\mathcal{K}$, a time horizon T, (ii) a convex function h defined[2]on $\mathbb{R}^d$ with an invertible gradient ∇h, and (iii) a starting point $x^{(1)} \in \mathcal{K}$

parameters: step sizes $\eta^{(1)}, \ldots, \eta^{(T)} \in \mathbb{R}_{>0}$

for $t = 1, \ldots, T$ **do**

    play $x^{(t)}$ and observe loss $f^{(t)}$

    $x^{(t+1)} =$ `MirrorDescentStep`$(x^{(t)}, \mathcal{K}, h, \nabla f^{(t)}, \eta^{(t)})$

OMD is an algorithm for the Online Convex Optimization (OCO) setup described earlier, paramterized by the *mirror map* h. Formally, suppose $h : \mathbb{R}^d \to \mathbb{R}$ is a convex function (referred to as the mirror map hereafter) with invertible gradient that induces the *Bregman divergence*

$$B_h(x \| y) := h(x) - h(y) - \langle \nabla h(y), x - y \rangle \qquad \forall\, x, y \in \mathbb{R}^d.$$

[2]For simplicity, we assume that h is defined on $\mathbb{R}^d$; however, one must be more careful and define it an appropriate set $X \supseteq \mathcal{K}$ that depends on h.

Suppose further that h is μ-strongly convex with respect to some norm $\|\cdot\|$ on $\mathbb{R}^d$, i.e.,

$$B_h(x\|y) \geq \frac{\mu}{2}\|x-y\|^2 \qquad\qquad \forall\, x, y \in \mathbb{R}^d.$$

Denote by $\|\cdot\|^*$ the dual norm to $\|\cdot\|$. Given starting point $x^{(1)}$ for OMD, denote by $D_h := \sqrt{\max_{z\in\mathcal{K}} B_h(z\|x^{(1)})}$ the diameter of $\mathcal{K}$ under B_h. Then, with convex loss functions $f^{(1)}, \ldots, f^{(T)}$, the regret of OMD with Bregman divergence B_h, step-sizes $\eta_t = \eta$, and iterates $x^{(1)}, \ldots, x^{(T)}$ satisfies for any $x^* \in \mathcal{K}$:

$$\texttt{regret}(T) := \sum_{t\in[T]} \left(f^{(t)}(x^{(t)}) - f^{(t)}(x^*)\right) \leq \frac{D_h^2}{\eta} + \frac{\eta \sum_{t\in[T]} \left(\|f^{(t)}(x^{(t)})\|^*\right)^2}{2\mu}$$

If we further have the bound $\|\nabla f^{(t)}(x^{(t)})\|^* \leq G^*$, then this implies

$$\texttt{regret}(T) \leq \frac{D_h^2}{\eta} + \frac{T\eta(G^*)^2}{2\mu}.$$

Choosing $\eta = \frac{D_h}{G^*}\sqrt{\frac{2\mu}{T}}$ gives

$$\texttt{regret}(T) \leq \frac{\sqrt{2}D_h G^* \sqrt{T}}{\sqrt{\mu}} \tag{7.2}$$

This also holds in expectation when gradients are *stochastic*, assuming the bound

$$\mathbb{E}\left[\left(\|\nabla f^{(t)}(x^{(t)})\|^*\right)^2\right] \leq (G^*)^2.$$

7.2.2 Block Norms

Here we define block norms that interpolate between the L_1 and L_2 norms. Then, we define the corresponding mirror maps that are strongly convex with respect to these block norms, thus interpolating between the OPGD and OEG setups.

Given integer $d > 0$ and *blocks* $\mathcal{B} = \{B_1, \ldots, B_n\}$ that partition $[d]$, define the corre-

sponding *block norm* $\|\cdot\|_{[\mathcal{B}]}$ on $\mathbb{R}^d$ by

$$\|x\|_{[\mathcal{B}]} := \sum_{j=1}^{n} \|x_{B_j}\|_2 \tag{7.3}$$

as the sum of the L_2 norms of individual blocks $x_{B_j} \in \mathbb{R}^{B_j}$, where x_{B_j} is the restriction of x to the coordinates in B_j. In particular, when n divides d and all blocks have equal size $|B_j| = \frac{d}{n}$. Throughout this work, we assume that this equal-size partition is chosen uniformly at random among all such partitions at $t = 0$, and fixed thereafter. We call this the nth block norm and denote it by $\|x\|_{[n]}$.

It is easy to check that the dual of a block norm is simply the maximum of the L_2 norms of each block:

$$\|x\|_{[\mathcal{B}]}^* = \max_{j=1}^{n} \|x_{B_j}\|_2 \tag{7.4}$$

We refer to it as the *dual block norm*, or as the *nth dual block norm* in the special case of the nth block norm.

Mirror Descent Framework. Block norms are useful in the OMD framework since there exist functions that are strongly convex with respect to the nth block norm for each n. In particular, we have the following result:

Theorem 7.1 ([141])**.** *Given positive integers d, n such that n divides d, define*

$$\gamma_n = \begin{cases} 1 & \text{if } n = 1, \\ \frac{1}{2} & \text{if } n = 2, \\ \frac{1}{e \ln n} & \text{if } n > 2, \end{cases} \qquad p_n = \begin{cases} 2 & \text{if } n \leq 2, \\ 1 + \frac{1}{\ln n} & \text{if } n > 2. \end{cases}$$

Consider any equal-sized blocks $\mathcal{B} = \{B_1, \ldots, B_n\}$ that partition the coordinate set $[d]$. Define the mirror map

$$h_n(x) = \frac{1}{\gamma_n p_n} \sum_{j=1}^{n} \|x_{B_j}\|_2^{p_n}.$$

Then h_n is 1-strongly convex with respect to the nth block norm corresponding to $\mathcal{B}$ over the unit norm ball $\{x : \|x\|_{[n]} \leq 1\}$ of the nth block norm.

For given dimension d and integer n that divides d, consider a random equal-size partition of coordinates $[d]$ and the corresponding block norm $\|\cdot\|_{[n]}$ and mirror map h_n. We refer to the corresponding OMD algorithm with h_n as OMD_n, and denote its regret as $\texttt{regret}_n$. We get the following result as a corollary of the general OMD bound:

Corollary 7.1. *Consider an OCO setting with convex body $\mathcal{K} \subseteq \mathbb{R}^d$, a starting point $x^{(1)} \in \mathcal{K}$, and loss functions $f^{(t)}, t \in [T]$. Define diameter $D_n := \sqrt{\max_{z \in \mathcal{K}} B_{h_n}(z \| x^{(1)})}$ and $G_n = \max_{x \in \mathcal{K}, t \in [T]} \sqrt{\mathbb{E}\left[\left(\|\nabla f^{(t)}(x)\|^*_{[n]}\right)^2\right]}$.*

If $\mathcal{K} \subseteq \{x : \|x\|_{[n]} \leq 1\}$, then for appropriate step sizes, the expected regret of OMD_n is at most

$$\mathbb{E}[\texttt{regret}_n(T)] = O(D_n G_n \sqrt{T}).$$

If $\max_{x \in \mathcal{K}} \|x\|_{[n]} > 1$ but $\mathcal{K}$ is bounded, we can first rescale $\mathcal{K}$ to fit in the norm ball $\{x : \|x\|_{[n]} \leq 1\}$ and then use OMD on the rescaled convex body. The loss functions must also be appropriately rescaled.

7.3 Block Norms and Improved Regret

In this section, we first present a general recipe for bounding the regret with the nth block norm when the gradients of the loss functions are sparse. Then, we present two examples showing that OMD with intermediate block norms ($1 < n < d$) can achieve asymptotically better regret than both OPGD ($n = 1$ blocks) and OEG[3].

- **First example (probability simplex in d dimensions):** First, we show that for linear loss functions with sparse gradients, using a suitable block norm improves the regret upper bound in Corollary 7.1 by a factor $\Omega\left(\frac{\sqrt{\ln d}}{\ln \ln d}\right)$ over both OPGD and OEG. Next,

[3]Or $n = d$ blocks, when OEG does not apply.

we give an example of a concrete OCO setting where this improvement in regret is realized.

- **Second example (convex hull of $d+1$ points):** We show that for linear loss functions with sparse gradients, using a suitable block norm improves the regret upper bound in Corollary 7.1 by a factor $\widetilde{\Omega}\left(d^{1/6}\right)$ over both $n = 1$ (OPGD) and $n = d$ block norms. Next, we give an example of a concrete OCO setting where this improvement in regret is realized.

To our knowledge, our second example is the first demonstration of a polynomial-factor improvement in regret obtained by using mirror maps beyond OPGD or OEG.

7.3.1 Regret for Sparse Gradients

Consider an OCO setting where the gradients $\nabla f^{(t)}(x)$ are all 0-1 vectors and S-sparse, i.e., $\|\nabla f^{(t)}(x)\|_1 \leq S$. We say that such loss functions are *S-sparse*.

For OMD with nth block norm, we bound the quantity G_n, which is the expectation of the square of the dual block norm for such gradients. The proof of the lemma, which uses Bernstein's inequality for negatively associated random variables, is deferred to Section E.1.

Lemma 7.1. *Consider a vector $c \in \{0,1\}^d$ that is S-sparse, i.e., $\|c\|_1 \leq S$, and a block norm $\|\cdot\|_{[n]}$ induced by a random equal n-partition of $[d]$. Then the expected square of the dual norm $\mathbb{E}\left[\left(\|c\|_{[n]}^*\right)^2\right]$ is bounded above by*

$$\mathbb{E}\left[\left(\|c\|_{[n]}^*\right)^2\right] \leq 6\max\left\{\frac{S}{n}, \ln n\right\}.$$

Therefore, we get the following result using Corollary 7.1:

Theorem 7.2. *Consider an OCO setting with on convex body $\mathcal{K} \subseteq \mathbb{R}^d$, a starting point $x^{(1)} \in \mathcal{K}$, and S-sparse loss functions $f^{(t)}, t \in [T]$. Define $D_n := \sqrt{\max_{z\in\mathcal{K}} B_{h_n}(z\|x^{(1)})}$.*

Then, if $\mathcal{K}$ lies in the unit norm ball $\{x \in R^d : \|x\|_{[n]} \leq 1\}$ for the nth block norms, we have for appropriate step sizes that the expected regret of OMD$_n$ is at most

$$\mathbb{E}[\texttt{regret}_n(T)] = \begin{cases} O\left(\sqrt{\frac{S}{n}} D_n \sqrt{T}\right) & \text{if } n \leq \frac{S}{\ln S}, \\ O\left(\sqrt{\ln n}\, D_n \sqrt{T}\right) & \text{if } n > \frac{S}{\ln S}. \end{cases}$$

Proof. From Lemma 7.1, we get that

$$G_n := \max_{t \in [T], x \in \mathcal{K}} \|\nabla f^{(t)}(x)\|^*_{[n]} \leq \sqrt{6 \max\left\{\frac{S}{n}, \ln n\right\}}.$$

For $n \leq \frac{S}{\ln S}$, we have $\ln n \leq \ln S$ and $\frac{S}{n} \geq \ln S$, therefore, $G_n \leq \sqrt{\frac{6S}{n}}$. Now suppose $n \geq \frac{S}{\ln S}$. Since $\ln S \leq \sqrt{S}$ for all $S > 0$, we have $n \geq \frac{S}{\sqrt{S}} = \sqrt{S}$, so that $\ln n \geq \frac{1}{2} \ln S$, and $\frac{S}{n} \leq \ln S$. Therefore, $\max\left\{\frac{S}{n}, \ln n\right\} \leq 2 \ln n$, implying $G_n \leq \sqrt{12 \ln n}$. Corollary 7.1 then immediately implies the result. □

7.3.2 Logarithmic Improvement in Regret Over Simplex

In this section, we show that block norms can achieve asymptotic improvements in regret over the probability simplex for sparse loss functions using block norms. To apply Theorem 7.2, we need an upper bound on the diameter D_n of the probability simplex $\mathbf{\Delta}_d := \{x \in \mathbb{R}^d : x \geq 0, \sum_{i \in [d]} x_i = 1\}$ for the Bregman divergence B_{h_n} corresponding to the nth block norm. The following lemma provides this bound; its proof is deferred to Section E.2.

Lemma 7.2. *Let h_n denote the nth block norm over the probability simplex. Then*

$$\max_{x \in \mathbf{\Delta}_d} \sqrt{B_{h_n}(x \| x^{(1)})} \leq \sqrt{\frac{1}{\gamma_n}} \leq 2\sqrt{1 + \ln n},$$

where γ_n is as defined in Theorem 7.1.

Given this bound on D_n, we can upper bound the regret for any block norm using

Theorem 7.2. Specifically, we have that for any S-sparse loss functions over the probability simplex,

$$\mathbb{E}[\texttt{regret}_n(T)] = \begin{cases} O\left(\sqrt{\frac{S}{n}(1+\ln n)} \cdot \sqrt{T}\right) & \text{if } n \leq \frac{S}{\ln S}, \\ O\left((\ln n)\sqrt{T}\right) & \text{if } n > \frac{S}{\ln S}. \end{cases}$$

In particular, for $S \simeq \ln d$, we have $\mathbb{E}[\texttt{regret}_1(T)] = O(\sqrt{(\ln d)T})$, $\texttt{regret}_d(T) = O((\ln d)\sqrt{T})$, and $\texttt{regret}_S(T) = O((\ln \ln d)\sqrt{T})$. Thus, we expect that the Sth block norm gives a factor $\frac{\sqrt{\ln d}}{\ln \ln d}$ improvement over the best of OPGD and OEG. We formalize this claim now, also providing lower bounds on the regrets of OPGD and OEG, which are denoted $\texttt{regret}_{\text{euc}}$ and $\texttt{regret}_{\text{ent}}$ respectively.

Theorem 7.3. *There exists an OCO setting with sparse loss functions where*

$$\min\{\mathbb{E}[\texttt{regret}_{\text{euc}}(T)], \mathbb{E}[\texttt{regret}_{\text{ent}}(T)]\} = \Omega\left(\frac{\sqrt{\ln d}}{\ln \ln d}\right) \mathbb{E}[\texttt{regret}_S(T)]$$

for $S = \ln d$. That is, OMD with the Sth block norm improves the regret by a factor $\Omega\left(\frac{\sqrt{\ln d}}{\ln \ln d}\right)$ over OPGD and OEG.

To prove this theorem, it is sufficient to provide the OCO setting and prove lower bounds on $\mathbb{E}[\texttt{regret}_{\text{euc}}(T)]$ and $\mathbb{E}[\texttt{regret}_{\text{ent}}(T)]$.

OCO Setting. Our convex body is the probability simplex $\mathbf{\Delta}_d$. Each loss function $f^{(t)}, t \in [T]$ is defined as $f^{(t)}(x) = -\langle c^{(t)}, x\rangle$ where $c^{(t)}$ is a 0-1 vector with exactly S non-zero coordinates, chosen as follows: $c_1^{(t)} = 1$ and the other $S-1$ non-zero coordinates are chosen uniformly at random from $[2, d]$. All algorithms must start at point $x^{(1)} = \frac{1}{d}\mathbf{1}_d$. The time horizon $T \geq \ln d$, and $S = \ln d$. Clearly, $x^* = (1, 0, \ldots, 0)$, with $\sum_{t\in[T]} f^{(t)}(x^*) = -T$.

First, we note that as discussed above, $\texttt{regret}_S(T) = O((\ln \ln d)\sqrt{T})$. Further, we have that for all $x \in \mathbf{\Delta}_d$,

$$\mathbb{E}[f^{(t)}(x)] = -\sum_{i\in[d]} \mathbb{E}[c_i^{(t)}]x_i = \sum_{i\in[d]} \Pr(c_i^{(t)} = 1)x_i$$

$$= -x_1 - \frac{S-1}{d-1}\sum_{i\in[2,d]} x_i = -x_1 - \frac{S-1}{d-1}(1-x_1)$$
$$\geq -x_1 - \frac{S-1}{d-1}. \tag{7.5}$$

Lower Bound for OPGD. The Euclidean diameter $D_{\text{euc}} = \max_{z\in\boldsymbol{\Delta}_d} \|z - x^{(1)}\|_2 = \sqrt{(1-1/d)^2 + (d-1)(1/d)^2} = \sqrt{1-1/d}$.

Further, clearly, $\mathbb{E}[G_{\text{euc}}^2] \geq S$. Therefore, the step size η is

$$\eta = \frac{D_{\text{euc}}}{\sqrt{\mathbb{E}[G_{\text{euc}}^2]T}} \leq \frac{\sqrt{1-1/d}}{\sqrt{ST}}.$$

By induction, we get that $x_1^{(t)} \leq x_1^{(1)} + (t-1)\eta$. Define $T_0 = 1 + \frac{(1-1/d)}{2\eta} = 1 + \frac{1}{2}\sqrt{\left(1-\frac{1}{d}\right)ST}$. Then $x_1^{(t)} \leq \frac{1}{d} + \frac{1}{2}$ for all $t \in [T_0]$. By Equation 7.5, for all $t \in [T_0]$,

$$\mathbb{E}[f^{(t)}(x^{(t)}) - f^{(t)}(x^*)] \geq -\left(\frac{1}{2} + \frac{1}{d}\right) - \frac{S-1}{d-1} - (-1)$$
$$\geq \frac{1}{2} - \frac{S}{d} \geq \frac{1}{4}.$$

The last inequality holds since $S = \ln d \leq \frac{d}{2}$ for all large enough d. Therefore,

$$\mathbb{E}[\texttt{regret}_{\text{euc}}(T)] = \sum_{t\in[T]} \mathbb{E}[f^{(t)}(x^{(t)}) - f^{(t)}(x^*)]$$
$$\geq \sum_{t\in[T_0]} \mathbb{E}[f^{(t)}(x^{(t)}) - f^{(t)}(x^*)]$$
$$\geq \sum_{t\in[T_0]} \frac{1}{4} = \frac{T_0}{4} = \Omega\left(\sqrt{ST}\right) = \Omega\left(\sqrt{(\ln d)T}\right).$$

Lower Bound for OEG. Consider OMD with the entropic mirror map, i.e., $h(x) := \text{ent}(x) := \sum_{i\in[d]} x_i \ln x_i$. The step size is

$$\eta = \frac{\sqrt{\mu \max_{x\in\boldsymbol{\Delta}_d} B_{\text{ent}}(x\|x^{(0)})}}{G_{\text{ent}}^*\sqrt{T}}$$

Since $x^{(0)} = \frac{1}{d}\mathbf{1}_d$, we have $\sqrt{\max_{x \in \mathbf{\Delta}_d} B_{\text{ent}}(x \| x^{(0)})} = \sqrt{\ln d}$. Since B_h is 1-strongly convex with respect to the L_1 norm and $\nabla f^{(t)}(x)$ is S-sparse, we have $\mu = 1$ and $G^*_{\text{ent}} = 1$, so that $\eta = \sqrt{\frac{\ln d}{T}}$.

Since $\nabla f^{(t)}(x) = -c^{(t)}$ for all x, we have that

$$x_i^{(t+1)} = \begin{cases} \frac{x_i^{(t)} e^{\eta}}{e^{\eta} \sum_{i \in [S]} x_i^{(t)} + \sum_{i \in [S+1,d]} x_i^{(t)}} & \text{if } c_i^{(t)} = 1, \\ \frac{x_i^{(t)}}{e^{\eta} \sum_{i \in [S]} x_i^{(t)} + \sum_{i \in [S+1,d]} x_i^{(t)}} & \text{if } c_i^{(t)} = 0. \end{cases}$$

Therefore, by induction on t, $x_1^{(t)} \leq \frac{1}{d} e^{\eta t}$ for all $t \in [T]$. Define $T_0 = \sqrt{T \ln d}$. By Equation 7.5, for all $t \in [T_0]$,

$$\mathbb{E}[f^{(t)}(x^{(t)}) - f^{(t)}(x^*)] \geq 1 - \frac{1}{d} e^{\eta t} - \frac{S-1}{d-1} = \frac{d-S}{d-1} - \frac{1}{d} e^{\eta t}.$$

Also, clearly, $\mathbb{E}[f^{(t)}(x^{(t)}) - f^{(t)}(x^*)] \geq 0$ for all t. Therefore, the expected regret is

$$\begin{aligned} \mathbb{E}[\texttt{regret}_{\text{ent}}(T)] &= \sum_{t \in [T]} \mathbb{E}[f^{(t)}(x^{(t)}) - f^{(t)}(x^*)] \\ &\geq \sum_{t \in [T_0]} \left(\frac{d-S}{d-1} - \frac{1}{d} e^{\eta t} \right) \\ &\geq \left(\frac{d-S}{d-1} \right) \left(T_0 - \frac{1 - 1/d}{e^{\eta} - 1} \right) \\ &\geq \left(\frac{d-S}{d-1} \right) \left(T_0 - \frac{1 - 1/d}{\eta} \right) \\ &= \left(\frac{d-S}{d-1} \right) \left(\sqrt{T \ln d} - \left(1 - \frac{1}{d} \right) \sqrt{\frac{T}{\ln d}} \right). \end{aligned}$$

Since $S \leq \frac{d}{2}$, we have $\frac{d-S}{d-1} \geq \frac{1}{2}$. Further, $\frac{1}{2}\sqrt{T \ln d} \geq \left(1 - \frac{1}{d}\right) \sqrt{\frac{T}{\ln d}}$ if $d \geq 10$, so that $\mathbb{E}[\texttt{regret}_{\text{ent}}(T)] \geq \frac{1}{4}\sqrt{T \ln d}$.

7.3.3 Polynomial Improvement in Regret

In this section, we give an OCO setting where the regret improvement through intermediate block norms is polynomial in the dimension. Specifically, we give a polytope P and loss functions $f^{(1)}, \ldots, f^{(T)}$ such that using a block norm with $n \neq \{1, d\}$ blocks improves the regret by a polynomial in d factor:

Theorem 7.4. *There exists an OCO setting on a convex body $\mathcal{K} \subseteq \mathbb{R}^d$ with S-sparse loss functions where*

$$\min\{\mathbb{E}[\texttt{regret}_1(T)], \mathbb{E}[\texttt{regret}_d(T)]\} = \widetilde{\Omega}(d^{1/6})\mathbb{E}[\texttt{regret}_S(T)]$$

for $S = d^{1/3}$, where $\widetilde{O}$ ignores all terms that are logarithmic in d and T.

Fix dimension $d \geq 50$. For $A \geq 0$, define the polytope P as the convex hull of the standard unit vectors $\mathbf{e}_1, \ldots, \mathbf{e}_d$ and $A\mathbf{1}_d$. We set $A = d^{-2/3}$.

OCO Setting. For $t \in [T]$, define loss functions $f^{(t)}(x) = -\langle c^{(t)}, x\rangle$, where $c^{(t)} \in \{0, 1\}^d$ is a random vector with exactly S non-zero coordinates, defined as follows: $c_1^{(t)} = 1$ and the remaining $S - 1$ non-zero coordinates of $c^{(t)}$ are chosen uniformly at random from among the remaining $d - 1$ coordinates, independently at different $t \in [T]$. We fix sparsity $S = d^{1/3}$. The starting point $x^{(1)} = A\mathbf{1}_d$. We set convex body $\mathcal{K} = P$ defined above.

As before, our twofold strategy is as follows:

1. For $n = S$ blocks, we show using Corollary 7.1 that the corresponding regret is at most $\mathbb{E}[\texttt{regret}_S(T)] = \widetilde{O}(\sqrt{T})$.

2. For $n \in \{1, d\}$ blocks, we show that the corresponding regret $\mathbb{E}[\texttt{regret}_S(T)] = \widetilde{\Omega}(d^{1/6}\sqrt{T})$. To prove this, we show that after t iterations, $x^{(t)}$ must still be 'close' to $x^{(1)}$, and therefore must be 'far away' from x^*, and incur large regret as a consequence. This is similar to our approach in Subsection 7.3.2, except that the notion of

'close' and 'far away' will change, and that the proofs will be more involved.

Recall that Corollary 7.1 for block norms requires the polytope P to lie within the unit norm ball of the corresponding block norm. In this case, for certain block norms, P may not lie within the corresponding unit norm ball. For example, for $n = d$ blocks, the corresponding block norm is the L_1 norm, and P does not lie in the unit norm ball $\{x \in \mathbb{R}^d : \|x\|_1 \leq 1\}$ since $\|x^{(1)}\|_1 = Ad = d^{1/3} > 1$. For such norms, we must first rescale P before applying Corollary 7.1.

Upper Bound for Sth Block Norm. We will show that $\mathbb{E}[\texttt{regret}_S(T)] = \widetilde{O}(\sqrt{T})$. First, we claim that P lies in the unit norm ball $\{x : \|x\|_{[S]} \leq 1\}$, so that a rescaling is not needed. To see this, we note that $\max_{x \in P} \|x\|_{[S]}$ is achieved at a vertex, and therefore by symmetry, $\max_{x \in P} \|x\|_{[S]} = \max\{\|\mathbf{e}_1\|_{[S]}, \|x^{(1)}\|_{[S]}\}$. However, $\|\mathbf{e}_1\|_{[S]} = 1$ and $\|x^{(1)}\|_{[S]} = A \cdot S \cdot \sqrt{d/S} = A\sqrt{dS}$. Since $A = d^{-2/3}$ and $S = d^{1/3}$, we have $A\sqrt{dS} = 1$. Therefore, P lies in the unit norm ball $\{x : \|x\|_{[S]} \leq 1\}$.

Next, to apply Corollary 7.1, we need to upper bound $D_S := \sqrt{\max_{z \in \mathcal{K}} B_{h_S}(z\|x^{(1)})}$ and $G_S = \max_{x \in \mathcal{K}, t \in [T]} \sqrt{\mathbb{E}\left[\left(\|\nabla f^{(t)}(x)\|^*_{[n]}\right)^2\right]}$.

To bound G_S, recall the characterization of the dual norms of block norms (Equation 7.4). For blocks $\mathcal{B} = (B_1, \ldots, B_S)$, we have

$$\|y\|^*_{[S]} = \max_{j \in [n]} \|y_{B_j}\|_2.$$

In our setting, $-\nabla f^{(t)}$ is a 0-1 vector with exactly S non-zero coordinates. Since the blocks are all of equal size and chosen uniformly at random, we can apply Lemma 7.1 to get that for all x,

$$\mathbb{E}\left[\left(\|\nabla f^{(t)}(x)\|^*_{[S]}\right)^2\right] = 6 \max\left\{\frac{S}{S}, \ln S\right\} = 6 \ln S = O(\ln d). \tag{7.6}$$

Therefore, it remains to show that $D_S = \widetilde{O}(1)$ in order to show that $\mathbb{E}[\texttt{regret}_S(T)] =$

$\widetilde{O}(\sqrt{T})$. We defer the proof of this fact to Section E.3.

Lemma 7.3. *For the polytope P defined above and diameter $D_S = \sqrt{\max_{z\in\mathcal{K}} B_{h_S}(z\|x^{(1)})}$, we have $D_S = \widetilde{O}(1)$.*

Lower Bound on OPGD. To establish lower bounds on regret, we first show that being a significant distance away from $x^* = \mathbf{e}_1$ incurs significant regret. Recall that $f^{(t)}(x) = -\langle c^{(t)}, x\rangle$ for all $t \in [T]$, where $c_1^{(t)} = 1$ and exactly $S-1$ other coordinates or $c^{(t)}$ are non-zero, chosen uniformly at random. Therefore, for all $x \in P$ and all $t \in [T]$, we have

$$\mathbb{E}[f^{(t)}(x)] = -\sum_{i\in[d]} \Pr(c_i^{(t)} = 1)x_i = -x_1 - \frac{S-1}{d-1}\sum_{i\in[2,d]} x_i \geq -x_1 - \frac{S}{d}\sum_{i\in[2,d]} x_i.$$

In particular, since $\sum_{i\in[2,d]} x_i \leq \|x\|_1 \leq \|A\mathbf{1}_d\|_1 = Ad$ for all $x \in P$, we get $\mathbb{E}[f^{(t)}(x)] \geq -x_1 - \frac{S}{d}(Ad) = -x_1 - AS$. Since $A = d^{-2/3}$, $S = d^{1/3}$, and $f^{(t)}(x^*) = -1$, we have

$$\mathbb{E}[f^{(t)}(x) - f^{(t)}(x^*)] \geq -(x_1 + d^{-1/3}) + 1 = 1 - d^{-1/3} - x_1. \tag{7.7}$$

Therefore, to lower bound the regret of OPGD, we need to establish upper bounds on the first coordinates $x_1^{(t)}$ of the iterates $\{x^{(t)} : t \in [T]\}$ of OPGD.

We first compute the learning rate of OPGD. By symmetry, the diameter

$$\begin{aligned} D_{\text{euc}} &= \max_{x\in P} \|x - x^{(1)}\|_2 = \|\mathbf{e}_1 - A\mathbf{1}_d\|_2 \\ &= \sqrt{(1-A)^2 + (d-1)A^2} \leq (1-A) + dA \leq 2(1-A). \end{aligned}$$

The last inequality holds since $A = d^{-2/3}$. The gradient bound is $G_{\text{euc}} = \max_{t\in[T],x\in P} \|\nabla f^{(t)}(x)\|_2 = \sqrt{S} = d^{1/6}$. Therefore, $\eta = \frac{D_{\text{euc}}}{G_{\text{euc}}\sqrt{T}} \leq \frac{2(1-A)}{d^{1/6}\sqrt{T}}$.

At time step t, denote the set of non-zero coordinates in $\nabla f^{(t)} = -c^{(t)}$ as $C^{(t)} = \{i \in$

$[d] : c_i^{(t)} = 1\}$. The unconstrained update rule of OPGD for coordinates $i \in C^{(t)}$ is

$$y_i^{(t)} = x_i^{(t)} + \eta.$$

Therefore, we get by induction on t that for the sequence $x^{(t)}$ of iterates of OPGD,

$$x_1^{(t)} \leq A + (t-1)\eta.$$

Denote $T_0 = 1 + \frac{1}{2\eta} = 1 + \frac{d^{1/6}}{4(1-A)}\sqrt{T}$. Then for $t \leq T_0$, we get $x_t^{(t)} \leq A + \frac{1}{2} = d^{-2/3} + \frac{1}{2}$. From Equation 7.7,

$$\begin{aligned}
\texttt{regret}_{\text{euc}}(T) &= \sum_{t\in[T]} \mathbb{E}[f^{(t)}(x^{(t)}) - f^{(t)}(x^*)] \\
&= \sum_{t\in[T_0]} \mathbb{E}[f^{(t)}(x^{(t)}) - f^{(t)}(x^*)] \\
&\quad + \sum_{t=T_0+1}^{T} \mathbb{E}[\underbrace{f^{(t)}(x^{(t)}) - f^{(t)}(x^*)}_{\geq 0}] \\
&\geq \sum_{t\in[T_0]} (1 - d^{-1/3} - x_1^{(t)}) && (Equation\ 7.7) \\
&\geq (1 - d^{-1/3} - (1/2 + d^{-2/3}))T_0 \\
&\geq \frac{T_0}{8} && (\text{assume } d \geq 50) = \Omega(d^{1/6}\sqrt{T}).
\end{aligned}$$

Lower Bound on dth Block Norm OMD. Next, we prove that the regret of the dth block norm OMD, $\texttt{regret}_d(T) = \widetilde{\Omega}(d^{1/6}\sqrt{T})$. First, we note that P does not lie in the unit norm ball $\{x \in \mathbb{R}^d : \|x\|_1 \leq 1\}$ of the L_1 norm, since $\|A\mathbf{1}_d\| = Ad = d^{1/3} > 1$. So we must rescale the polytope so that it fits inside the L_1 norm ball before applying Corollary 7.1. Define the scaled polytope $\hat{P} = \frac{1}{Ad}P$ to be the convex hull of $\frac{1}{Ad}\mathbf{e}_1, \ldots, \frac{1}{Ad}\mathbf{e}_d$ and $\frac{1}{d}\mathbf{1}_d$. For convenience, we will denote this rescaling factor $R = Ad$. The loss functions must also be scaled by a factor R, and the new losses are denoted $\hat{f}^{(t)} = Rf^{(t)}$.

The algorithm will converge to the optimal point $z^* = \left(\frac{1}{R}, 0, \ldots, 0\right)$ in the scaled polytope $\hat{P}$. As before, we will show that it converges in a large number of steps, accumulating high regret. Specifically, we will show the following bound on the iterates of the algorithm:

Lemma 7.4. *The iterates $z^{(1)}, \ldots, z^{(T)} \in \hat{P}$ of OMD with dth block norm satisfy with high probability*

$$z_1^{(t)} \leq \frac{1}{d} + \frac{\sqrt{K}}{R\sqrt{R}}(t-1),$$

where $K = \frac{128}{T}\ln^2(dT)$.

Given this lemma, we prove that the total regret of the algorithm, denoted $\texttt{regret}_d(T) = \widetilde{\Omega}(d^{1/6}\sqrt{T})$. Indeed, for the optimal point $z^* = \left(\frac{1}{R}, 0, \ldots, 0\right)$, we have that $\mathbb{E}[\hat{f}^{(t)}(z^*)] = R\,\mathbb{E}[f^{(t)}(z^*)] = R(-1 \cdot \frac{1}{R}) = -1$. Further, for any $z \in \hat{P}$, analogous to Equation 7.7, we have for all $t \in [T]$ that

$$\mathbb{E}[\hat{f}^{(t)}(z)] = -Rz_1 - R \cdot \frac{S-1}{d-1} \sum_{i \in [2,d]} z_i \geq -Rz_1 - \frac{R(S-1)}{d} \geq -Rz_1 - \frac{RS}{d}.$$

Consider Lemma 7.4. Denote $T_0 = 1 + \frac{1}{2}\sqrt{\frac{R}{K}}$ where $K = \frac{128}{T}\ln^2(dT)$. Then, $z_1^{(t)} \leq \frac{1}{d} + \frac{1}{2R}$ for all $t \leq T_0$. We can lower bound the regret between $1 \leq t \leq T_0$ as follows using the above equation and Lemma 7.4:

$$\begin{aligned}
\sum_{t \in [T_0]} \mathbb{E}\left[\hat{f}^{(t)}(z^{(t)}) - \hat{f}^{(t)}(z^*)\right] &\geq \sum_{t \in [T_0]} \left((-Rz_1^{(t)} - \frac{RS}{d} - (-1)\right) \\
&\geq \sum_{t \in [T_0]} \left((-R\left(\frac{1}{2R} + \frac{1}{d}\right) - \frac{RS}{d} - (-1)\right) \\
&\geq T_0\left(\frac{1}{2} - \frac{(S+1)R}{d}\right).
\end{aligned}$$

Since $R = Ad = d^{1/3}$ and $S = d^{1/3}$, we get that for all $d \geq 50$, this is at least $\frac{T_0}{4} =$

$\Omega\left(\sqrt{\frac{R}{K}}\right) = \widetilde{\Omega}\left(d^{1/6}\sqrt{T}\right)$. This proves the regret bound

$$\texttt{regret}_d(T) \geq \sum_{t\in[T_0]} \mathbb{E}\left[\hat{f}^{(t)}(z^{(t)}) - \hat{f}^{(t)}(z^*)\right] = \widetilde{\Omega}\left(d^{1/6}\sqrt{T}\right).$$

To prove Lemma 7.4, we show that the Bregman divergence $B_{h_d}(z^{(t)}\|z^{(t+1)})$ is sandwiched between

$$\frac{1}{2}\|z^{(t)} - z^{(t+1)}\|_1^2 \leq B_{h_d}(z^{(t)}\|z^{(t+1)}) \leq \frac{K}{2R}. \tag{7.8}$$

Further, we prove in Section E.3 that this L_1 norm bound implies the coordinate-wise bound in Lemma 7.4.

The lower bound in Equation 7.8 follows from Theorem 7.1 (since B_{h_d} is 1-strongly convex with respect to the L_1 norm in the L_1 norm ball). The upper bound is more involved and uses the structure of loss functions and the polytope. We give a high-level sketch of the proof here, with details deferred to Section E.3.

Specifically, if $y^{(t)}$ denotes the intermediate point between $z^{(t)}$ and $z^{(t+1)}$, then $z^{(t+1)}$ is the minimizer of $B_{h_d}(z\|y^{(t)})$. Therefore, using the generalized Pythagorean theorem for Bregman divergences,

$$B_{h_d}(z^{(t+1)}\|z^{(t)}) \leq B_{h_d}(z^{(t+1)}\|y^{(t)}) + B_{h_d}(z^{(t)}\|y^{(t)}) \leq 2B_{h_d}(z^{(t)}\|y^{(t)}).$$

Therefore, it is sufficient to upper bound $B_{h_d}(z^{(t)}\|y^{(t)})$. However, given $z^{(t)}$, we can explicitly compute $y^{(t)}$ as a function of $z^{(t)}$, dimension d, and the step size η of the algorithm.

7.4 Learning the Optimal Mirror Map

In this section, we discuss learning the optimal mirror map for a given OCO setting from among a family of mirror maps, e.g., block norm mirror maps.

7.4.1 Alternating between Mirror Maps is Suboptimal

The most natural way to combine mirror maps is to use different mirror maps at different time steps $t \in [T]$. We construct instances where naively alternating between mirror maps can incur regret $\Omega(T)$, thus demonstrating that more sophisticated strategies are needed to combine mirror maps. In particular, we give instances where using the OPGD and OEG at alternate steps leads to $\Omega(T)$ regret. Contrast this with OPGD or OEG, each of which results in $O(\sqrt{T})$ regret. Our result holds for *all (fixed) step sizes*, thus showing that adjusting the step sizes is not sufficient to produce an optimal-regret algorithm.

Theorem 7.5. *An algorithm that alternates between OPGD and OEG with fixed step sizes has regret $\Omega(T)$ in the worst case, irrespective of the step sizes.*

To prove this result, we give a specific OCO setting in 2 dimensions where the alternating algorithm incurs $\Omega(T)$ regret.

OCO Setting. The convex body is the probability simplex $\mathbf{\Delta}_2 = \{x \in \mathbb{R}^2 : x_1 + x_2 = 1, x_1, x_2 \geq 0\}$ in 2 dimensions. All algorithms must start at $x^{(1)} = (1/2, 1/2)$. We are given that the loss functions $f^{(t)}, t \in [T]$ all satisfy $\|\nabla f^{(t)}(x)\|_2 \leq 2$ for all $x \in \mathbb{R}^2$. These will be chosen adversarially based on the choices made by the algorithm.

Consider an algorithm that runs `MirrorDescentStep` (Algorithm 10) with the Euclidean mirror map $h_{\text{euc}}(x) = \frac{1}{2}\|x\|_2^2$ on odd steps and with the entropic mirror map $h_{\text{ent}} = \sum_{i \in [d]} x_i \ln x_i$ on even steps. Suppose the step sizes for these are η_{euc} and η_{ent} respectively. Using a case analysis on the step sizes, we show that there always exist loss functions so that this algorithm has regret $\Omega(T)$.

Case 1: $\eta_{\text{euc}} \geq \frac{16}{T}$. The loss functions are specified as follows:

$$f^{(t)}(x) = \begin{cases} -x_1 & \text{if } t \text{ is odd,} \\ 0 & \text{if } t \text{ is even and } t \leq T/8, \\ -2x_2 & \text{if } t \text{ is even and } t > T/8. \end{cases}$$

Since $\sum_{t\in[T]} f^{(t)}(x) = -\frac{T}{2}x_1 - \frac{3T}{4}x_2$, the optimal point $x^* = \mathbf{e}_2 = (0, 1)$, and the optimal total loss is $-\frac{3T}{4}$.

We will claim that the algorithm hits and stays at vertex $\mathbf{e}_1 = (1, 0)$ in $\leq \frac{T}{8}$ time steps and then does not move, i.e., $x^{(t)} = \mathbf{e}_1$ for all $t \geq \frac{T}{8}$. Assuming this claim, the loss of the algorithm is at least $-\frac{1}{2} \cdot \frac{T}{8}$ in the first $\frac{T}{8}$ time steps, and exactly $-\frac{1}{2} \cdot \frac{7T}{8}$ thereafter. Thus, the total loss of the algorithm is $-\frac{T}{2}$. Therefore, the regret is $-\frac{T}{2} + \frac{3T}{4} = \frac{T}{4} = \Omega(T)$.

To see the claim, first note that $x^{(t+1)} = x^{(t)}$ for all even $t \leq T/8$ since $f^{(t)}(x) = 0$. Therefore, any movement in $x^{(t)}$ is due to the Euclidean mirror map update step, which moves $x^{(t)}$ towards $\mathbf{e}_1$ since $\nabla f^{(t)}(x) = (-1, 0)$ for all odd t. Specifically, for all odd t,

$$y_1^{(t+1)} = x_1^{(t)} + \eta_{\text{euc}}, \qquad x_1^{(t+1)} = \Pi_{\text{euc}}(y^{(t+1)}) = \min\left(x_1^{(t)} + \frac{\eta_{\text{euc}}}{2}, 1\right).$$

Thus, as long as $x_1^{(t)} < 1$, the Euclidean update step moves it towards 1 by a distance of $\frac{\eta_{\text{euc}}}{2}$ at every odd time step $t \leq T/8$. Since $\eta_{\text{euc}} \geq \frac{16}{T}$, and since $x_1^{(1)} = 1/2$, this implies that $x_1^{(T/8)} = \frac{1}{2} + \frac{T}{16} \times \frac{\eta_{\text{euc}}}{2} = 1$.

Once $x^{(t)} = (1, 0)$, the entropic map cannot revive zero mass on $x_2^{(t)}$, since its update rule takes

$$x_2^{(t+1)} \propto x_2^{(t)} \exp(-\eta_{\text{ent}} \nabla_2 f^{(t)}(x^{(t)})) = 0.$$

Case 2: $\eta_{\text{euc}} < \frac{16}{T}$. The loss functions are specified as follows:

$$f^{(t)}(x) = \begin{cases} -x_2 & \text{if } t \text{ is odd,} \\ 0 & \text{if } t \text{ is even.} \end{cases}$$

Clearly, the optimal point is $x^* = \mathbf{e}_2 = (0, 1)$, with optimal total loss $-T/2$. As in Case 1, the Euclidean mirror map moves towards the optimal by a distance of $\leq \frac{\eta_{\text{euc}}}{2}$ at every odd time step. There is no movement at even time steps. That is, for all t,

$$x_2^{(t)} \leq \frac{1}{2} + \frac{\eta_{\text{euc}}}{2} \cdot \frac{t}{2}.$$

Therefore, since $\eta_{\text{euc}} < \frac{16}{T}$, we have $x_2^{(t)} \leq \frac{3}{4}$ for all $t \leq T/16$. The regret up to $T/16$ is therefore $\geq \left(1 - \frac{3}{4}\right) \cdot \frac{T}{32} = \frac{T}{128}$. Since the regret is nonnegative thereafter, the total regret is at least $\frac{T}{128}$.

Finally, we remark that the adversary need not handle the two cases separately in designing the loss functions: it can give the loss functions in Case 1 with probability $1/2$ and those in Case 2 with probability $1/2$ without any knowledge of the step sizes of the algorithm. Thus, our construction is completely oblivious to step sizes.

7.4.2 Multiplicative Weight Update over Mirror Maps

Having established that intermediate block norms can yield significant gains, we now ask whether an algorithm can adaptively learn which mirror map to use – without prior knowledge of the structure of loss functions.

We show that a Multiplicative Weight (MW) update algorithm that maintains each mirror map as an 'expert', obtains regret at most the minimum of the best mirror map's regret, plus an additional $O(\rho\sqrt{\ln N}\sqrt{T})$ term, where $\rho \geq \max_{x,z\in\mathcal{K},t\in[T]} f^{(t)}(x) - f^{(t)}(z)$ is an upper bound on the loss function differentials. The algorithm requires knowing ρ. The following theorem formalizes this:

Algorithm 12 `MirrorWeights`

input: convex body $\mathcal{K} \subseteq \mathbb{R}^d$, time horizon T, starting point $x^{(1)} \in \mathcal{K}$, strongly convex distance generating functions $h_1, \dots, h_N$, upper bound $\rho \geq \max_{x,z \in \mathcal{K}, t \in [T]} f^{(t)}(x) - f^{(t)}(z)$ on the differential in loss function values

parameters: step sizes $\{\eta_\ell^{(t)}, \ell \in [N], t \in [T]\}$, and learning rate $\varepsilon > 0$

initialize $X \in \mathbb{R}^{d \times N}$ with all columns $X_\ell \leftarrow x^{(1)}$
initialize probabilities $p_\ell \leftarrow \frac{1}{N}$ for all $\ell \in [N]$
for $t = 1, \dots, T$ **do**
    play $x^{(t)} = \sum_{\ell=1}^N p_\ell X_\ell$ and observe loss $f^{(t)}$
    for $\ell = 1, \dots, N$ **do**
        $p_\ell \leftarrow p_\ell \cdot \exp\left(-\frac{\varepsilon f^{(t)}(X_\ell)}{\rho}\right)$
        $X_\ell \leftarrow \texttt{MirrorDescentStep}(X_\ell, \mathcal{K}, h_\ell, \nabla f^{(t)}, \eta_\ell^{(t)})$
    normalize $p \leftarrow p / \|p\|_1$

Theorem 7.6 (Combining mirror maps)**.** *Consider an OCO setting with $f^{(t)}(x) - f^{(t)}(z) \leq \rho$ for all $t \in [T]$ and $x, z \in \mathcal{K}$. Given N mirror maps $h_1, \dots, h_N$ and their step sizes, if $T \geq \ln N$, then* `MirrorWeights` *(Algorithm 12) achieves regret at most*

$$\texttt{regret}(T) \leq \min_{\ell \in [N]} \texttt{regret}_{h_\ell}(T) + 2\rho\sqrt{T \ln N}$$

for learning rate $\varepsilon = \sqrt{\frac{\ln N}{T}}$.

The proof follows standard MW arguments and is deferred to Section E.4. Next, we combine this idea with the $1 + \log_2 d$ block norm mirror maps corresponding to $n \in \{2^0, 2^1, \dots, 2^{\log_2 d}\}$. In this setting, even the input on the step sizes is not required: given the Euclidean diameter $D_{\text{euc}} = \max_{x,z \in \mathcal{K}} \|x - z\|_2$, and the Euclidean gradient norm bound $G_{\text{euc}} = \max_{x \in \mathcal{K}, t \in [T]} \|\nabla f^{(t)}(x)\|_2$, the diameter D and gradient G for any other block norm is within factor d of D_{euc} and G_{euc} respectively. Thus, we can search over the step sizes for each block norm (up to a factor of 2) by making $O(\log d)$ copies of each block norm mirror map, but with different step sizes[4]. We have the following result:

Theorem 7.7 (Combining block norms)**.** *Consider an OCO setting with convex body $\mathcal{K}$*

[4]This search over step sizes is essentially the MetaGrad algorithm in [46].

that lies within the L_1 norm ball in $\mathbb{R}^d$. Suppose the number of time steps $T \geq 4 \ln \ln d$, and suppose the differential in loss functions $\max_{x,z \in \mathcal{K}, t \in [T]} f^{(t)}(x) - f^{(t)}(z) = 1$. Then, given the Euclidean diameter $D_{\text{euc}} = \max_{x,z \in \mathcal{K}} \|x - z\|_2$ and Euclidean gradient norm bound $G_{\text{euc}} = \max_{x \in \mathcal{K}, t \in [T]} \|\nabla f^{(t)}(x)\|_2$, `MirrorWeights` *(Algorithm 12) with block norms achieves regret at most*

$$\texttt{regret}(T) = O(\sqrt{\ln \ln d}) \cdot \min_{n \in \{2^0, \ldots, 2^{\log_2 d}\}} D_n G_n \sqrt{T},$$

where D_n is the diameter under the corresponding Bregman divergence B_{h_n} and G_n is the gradient norm upper bound in the dual norm to nth block norm.

Proof. By our construction above and by Theorem 7.6, we have that

$$\texttt{regret}(T) = O\left(\min_n D_n G_n \sqrt{T} + \rho \sqrt{(\ln \ln d) T}\right).$$

Thus, it is sufficient to show that $\rho \leq D_n G_n$ for each n. To show this, suppose $\rho = f^{(t)}(x) - f^{(t)}(z)$ for some $x, z \in \mathcal{K}$ and $t \in [T]$. Then,

$$\begin{aligned}
\rho = f^{(t)}(x) - f^{(t)}(z) &= \int_0^1 \langle \nabla f^{(t)}(z + \beta(x - z)), x - z \rangle \, \mathrm{d}\beta \\
&\leq \int_0^1 \left\| \nabla f^{(t)}(z + \beta(x - z)) \right\|_{[n]}^* \|x - z\|_{[n]} \, \mathrm{d}\beta \\
&\leq G_n \|x - z\|_{[n]}.
\end{aligned}$$

Therefore, it is sufficient to show that $\|x - z\|_{[n]} \leq O(D_n)$. Recall that $D_n^2 = \max_{x' \in \mathcal{K}} B_{h_n}(x' \| x^{(1)}) \geq \max_{x' \in \mathcal{K}} \frac{1}{2} \|x' - x^{(1)}\|_{[n]}$, where the inequality holds since h_n is 1-strongly convex with respect to $\|\cdot\|_{[n]}$ (Theorem 7.1). Therefore,

$$2\sqrt{2} D_n \geq \|x - x^{(1)}\|_{[n]} + \|z - x^{(1)}\|_{[n]} \geq \|x - z\|_{[n]}.$$

This completes the proof. □

Note that the restriction that $\mathcal{K}$ lies in the L_1 norm ball can be removed by rescaling $\mathcal{K}$, as long as it is bounded.

7.5 Experiment

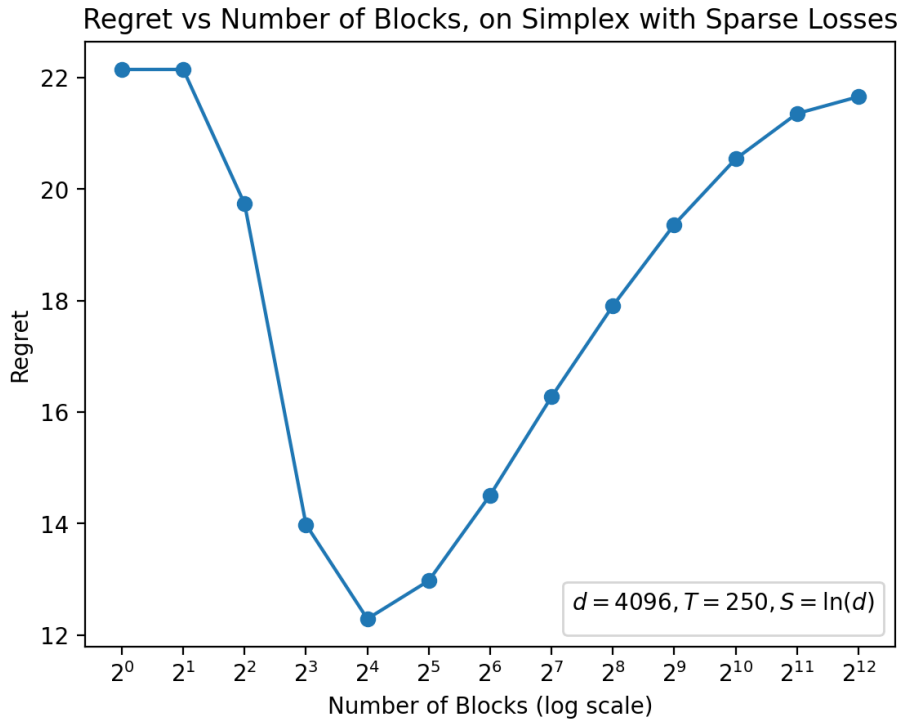


Figure 7.1: Total regret against the number of blocks in the distance generating function after $T = 250$ time steps.

In this section, we describe a numerical experiment over the probability simplex to show that intermediate block norms can achieve regret better than either OPGD or OEG.

OCO Setting. Our convex body is the probability simplex $\mathbf{\Delta}_d = \{x \in \mathbb{R}^d : \sum_{i\in[d]} x_i = 1, x \geq 0\}$ in $d = 2^{12} = 4096$ dimensions. The time horizon $T = 250$, and for each $t \in [T]$, the loss function $f^{(t)}(x) = -\langle c^{(t)}, x\rangle$ where $c^{(t)} \in \{0,1\}^d$ is a S-sparse vector where $S = \lfloor \ln d \rfloor$, described next.

At each t, we choose a main coordinate $i(t) \in [d]$ such that $c^{(t)}_{i(t)} = 1$ and the remaining $S-1$ non-zero coordinates of $c^{(t)}$ are chosen uniformly at random from among the remain-

ing $d-1$ coordinates. $i(t)$ is chosen as follows: fix $T_0 = \lfloor 2\sqrt{T} \rfloor$. If $t \leq T_0$, then $i(t)$ alternates between 1 and 2, and for $t > T_0$, $i(t)$ alternates between 3 and 4. That is,

$$i(t) = \begin{cases} 1 & \text{if } t \leq T_0 \text{ and } t \text{ is odd,} \\ 2 & \text{if } t \leq T_0 \text{ and } t \text{ is even,} \\ 3 & \text{if } t > T_0 \text{ and } t \text{ is odd,} \\ 4 & \text{if } t > T_0 \text{ and } t \text{ is even.} \end{cases} \tag{7.9}$$

Experiment. We ran OMD with the nth block norm for $n \in \{2^0, 2^1, \ldots, 2^{12}\}$, with $n = 1$ corresponding to OPGD and $n = d = 2^{12}$ (roughly) corresponding to OEG. For each block norm, we searched for the step size[5] $\eta \in [10^{-2}, 10^2]$ that minimizes the final regret $\texttt{regret}_n(T)$ after T time steps.

Results. As Figure 7.1 shows, $\texttt{regret}_n(T)$ first decreases with the number n of blocks and then increases. In particular, using $n = 16$ blocks leads to a regret of < 12.5, as opposed to a regret of > 21 for either OPGD or OEG. This is in line with what is expected from our theoretical results and shows the usefulness of block norms in certain regimes.

7.6 Conclusion

The choice of the mirror map in OMD captures the geometry of the OCO problem, and can be exploited to show regret improvements in specific regimes of loss functions. As we showed, specific *block norm mirror maps* can yield polynomial-in-dimension improvements in regret in certain regimes over standard algorithms like OPGD and OEG.

However, very little may be known about the loss functions *a priori*, and adhering to a fixed mirror map may lead to suboptimal regret. This is where our contribution of a *portfolio of block norm mirror maps* comes in. We show that we can switch between

[5] Optimized using ternary search in the log-space.

various block norms at runtime, achieving the regret rate of the best block norm, up to a $\sqrt{\ln \ln d}$ factor.

A natural direction for future work is to explore other families of mirror maps that interpolate between Euclidean and entropic geometries and to investigate whether a small, universal portfolio of geometries can achieve near-optimal regret across all OCO settings. More broadly, our results suggest viewing geometry itself as a learnable component of online optimization.

CHAPTER 8
CONCLUSION AND FUTURE DIRECTIONS

And indeed there will be time

To wonder, "Do I dare?" and, "Do I dare?"

– T. S. Eliot

We studied the notion of portfolios, which are subsets of feasible solutions to optimization problems such that optimizing over a portfolio is approximately the same as optimizing over the complete set of feasible solutions. We designed portfolios for several problems, spanning from combinatorial optimization to online learning to reinforcement learning.

There are many ways to interpret what portfolios are. Some of these interpretations explain the thinking behind our results, while others motivate new applications. We present some of these next.

Portfolios can be viewed as tools to make an optimization problem free of modeling choices. This was the view we took in various fairness applications, where modeling choices about fairness were built into the objective function as a parameter: for example, as 'p' in L_p norms (Chapter 3) or p-means (Chapter 6), or as the weight vector w in ordered norms (Chapter 4). By giving portfolios that work across all p or across all weight vectors, we largely decouple them from the process of optimization and algorithm design. Additionally, we hand back the power to make decisions about fairness to decision-makers, who can look at the portfolio of options and choose their preferred option (rather than fixing an abstract parameter for fairness). Our applications to suggest new pharmacies (see Figure 1.1) and resource allocation policies (see Figure 1.2) are based on this idea.

One can also view portfolios as tools for parameter-free optimization. For example, in Chapter 7, we construct portfolios across the problem geometry (choice of mirror map

through parameter 'n' for the number of blocks) and step sizes (additional parameters). These automatically tune to the problem geometry. More generally, in any optimization or learning problem, one can seek a small portfolio of parameters that approximates learning across the set of all parameters. This can have several benefits, ranging from computational speed-ups (when portfolios are small, so one can restrict to searching over the parameters in the portfolio only) to systematizing hyperparameter tuning in machine learning (which is usually ad-hoc).

Another way to view portfolios is as a covering problem: say that a feasible solution *covers* an objective if it is α-approximate for it. Then finding the smallest α-approximate portfolio is the same as finding the minimum cover of solutions for all objectives. This view gives structural insights into the problem at hand and can have interesting applications for personalization. For example, different ML models may fit the needs of different people in a population better, but can we find a small subset that is guaranteed to cater to every person? This view also relates to the question of algorithmic robustness: how much can we change the objective before a good solution becomes a poor solution?

Another useful way to think about portfolios is as a discretization of a continuous space (of feasible solutions), where the discretization is tailored to a specific geometry (induced by the objective functions). We took this view in Chapter 3 to design portfolios for conic combinations, for example. This raises interesting questions about the suitability of different discretizations in different settings. It may also help translate continuous optimization problems into discrete optimization problems. This would be particularly helpful if the portfolio size is polynomial in the problem dimension, as is the case with most of our results.

These interpretations suggest that portfolios can (1) serve as a useful practical tool that (2) unify similar ideas across several fields and (3) raise interesting algorithmic, combinatorial, and optimization questions.

Finally, we list some interesting open directions for the problems studied in this the-

sis and for portfolios more broadly. These are in addition to the problem-specific open problems listed separately in each chapter.

(α, β)-portfolios. Portfolios guarantee that for all objectives in a given class $\mathbf{C}$, there is some portfolio solution that is an α-approximation for the objective. Often, it is also useful for the solutions to have a universal performance guarantee of some kind. For example, we may also want each portfolio solution x to not be too bad for any objective $f \in \mathbf{C}$, i.e., x should be a β-approximation for each $f \in \mathbf{C}$ for some β. That is, each portfolio solution is also a simultaneous β-approximation for $\mathbf{C}$. It would be interesting to explore this, or similar *bicriteria* guarantees with one 'local' and one global criterion.

Is maximization harder than minimization? Chapter 3, Chapter 4, and Chapter 5 discussed many classes of objectives for minimization problems: conic combinations, L_p norms, top-ℓ norms, ordered norms, and symmetric monotonic norms. Chapter 6 discussed the class of p-means for maximization problems, the direct analogue of L_p norms in minimization. However, (1) in Chapter 6, we had to make the necessary assumption that the underlying vectors were bounded away from 0 to get a portfolio for p-means, and the portfolio size depended on this bound. No such assumption was necessary for L_p norms in minimization (Chapter 3). Further, it is unclear how (and whether) the techniques employed for symmetric monotonic norms for minimization extend to analogous concave functions for maximization. As we argue below, it seems that such an extension is not possible.

For example, consider our technique for obtaining portfolios for symmetric monotonic norms: we exploited the fact that each symmetric monotonic norm in d dimensions is $O(\log d)$-approximated by some ordered norm. This allowed us to focus on portfolios for ordered norms, which can be easier to handle since they are piecewise linear convex functions. The analogue of this result does not hold for concave functions: the following claim shows that the function $f(x) = \sqrt{x_1 x_2}$ in 2 dimensions[1] cannot be approximated by

[1]This function, like symmetric monotonic norms, is invariant to the permutation of coordinates, is mono-

the concave analogues of ordered norms:

Claim 8.1. *For $x \in \mathbb{R}^2_{\geq 0}$, define the function $f(x) := \sqrt{x_1 x_2}$ cannot be α-approximated by an ordered concave function for any $\alpha > 0$.*

Proof. Suppose not, so that there exist $0 \neq w_1 \geq w_2 \geq 0$ such that $g_w(x) := w_1 x_1^{\uparrow} + w_2 x_2^{\uparrow}$ is an α-approximation to $f(x) = \sqrt{x_1 x_2}$ for all $x \in \mathbb{R}^2_{\geq 0}$, i.e., $\frac{1}{\alpha} f(x) \leq g_w(x) \leq \alpha f(x)$ for all $x \geq 0$. Then for all $t \geq 1$, choosing $x = (\kappa, 1)$ gives that $\frac{1}{\alpha}\sqrt{\kappa} \leq w_1 + w_2\kappa \leq \alpha\sqrt{\kappa}$. The second inequality implies $w_2 = 0$, but in that case the first inequality cannot hold. □

Further, it is also unclear if restricting the domain – as we did in Chapter 6 for maximizing p-means functions – helps in this case. Specifically, even if we restrict that each coordinate of each vector $x \in \mathcal{D}$ must be bounded between 1 and some $\kappa > 1$, the above example shows that the best approximation we can hope for is $\Omega(\sqrt{\kappa})$.

Online optimization with portfolios. A promising direction for future work is to develop a unified theory of *online optimization over small portfolios of decisions*. Many classical online problems – ranging from clustering and facility location to caching, ensemble selection, and robust decision-making – share the structural feature that the learner must repeatedly choose a subset of feasible actions rather than a single point. These problems naturally involve two competing forces: the desire for flexibility, which pushes the algorithm to adapt its chosen subset in response to new information, and the desire for stability, which penalizes excessive changes through switching or movement costs. A general model that simultaneously captures these effects would provide a common lens through which diverse settings could be viewed.

One can imagine an online framework where, in each round, the learner selects a subset of at most k feasible solutions and incurs a worst-case loss over an adversarially chosen class of functions, while also paying a cost for modifying the subset relative to the previous round. Such a formulation interpolates smoothly between several well-studied settings:

tone in each coordinate, and positively homogenous.

online convex optimization (where $k = 1$), dynamic clustering and facility location (where at most k facilities can be open at any time, and some cost must be paid to switch open facilities), caching and paging (where the subset represents the cache), and multi-expert or ensemble methods (where k active predictors are maintained). At the same time, it introduces new conceptual challenges, as the learner's state is now a *combinatorial object* whose evolution must be controlled.

Building a theory for this class of problems could require new algorithmic ideas and analytical tools that go beyond current techniques for either combinatorial online learning or metric facility location. The interplay between robustness (through the max-min structure of the loss), sparsity (through the constraint that at most k feasible solutions can be played at any point), and stability (through the modification cost) seems largely unexplored in the general setting. Understanding the trade-offs between these forces could lead to principled algorithms for a wide range of applications in machine learning, optimization, and operations research, where decisions must be both adaptive and stable. This would offer a natural continuation of the themes of this thesis – especially the use of structured decision spaces and portfolios – while opening the door to a broad family of new problems.

Randomized portfolios. Another interesting direction is to ask if allowing randomization strengthens the portfolio guarantees. Specifically, if we are allowed to use (feasible) convex combinations of the solutions in the portfolio, can we improve the approximate guarantee for a given portfolio size? In this case, given class $\mathbf{C}$ of objectives, and some set $\mathcal{D}$ of feasible solutions, the approximation ratio of a portfolio $X \subseteq \mathcal{D}$ is defined as

$$\max_{f \in \mathbf{C}} \min_{\lambda \in \mathbf{\Delta}_X} f\left(\sum_{x \in X} \lambda_x x\right), \tag{8.1}$$

where $\mathbf{\Delta}_X$ is the probability simplex in $\mathbb{R}^X$.

This has a natural connection with the concept of *integrality gap* in approximation algorithms. Given a combinatorial problem, a common approach is to write an integer

program (IP) for it so that the vertices of the IP correspond to feasible solutions of the combinatorial problem. Frequently, the IP is relaxed to a convex program (CP). The optimal (fractional) solution to the LP is efficiently computed using standard machinery and then rounded to a feasible solution of the original problem. The integrality gap is defined as the ratio of the optimal over IP to the optimal over CP.

This connects to randomized portfolios defined above as follows: when the portfolio size is 1, and there is a single function $\mathbf{C} = \{f\}$ to optimize, then the optimal of (Equation 8.1) is precisely the optimal value of f for the original problem. However, as the portfolio size increases, fractional solutions are allowed, and eventually the optimal of (Equation 8.1) converges to the optimal value of f over CP.

In summary, the study of portfolios highlights how much structure can be revealed by looking across objectives rather than optimizing any single one. As stated above, many questions remain open, from sharper bounds to broader classes of objectives and dynamic or data-driven variants.

Appendices

APPENDIX A

OMITTED PROOFS FROM Chapter 3

We give proofs omitted from Chapter 3 here. First, we give hardness results for FSFL in Section A.1. Then, we give proofs omitted from the analysis of FSFL in Section A.2. Finally, in Section A.3, we prove that portfolios for top-ℓ norms are not portfolios for L_p norms, and can in fact have much bigger sizes.

A.1 Hardness Results for FSFL

In this section, we prove Theorem 3.3, which states that FSFL is hard to approximate within constant factors (assuming P $\neq$ NP), and that even checking whether a solution is feasible for the problem may be NP-hard. This implies that bicriteria approximation oracles are unavoidable for the problem.

Theorem 3.3. *Unless P = NP, (A) FSFL is inapproximable to within any constant factor even when the objective is the sum of client distances, and (B) there is no polynomial-time algorithm to check the feasibility of a solution to FSFL.*

Proof. We show that solving FSFL to within any constant factor implies a polynomial-time algorithm for the Subset Sum problem, which in turn implies P = NP. Recall that in the Subset Sum problem, we are given a set $A = \{a_1, \ldots, a_T\}$ of positive integers and need to determine whether we can partition A into two subsets with equal sum.

Consider a metric space with $|C| = T + 2T^2$ clients and $|F| = 2$ facilities. Each of the two facilities f_1 and f_2 has opening cost $c = c_{f_1} = c_{f_2} = \frac{1}{2}\sum_{t\in[T]} a_t$. There are also T^2 clients, each with revenue 0 at the location of each of the two facilities. The remaining T clients have revenues $a_1, \ldots, a_T$ respectively, and each of them is at a distance of 1 from each facility; see Figure A.1. We set the subsidy parameter $\delta = 0$, and the objective

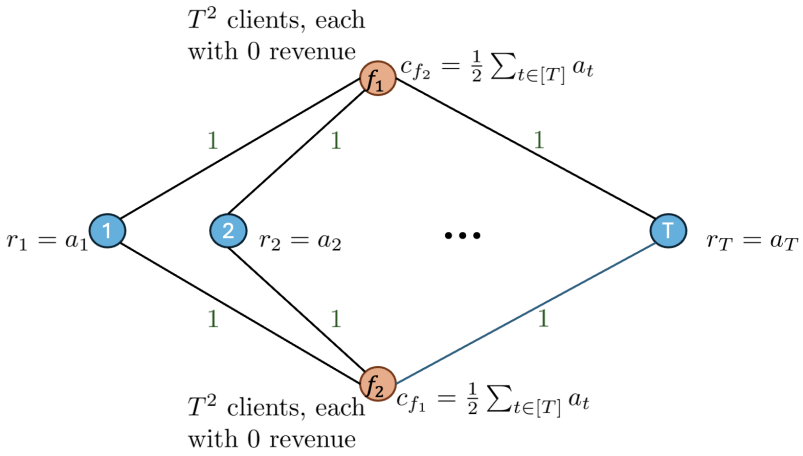


Figure A.1: The FSFL instance used in proof of Theorem 3.3

function $g : \mathbb{R}^C \to \mathbb{R}$ is the L_1 norm, i.e., we seek to minimize the sum of distances of clients to their assigned facilities.

Consider an algorithm $\mathcal{A}$ that is a constant-factor approximation for FSFL. We construct an algorithm for subset-set as follows: (i) Run $\mathcal{A}$ on the above instance. (ii) If the solution (F', Π) returned by $\mathcal{A}$ opens both facilities, declare "Yes" (i.e., there is a subset $S \subseteq A$ whose elements sum exactly $\frac{1}{2}\sum_{t\in T} a_t$). (iii) If (F', Π) opens only one $|F'| = 1$ facility, declare "No".

The key observation is that the following statements are equivalent:

1. A can be partitioned into two subsets with equal sums.

2. Solution returned by $\mathcal{A}$ has objective value $O(T)$

3. Facility set F' output by algorithm $\mathcal{A}$ opens both facilities, i.e., $|F'| = 2$.

It is easy to see that (2) implies (3): if one of the facilities is not open, all T^2 clients at that location travel distance 2 to the other facility so that the objective value is $\geq 2T^2$. Next, if (3) holds, then since this solution is feasible for FSFL, the total loss must be $\delta \times R = 0 \times \sum_{j\in C} r_j = 0$. That is, we must have $\sum_{j:\Pi(j)=f_1} r_j \geq c_{f_1} = \frac{1}{2}\sum_{t\in[T]} a_t$ and similarly $\sum_{j:\Pi(j)=f_2} r_j \geq c_{f_2} = \frac{1}{2}\sum_{t\in[T]} a_t$. Since $\sum_t a_t = \sum_j r_j = \sum_{j:\Pi(j)=f_1} r_j + \sum_{j:\Pi(j)=f_2} r_j$, we get a partition of A into two sets of equal sum.

Finally, we show that (1) implies (2): let $(S, A \setminus S)$ be the partition of A with equals sums. Consider the following solution to FSFL: open both facilities, and assign each $a_t \in S$ to f_1 and each $a_t \in A \setminus S$ to f_2. Similar to the argument above, this solution is feasible for the problem. Further, it has objective value T, which is the best possible. Since $\mathcal{A}$ is an $O(1)$-approximation for FSFL, its objective value must be $O(T)$. This concludes the proof of Part 1.

This equivalence also implies that checking feasibility is NP-hard: indeed, the solution with both facilities open is feasible if and only if A can be partitioned into two subsets with equal sums. □

A.2 Omitted Algorithms and Proofs for FSFL

We present omitted algorithms and proofs for the rounding algorithm for FSFL in Section 3.5 here. First, we present algorithm α-`PointRounding` and its proof in Subsection A.2.1. Lemma 3.5, which bounds the subsidy of `RoundToIntegralFacilities`, is proven in Subsection A.2.2. Proof of Lemma 3.6 that gives integral assignments given integral facilities is given in Subsection A.2.3.

A.2.1 Algorithm α-`PointRounding`

We give the α-point rounding algorithm from [71] for completeness in Algorithm 13.

We restate the guarantee of the algorithm here:

Lemma 3.2. *The fractional solution $(\overline{x}, \overline{y})$ output by α-`PointRounding`(x, y) (Algorithm 13) satisfies:*

1. *It is Δ-close where $\Delta_j \leq 4 \max\left(1, \frac{1}{\delta}\right) \tau_j$ for all $j \in C$, where $\tau_j = \sum_f x_{j,f} \texttt{dist}_{j,f}$ is the expected distance in (x, y). That is, for any facility f,*

$$\overline{x}_{j,f} > 0 \implies \texttt{dist}_{j,f} \leq 4 \max\left(1, \frac{1}{\delta}\right) \tau_j.$$

Algorithm 13 α-PointRounding(x, y)

input: fractional solution (x, y)
output: fractional solution $(\overline{x}, \overline{y})$

set $\alpha = \frac{\delta+2}{2(\delta+1)}$
for each client $j \in C$ **do**
    find order π on facilities in F so that $\texttt{dist}_{j,\pi(1)} \leq \texttt{dist}_{j,\pi(2)} \leq \ldots$
    for indices $k = 1, 2, \ldots$ denote the prefix sum $z_{j,k} := \sum_{i \leq k} x_{j,\pi(i)}$
    let k_j be the smallest index such that $z_{j,k_j} \geq \alpha$
    set

$$\overline{x}_{j,\pi(i)} = \begin{cases} \frac{1}{z_{j,k_j}} \cdot x_{j,\pi(i)} & \text{if } i \leq k_j, \\ 0 & \text{if } i > k_j. \end{cases}$$

for each facility $f \in F$, set $\overline{y}_f = \frac{1}{\alpha} y_f$
return $(\overline{x}, \overline{y})$

2. *The total loss of $(\overline{x}, \overline{y})$ is at most fraction 2δ of the total revenue, i.e., $\sum_{f \in F} \overline{\ell_f} \leq 2\delta \sum_{j \in C} r_j$.*

To prove this lemma, we first need another result for α-PointRounding:

Lemma A.1. *For all clients $j \in C$ and all facility subsets $F_1 \subseteq F$,*

$$\sum_{f \in F_1} \overline{x}_{j,f} \geq -(1 - \alpha) + \sum_{f \in F_1} x_{j,f}.$$

Proof. Note that we have $\sum_{f \in F} x_{j,f} = \sum_{f \in F} \overline{x}_{j,f} = 1$ along with nonnegativity of $x, \overline{x}$, and therefore α-PointRounding can be viewed as shifting some mass from the prior distribution x_j to the posterior distribution $\overline{x}_j$. Intuitively, the lemma says that the algorithm can shift no more than fraction $1 - \alpha$ of the mass away from any subset F_1 of facilities.

To see this, we use some notation from the algorithm: π is the ordering on facilities such that $\texttt{dist}_{j,\pi(1)} \leq \texttt{dist}_{j,\pi(2)} \leq \ldots$, and k_j is the least index k such that $\sum_{i \leq k} x_{j,\pi(i)} \geq \alpha$ Call facility $\pi(i)$ *massive* if $i \leq k_j$ and *light* otherwise. Therefore the total prior mass of light facilities is at most $1 - \sum_{i \leq k} x_{j,\pi(i)} \leq 1 - \alpha$. By construction, $\overline{x}_{j,\pi(i)} < x_{j,\pi(i)}$ if and

only if i is light. Therefore, for any $F_1 \subseteq F$, at most $1-\alpha$ mass can be reduced from it:

$$\sum_{f \in F_1} x_{j,f} - \sum_{f \in F_1} \overline{x}_{j,f} \leq 1 - \alpha. \qquad \square$$

Proof of Lemma 3.2. (Part 1) This part follows the argument from [71]. We present the proof here for completeness. For any client $j \in C$, by definition,

$$\begin{aligned}\tau_j &= \sum_i \texttt{dist}_{j,\pi(i)} x_{j,\pi(i)} = \sum_{i<k_j} \texttt{dist}_{j,\pi(i)} x_{j,\pi(i)} + \sum_{i \geq k_j} \texttt{dist}_{j,\pi(i)} x_{j,\pi(i)} \\ &\geq \sum_{i \geq k_j} \texttt{dist}_{j,\pi(i)} x_{j,\pi(i)} \geq \sum_{i \geq k_j} \Delta_j x_{j,\pi(i)} = \Delta_j \sum_{i \geq k_j} x_{j,\pi(i)} \geq \Delta_j (1-\alpha),\end{aligned}$$

where the second inequality holds since $\Delta_j = \max_{f:\overline{x}_{j,f}>0} \texttt{dist}_{j,f} = \texttt{dist}_{j,\pi(k_j)} \geq \texttt{dist}_{j,\pi(i)}$ for all $i \geq k_j$ by definition of ordering π. The third inequality holds since k_j is the first index such that $\sum_{i \leq k_j} x_{j,\pi(i)} \geq \alpha$, so that $\sum_{i<k_j} x_{j,\pi(i)} < \alpha$. Therefore, since $\alpha = \frac{\delta+2}{2(\delta+1)}$, we have

$$\Delta_j \leq \frac{1}{1-\alpha} \tau_j = \frac{2(\delta+1)}{\delta} \tau_j \leq \frac{4}{\min(1,\delta)} \tau_j.$$

(Part 2) Intuitively, loss for a facility increases due to an increase in the operating cost $c_f y_f$, and a decrease in client revenue. Denote by $F_1 = \{f \in F : c_f \overline{y}_f > \sum_{j \in C} r_j \overline{x}_{j,f}\}$ the set of unprofitable facilities in fractional solution $(\overline{x}, \overline{y})$. Then,

$$\begin{aligned}\sum_{f \in F} \overline{\ell_f} = \sum_{f \in F_1} \overline{\ell_f} &= \sum_{f \in F_1} \left(c_f \overline{y}_f - \sum_{j \in C} r_j \overline{x}_{j,f} \right) \\ &= \frac{1}{\alpha} \sum_{f \in F_1} c_f y_f - \sum_{j \in C} r_j \sum_{f \in F_1} \overline{x}_{j,f}.\end{aligned}$$

Next, from Lemma A.1, we have that $\sum_{f \in F_1} \overline{x}_{j,f} \geq -(1-\alpha) + \sum_{f \in F_1} \overline{x}_{j,f}$ for all clients

$j \in C$, so that the above becomes

$$\sum_{f \in F} \overline{\ell_f} \leq \frac{1}{\alpha} \sum_{f \in F_1} c_f y_f - \sum_{j \in C} r_j \sum_{f \in F_1} x_{j,f} + (1-\alpha) \sum_{j \in C} r_j.$$

Next, we note that $c_f y_f \leq \ell_f + \sum_{j \in C} r_j x_{j,f}$ since (x, y) satisfies Equation 3.6 in Equation IP, so that the above becomes

$$\begin{aligned}
\sum_{f \in F} \overline{\ell_f} &\leq \frac{1}{\alpha} \left(\sum_{f \in F_1} \ell_f + \sum_{j \in C} r_j \sum_{f \in F_1} x_{j,f} \right) - \sum_{j \in C} r_j \sum_{f \in F_1} x_{j,f} + (1-\alpha) \sum_{j \in C} r_j \\
&= \frac{1}{\alpha} \sum_{f \in F_1} \ell_f + \left(\frac{1}{\alpha} - 1 \right) \sum_{j \in C} r_j \sum_{f \in F_1} x_{j,f} + (1-\alpha) \sum_{j \in C} r_j \\
&\leq \frac{1}{\alpha} \cdot \delta \sum_{j \in C} r_j + \left(\frac{1}{\alpha} - 1 \right) \sum_{j \in C} r_j + (1-\alpha) \sum_{j \in C} r_j \\
&\leq \left(\frac{\delta}{\alpha} + 2 \left(\frac{1}{\alpha} - 1 \right) \right) \sum_{j \in C} r_j.
\end{aligned}$$

The second last inequality follows since (x, y) has subsidy $\leq \delta$, and the last inequality follows since $(1 - \alpha) \leq (1/\alpha - 1)$ for all $\alpha \in (0, 1)$. Setting $\alpha = \frac{\delta+2}{2(\delta+1)}$ gives $\frac{\delta}{\alpha} + 2\left(\frac{1}{\alpha} - 1\right) = \frac{\delta+2}{\alpha} - 2 = 2\delta$. □

A.2.2 Proof of Lemma 3.5

For every core client $j^* \in C^*$,

$$\begin{aligned}
c_{f(j^*)} &= c_{f(j^*)} \sum_{f \in \texttt{feas}_{j^*}} \overline{x}_{j^*,f} && (\textit{Equation } 3.4) \\
&\leq c_{f(j^*)} \sum_{f \in \texttt{feas}_{j^*}} \overline{y}_f && (\textit{Equation } 3.5) \\
&\leq \sum_{f \in \texttt{feas}_{j^*}} c_f \overline{y}_f. && (\text{step } 5)
\end{aligned}$$

Further, the revenue of clients assigned to $f(j^*)$ is

$$\begin{aligned}\sum_{j\in C} x'_{j,f(j^*)} &\geq \sum_{j\in C} r_j \sum_{f\in \texttt{feas}_{j^*}} \overline{x}_{j,f} && \text{(step 8)}\\ &= \sum_{f\in \texttt{feas}_{j^*}} \sum_{j\in C} r_j \overline{x}_{j,f}\\ &\geq \sum_{f\in \texttt{feas}_{j^*}} \left(c_f \overline{y}_f - \overline{\ell}_f\right). && (Equation\ 3.6)\end{aligned}$$

Denote unprofitable facilities in (x', y') by $F_1 = \{f \in F : \ell'_f > 0\}$. Since the only open facilities in (x', y') are $f(j^*), j^* \in C^*$, we get from above that the total loss of (x', y') is

$$\begin{aligned}\sum_{f\in F} \ell'_f &= \sum_{j^*: f(j^*)\in F_1} \left[c_{f(j^*)} - \sum_{j\in C} r_j \overline{x}_{j,f(j^*)}\right]\\ &\leq \sum_{j^*: f(j^*)\in F_1} \left[\sum_{f\in \texttt{feas}_{j^*}} \left[c_f \overline{y}_f - (c_f \overline{y}_f - \overline{\ell}_f)\right]\right]\\ &= \sum_{j^*: f(j^*)\in F_1} \sum_{f\in \texttt{feas}_{j^*}} \overline{\ell}_f \leq \sum_{j^*\in C^*} \sum_{f\in \texttt{feas}_{j^*}} \overline{\ell}_f.\end{aligned}$$

Since sets $\texttt{feas}_{j^*}, j^* \in C^*$ are disjoint for different core clients j^*, we get that this is bounded by $\sum_{f\in F} \overline{\ell}_f$. □

A.2.3 Proof of Lemma 3.6

In this section, we prove Lemma 3.6 that given a δ'-subsidized Δ'-close fractional solution (x', y') with integral y', obtains a $(\delta' + \theta)$-subsidized Δ'-close integral solution (x'', y').

Proof of Lemma 3.6. We prove the guarantee of distances first, and then on the losses.

Distances. By construction (Lemma 3.7), x'' only assigns j to some facility f such that $x'_{j,f} > 0$ (i.e., f is in the support of x'_j). Therefore, $\texttt{dist}_{j,f} \leq \Delta'_j$ by definition.

Subsidy. For brevity, let us denote the initial load/revenue of facility f by $r'(f) := \sum_{j\in F_C} x'_{j,f} r_j$, the final load/revenue by $r''(f) := \sum_{j\in F_C} x''_{j,f} r_j$, and the change in revenue by $\rho(f) := r''(f) - r'(f)$. The above then says that $\rho(f) \leq r_{j(f)}$ for some $j \in C_f := \{j' \in$

$C : x'_{j,f} > 0\}$ for all $f \in F$. By the small revenues assumption, $r_{j(f)} \leq \theta c_f$, so that

$$r''(f) - r'(f) = \rho(f) \leq \theta c_f.$$

Denote $F_\uparrow$ to be the set of all facilities f whose load/revenue increases, i.e., $\rho(f) \geq 0$, and similarly $F_\downarrow$ is the set of all facilities f whose load/revenue decreases. Since the total revenue $r(C) := \sum_{j \in C} r_j$ is constant, the total decrease in revenue across facilities $F_\downarrow$ must come from the total increase in revenue across facilities $F_\uparrow$. Therefore, to track the total decrease in revenue in $F_\downarrow$ (and therefore the increase in total loss), we will track the total increase in revenue across facilities in $F_\uparrow$. Consider a facility $f \in F_\uparrow$. There are three cases:

1. f was profitable in (x', y'), i.e., we had $c_f \leq r'(f)$. In this case, the contribution of f to the increase in revenue is

$$\rho(f) \leq \theta c_f \leq \theta r'(f) = \theta(r''(f) - \rho(f)) \leq \theta r''(f).$$

2. f was unprofitable in (x', y') and loss ℓ'_f of f in (x', y') was at least $\rho(f)$, i.e., $\ell'_f \geq \rho(f)$. In this case, the new loss of f in (x'', y') is $\ell''_f = \ell'_f - \rho(f)$. That is, loss decreases by $\rho(f)$. Then, any decrease in revenue in $F_\downarrow$ due to $\rho(f)$ is compensated by this decrease in loss.

3. f was unprofitable in (x', y') and loss ℓ'_f of f in (x', y') was at most $\rho(f)$, i.e., $\ell'_f \leq \rho(f)$. In this case,

$$\rho(f) \leq \theta c_f = \theta \left(\ell'_f + r'(f)\right) \leq \theta \left(\rho(f) + r'(f)\right) = \theta r''(f).$$

The total increase in revenue in cases 1 and 3 combined is therefore at most

$$\theta \sum_{f \in F_\uparrow} r''(f) = \theta \sum_{f \in F_\uparrow} \sum_{j \in C} x''_{j,f} r_j = \theta \sum_{j \in C} r_j \sum_{f \in F_\uparrow} x''_{j,f} \leq \theta \sum_{j \in C} r_j.$$

That is, the total decrease in loss overall when rounding from (x', y') to (x'', y') is upper bounded by $\theta \sum_{j \in C} r_j$, thus increasing the subsidy by at most θ. □

A.3 Gap Between Portfolio Sizes for Top-ℓ Norms and L_p Norms

In this section, we show that portfolios for one class of interpolating functions may not be portfolios for another class of interpolating functions. Specifically, we give a set $\mathcal{D}$ and base functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ where the class of top-ℓ norms admits an *optimal* portfolio of size 2 but any $O(1)$-approximate portfolio for L_p norms must have size $\simeq (\ln d)^{1/3}$. This shows that large gaps between portfolio sizes for different norms are possible. Compare this with the result for portfolios of size 1 or simultaneous approximations, where [9] show that simultaneous α-approximations for top-ℓ norms are also simultaneous α-approximations for L_p norms.

Lemma A.2. *For all large enough $d \in \mathbb{Z}_{\geq 0}$, there exists a set $\mathcal{D} \subseteq$ and base functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$ such that*

1. *There is an optimal (i.e., 1-approximate) portfolio X of size 2 for all top-ℓ norms.*
2. *Any $O(1)$-approximate portfolio X' for all L_p norms must have size $\Omega\left(\left(\frac{\ln d}{\ln \ln d}\right)^{1/3}\right)$.*

Proof. Let $S = S(d)$ be a super-constant that we will fix later, and N be the largest integer such that $S^{N^2} \leq d$. Then $N = \Omega(\sqrt{\log_S d})$. For $s \in [1, N]$, define vectors $v(s) \in \mathbb{R}^d$ as:

$$v(s) = (\underbrace{S^{-2s}, \ldots, S^{-2s}}_{S^{s^2}}, 0, \ldots, 0).$$

Let $\mathcal{D} = \{v(1), \ldots, v(N)\}$. Base functions $h_i(x) = x_i$ for all $x \in \mathcal{D}$. We prove Part 1

of the lemma statement first. Specifically, we will show that either $v(1)$ or $v(N)$ is optimum for all top-ℓ norms, so that $X = \{v(1), v(N)\}$ is an optimal portfolio for all top-ℓ norms. We have that the top-ℓ norm of $v(s)$ is

$$\|v(s)\|_{(\mathbf{1}_\ell)} = \begin{cases} \ell S^{-2s} & \text{if } \ell \leq S^{s^2}, \\ S^{s^2-2s} & \text{if } \ell > S^{s^2}. \end{cases}$$

Fix ℓ. For s such that $S^{s^2} < \ell$, $\|v(s)\|_{(\mathbf{1}_\ell)} = S^{s^2-2s}$ increases as s increases since $s \geq 1$. For s such that $S^{s^2} > \ell$, $\|v(s)\|_{(\mathbf{1}_\ell)} = \ell S^{-2s}$ decreases as s increases. Therefore, for each top-ℓ norm, either $v(1)$ or $v(N)$ is optimum. This proves Part 1.

We move to Part 2. Consider L_p norms for $p \in [1, N]$. Then we claim that for appropriate choice of $S = S(d)$, (1) for all $p \in [1, N]$, $\arg\min_{v(s)} \|v(s)\|_p = v(p)$. That is, vector $v(p)$ has the minimum norm $\|\cdot\|_p$ among all vectors in $\mathcal{D}$, and (2) for all $p \in [1, N]$ and $s \neq p$, $v(s)$ is not an $O(1)$-approximation for minimizing $\|\cdot\|_p$. Together, the two claims imply that any $O(1)$-approximate portfolio for L_p norms, $p \in [1, N]$ must contain each of $v(1), \ldots, v(N)$. Note first

$$\|v(s)\|_p = \left(S^{s^2} \cdot S^{2ps}\right)^{1/p} = S^{\frac{s^2}{p}-2s}. \tag{A.1}$$

To show that this is minimum at $s = p$, consider $\phi(x) = \frac{x^2}{p} - 2x$. It attains its minimum at $x = p$. Since $S > 1$, this implies that $\arg\min_{v(s)} \|v(s)\|_p = v(p)$, and the minimum is S^{-p}.

Further, for any $s \neq p$, write $s = p+\theta$ for $\theta \in [1, N]$. We have $\log_S \|v(s)\|_p = \frac{(p+\theta)^2}{p} - 2(p+\theta) = (p+\theta)\left(1 + \frac{\theta}{p} - 2\right) = \frac{\theta^2-p^2}{p} = \frac{\theta^2}{p} - p = \frac{\theta^2}{p} + \log_S \|v(p)\|_p$. Therefore, $\frac{\|v(s)\|_p}{\|v(p)\|_p} \geq S^{\frac{\theta^2}{p}} \geq S^{\frac{1}{p}} = \exp((\ln S)/p)$ Since $p \leq N$, this is at least $\exp(\frac{\ln S}{N})$. Choose S such that $\ln S = (\ln d)^{1/3}(\ln\ln d)^{2/3}$. Then $N = \Theta(\sqrt{\log_S d}) = \Theta\left(\sqrt{\frac{\ln d}{\ln S}}\right) = \Theta\left(\left(\frac{\ln d}{\ln\ln d}\right)^{\frac{1}{3}}\right)$. Therefore, $\ln\left(\frac{\|v(s)\|_p}{\|v(p)\|_p}\right) \geq \frac{\ln S}{N} = \Theta(\ln\ln d)$. That is, $\|v(s)\|_p = \omega(\|v(p)\|_p)$. The size of portfolio $\{v(1), \ldots, v(N)\}$ for L_p norms is $\Theta\left(\left(\frac{\ln d}{\ln\ln d}\right)^{1/3}\right)$. □

APPENDIX B

OMITTED PROOFS FROM Chapter 4

We supply proofs omitted from Chapter 4 here.

B.1 Omitted proofs from Section 4.2

Proof of Lemma 4.1. 1. For any objective $h \in \mathbf{C}$,

$$\min_{x \in X_2} h(x) \leq \alpha_2 \min_{x \in X_1} h(x) \leq \alpha_2 \alpha_1 \min_{x \in \mathcal{D}} h(x).$$

The first inequality follows since X_2 is an α_2-approximate portfolio for $\mathbf{C}$ over X_1, and the second follows since X_1 is an α_1-approximate portfolio for $\mathbf{C}$ over $\mathcal{D}$.

2. For each $h \in \mathbf{C}$,

$$\min_{x \in \mathcal{D}} h(x) = \min_{i \in [n]} \min_{x \in \mathcal{D}_i} h(x) \leq \min_{i \in [n]} \alpha \min_{x \in X_i} h(x) = \alpha \min_{x \in \cup_{i \in [n]} X_i} h(x).$$

Therefore, $\cup_{i \in [n]} X_i$ is an α-approximate portfolio for $\mathbf{C}$ over $\mathcal{D}$. □

Next, we prove Lemma 4.5 that gives $(1+\varepsilon)$-approximate portfolio of size $\mathrm{poly}(d^{1/\varepsilon})$ for symmetric monotonic norms for arbitrary feasible set $\mathcal{D}$ and base objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\geq 0}$. Our proof is a slight modification of [20]'s proof that counts the number of ordered norms up to a $(1+\varepsilon)$-approximation.

Proof of Lemma 4.5. Denote $v^* = \min_{x \in \mathcal{D}} \|\mathbf{h}(x)\|_\infty$, with the corresponding vector denoted x^*. Let $\overline{\mathcal{D}} = \{x \in \mathcal{D} : \|\mathbf{h}(x)\|_\infty \leq dv^*\}$. We first claim that $\overline{\mathcal{D}}$ is an optimal portfolio for all symmetric monotonic norms over $\mathcal{D}$, i.e., for each symmetric monotonic norm $\|\cdot\|$, the corresponding minimum norm point $\arg\min_{x \in \mathcal{D}} \|\mathbf{h}(x)\| \in \overline{\mathcal{D}}$. To see this,

let $\overline{x} = \arg\min_{x \in \mathcal{D}} \|\mathbf{h}(x)\|$. Then,

$$\begin{aligned}
\|\mathbf{h}(\overline{x})\|_\infty \|(1,0,\ldots,0)\| &\le \|\mathbf{h}(\overline{x})\| && (\|\cdot\| \text{ is symmetric}) \\
&\le \|\mathbf{h}(x^*)\| && (\text{optimality of } \overline{x}) \\
&\le \|\mathbf{h}(x^*)\|_\infty \|(1,\ldots,1)\| \\
&\le v^* d \|(1,0,\ldots,0)\|.
\end{aligned}$$

This implies that $\|\mathbf{h}(\overline{x})\|_\infty \le dv^*$, or that $\overline{x} \in \overline{\mathcal{D}}$. Next, we will place all vectors in $\overline{\mathcal{D}}$ in one of $d^{O(1/\varepsilon)}$ buckets such that for any two vector $\mathbf{h}(x), \mathbf{h}(y)$ in the same bucket, $\mathbf{h}(x) \preceq (1+\varepsilon)\mathbf{h}(y)$ and $\mathbf{h}(y) \preceq (1+\varepsilon)\mathbf{h}(x)$, so that by Lemma 2.1, $\|\mathbf{h}(x)\| \simeq_{1+\varepsilon} \|\mathbf{h}(y)\|$ for all symmetric monotonic norms $\|\cdot\|$. Consequently, it is sufficient to pick just one vector in each bucket to get a $(1+\varepsilon)$-approximate portfolio for all symmetric monotonic norms over $\mathcal{D}$.

Denote $T = \lceil \log_{1+\frac{\varepsilon}{3}} d \rceil$. Each bucket $B(a_1, \ldots, a_T)$ is specified by an increasing sequence $a_1 \le a_2 \le \cdots \le a_T$ of integers that lie in $[0, 2T]$. The number of such sequences is $\binom{3T}{T} \le 3^T = d^{O(1/\varepsilon)}$, bounding the number of buckets. Let $c_i = \lfloor (1+\varepsilon/3)^i \rfloor$ for $i \in [T]$. Then x lies in bucket $B(a_1, \ldots, a_T)$ where $a_i = \left\lfloor \log_{1+\frac{\varepsilon}{3}} \left(\frac{1}{v^*} \|\mathbf{h}(x)\|_{(\mathbf{1}_{c_i})} \right) \right\rfloor$.

First, we show that this assignment is valid, i.e., each $a_i \in [0, 2T]$. Indeed,

$$\frac{1}{v^*} \|\mathbf{h}(x)\|_{(\mathbf{1}_{c_i})} \le \frac{1}{v^*} c_i \|\mathbf{h}(x)\|_\infty \le \frac{d\|\mathbf{h}(x)\|_\infty}{v^*} \le d^2.$$

The final inequality follows since $x \in \overline{\mathcal{D}}$. Therefore, $a_i \le \log_{1+\frac{\varepsilon}{3} d^2} \le 2T$. Next, we claim that for any $x, y \in B(a_1, \ldots, a_d)$, $\mathbf{h}(x) \preceq (1+\varepsilon)\mathbf{h}(y)$. Fix any $k \in [d]$, and let $i \in [0, T]$ such that $c_i \le k < c_{i+1}$. Note that by definition of a_i, we have $a_i \le \log_{1+\frac{\varepsilon}{3}} \left(\frac{1}{v^*} \|\mathbf{h}(x)\|_{(\mathbf{1}_{c_i})} \right) \le a_i + 1$, and the same inequality also holds for y. Then,

$$\|\mathbf{h}(x)\|_{(\mathbf{1}_k)} \le \frac{k}{c_i} \|\mathbf{h}(x)\|_{(\mathbf{1}_{c_i})} \le \frac{k}{c_i} \left(v_i \left(1+\varepsilon/3\right)^{a_i+1} \right)$$

$$= \frac{k(1+\varepsilon/3)}{c_i}\left(v_i\,(1+\varepsilon/3)^{a_i+1}\right)$$
$$\le \frac{k(1+\varepsilon/3)}{c_i}\|\mathbf{h}(y)\|_{(\mathbf{1}_{c_i})} \le \frac{k(1+\varepsilon/3)}{c_i}\|\mathbf{h}(y)\|_{(\mathbf{1}_k)}.$$

Finally, $\frac{k}{c_i} \le \frac{c_{i+1}-1}{c_i} \le (1+\varepsilon/3)$, so that $\frac{\|\mathbf{h}(x)\|_{\mathbf{1}_k}}{\|\mathbf{h}(y)\|_{\mathbf{1}_k}} \le (1+\varepsilon/3)^2 = 1+\frac{2}{3}\varepsilon+\frac{1}{9}\varepsilon^2 \le 1+\varepsilon$ for all $\varepsilon \in (0,1]$. □

Next, we prove Lemma 4.3, which characterizes the dual norms of ordered norms. Given a vector $v \in \mathbb{R}^d$, we denote $\sigma v = (v_1, v_1+v_2, \ldots, v_1+\cdots+v_d)$ and $\Delta v = (v_1-v_2, v_2-v_3, \ldots, v_d)$ for brevity. Note that with this notation, we have the top-ℓ norm $\|v\|_{(\mathbf{1}_\ell)} = (\sigma|v|^{\downarrow})_\ell$. Further, we have $v^\top u = (\sigma v)^\top(\Delta u)$ for all vectors $u, v \in \mathbb{R}^d$.

Proof of Lemma 4.3. Let $K = \{x \in \mathbb{R}^d : \|x\|_{(w)} \le 1\}$ be the unit norm ball for $\|\cdot\|_{(w)}$, and let $K^* = \{y \in \mathbb{R}^d : y^\top x \le 1\ \forall\, x \in K\}$ be the unit norm ball of its dual norm. Also denote $\overline{K} = \left\{y \in \mathbb{R}^d : \max_{k\in[d]} \frac{\|y\|_{(\mathbf{1}_k)}}{\|w\|_{(\mathbf{1}_k)}} \le 1\right\}$. We will show that $\overline{K} = K^*$.

Suppose $y \in \overline{K}$. For any $x \in K$, we have

$$\begin{aligned}
y^\top x &\le (|y|^{\downarrow})^\top |x|^{\downarrow} && \text{(rearrangment inequality)}\\
&= (\sigma|y|^{\downarrow})^\top(\Delta|x|^{\downarrow}) && \text{(alternating sum)}\\
&\le (\sigma w)^\top(\Delta|x|^{\downarrow}) && (y \in \overline{K})\\
&= \|x\|_{(w)} && \text{(alternating sum)}\\
&\le 1. && (x \in K)
\end{aligned}$$

That is, $y \in K^*$. Conversely, assume $y \in K^*$ so that $y^\top x \le 1$ for each $x \in K$. Since K^* is symmetric, assume without loss of generality that $y_1 \ge \cdots \ge y_d \ge 0$, other cases are handled similarly. It is easy to check that for each $k \in [d]$, $x(k) := \frac{1}{(\sigma w)_k}(\underbrace{1,\ldots,1}_{k},0,\ldots,0)$ is in K. Therefore $1 \ge y^\top x(k) = \frac{(\sigma y)_k}{(\sigma w)_k} = \frac{(\sigma|y|^{\downarrow})_k}{(\sigma w)_k}$, implying that $y \in \overline{K}$. □

Next, we prove the Ordered Cauchy-Scwarz inequality (Lemma 4.4).

Proof of Lemma 4.4. This proof is similar to the proof of Lemma 4.3. For any $x, y \in \mathbb{R}^d$, we have

$$\begin{aligned} y^\top x &\le (|y|^\downarrow)^\top |x|^\downarrow && \text{(rearrangment inequality)} \\ &= \sum_{k\in[d]} (\sigma|y|^\downarrow)_k (\Delta|x|^\downarrow)_k && \text{(alternating sum)} \\ &\le \|y\|^*_{(w)} \sum_{k\in[d]} (\sigma w)_k (\Delta|x|^\downarrow)_k && \text{(definition of } \|y\|^*_{(w)}) \\ &= \|y\|^*_{(w)} \|x\|_{(w)} && \text{(alternating sum).} \end{aligned}$$

Further, the first inequality holds if and only if x, y are order-consistent, i.e., if and only if there exists an order π such that $x^\downarrow = x_\pi$ and $y^\downarrow = y_\pi$. The second inequality holds if and only if for each k, $(\sigma|y|^\downarrow)_k(\Delta|x|^\downarrow)_k = \|y\|^*_{(w)}(\sigma w)_k(\Delta|x|^\downarrow)_k$, which happens if and only if $(\Delta|x|^\downarrow)_k = 0$ or $\frac{(\sigma|y|^\downarrow)_k}{(\sigma w)_k} = \|y\|^*_{(w)}$. □

Lower bound on portfolio sizes. We prove the following theorem that lower bounds the portfolio sizes for ordered and symmetric monotonic norms:

Theorem 4.3. *There exist sets $\mathcal{D}$ and base functions $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}_{\ge 0}$ such that any $O(\log d)$-approximate portfolio for ordered norms must have size $d^{\Omega(1/\log\log d)}$. The same bound holds for symmetric monotonic norms.*

Since ordered norms are a subset of symmetric monotonic norms, it is sufficient to prove the result for ordered norms. First, we need a counting lemma:

Lemma B.1. *Given $L \ge 1$, Let T be the set of integral sequences $a = (a_0, \ldots, a_L)$ such that $a_{i-1} \le a_i \le a_{i-1} + 1$ for all $i \in [L]$ and $a_0 = 0$. Then there exists a subset $\overline{T} \subseteq T$ such that (1) $|\overline{T}| \ge 2^L/(2L^2)$, and (2) for any two sequences $a, a' \in \overline{T}$, there exists an i such that $a'_i < a_i$, and vice-versa.*

Proof. We first show that $|T| = 2^L$. For any such sequence a, consider $\phi(a) = (a_1 - a_0, \ldots, a_L - a_{L-1})$. Then $\phi(a)$ maps sequences in T to binary sequences $(b_1, \ldots, b_L)$;

further, ϕ is bijective. Therefore, $|T|$ is the number of binary sequences $(b_1, \ldots, b_L)$, which is 2^L.

Also note that $\geq$ is a partial order on T: $a \leq a'$ if and only if $a'_i \geq a_i$ for all $i \in [0, L]$. For any distinct a, a' such that $a' \geq a$, we must have that $\sum_{i\in[L]} a'_i \geq 1 + \sum_{i\in[L]} a_i$. Further, $\sum_{i\in[L]} a_i \leq L^2$ for all $a \in T$. Therefore, the length of any chain in order $\geq$ on T is at most $L^2 + 1$. This means that any chain decomposition of $\geq$ on T must have at least $|T|/(L^2 + 1) \geq 2^L/(2L^2)$ chains. By Dilworth's theorem [146], this is also the size of the largest anti-chain. But an anti-chain is exactly the set $\overline{T}$ we are looking for. $\square$

We are ready to prove Theorem 4.3:

Proof of Theorem 4.3. Throughout this proof, we will have $h_i(x) = x_i$, so that $\mathbf{h}(x) = x$, and minimizing $\|\mathbf{h}(x)\|$ is the same as minimizing $\|x\|$.

Let $S = \log^3 d$, and let L be such that $S^0 + S^1 + \cdots + S^L = d$. Then $L = \Theta\left(\frac{\log d}{\log S}\right) = \Theta\left(\frac{\log d}{\log\log d}\right)$, or that $S/L = \Omega(\log^2 d)$.

Let $\overline{T}$ be the set of integral sequences from the previous lemma, i.e., each sequence $a = (a_0, \ldots, a_L)$ is such that $a_{i-1} \leq a_i \leq a_{i-1} + 1$ for all $i \in [L]$ and $a_0 = 0$, and for any two sequences $a, a' \in \overline{T}$, there exists i such that $a'_i < a_i$. Define

$$x(a) = \Big(\underbrace{S^{-a_0}}_{S^0}, \underbrace{S^{-a_1}, \ldots, S^{-a_1}}_{S^1}, \ldots, \underbrace{S^{-a_L}, \ldots, S^{-a_L}}_{S^L} \Big).$$

Note that since $a_i \geq a_{i-1}$, $x^{\downarrow} = x$. Further, since $a_i \leq a_{i-1} + 1$, we have $a_i - i \leq a_{i-1} - (i-1)$. Define

$$w(a) = \Big(\underbrace{S^{a_0-0}}_{S^0}, \underbrace{S^{a_1-1}, \ldots, S^{a_1-1}}_{S^1}, \ldots, \underbrace{S^{a_L-L}, \ldots, S^{a_L-L}}_{S^L} \Big).$$

Then

$$\|x(a)\|_{(w(a))} = x(a)^\top w(a) = \sum_{i\in[0,L]} S^{-a_i} S^{a_i - i} S^i = L.$$

Further, for any other $a' = (a'_0, \ldots, a'_L) \in \overline{T}$, there exists i such that $a'_i < a_i$, we get

$$\|x(a')\|_{(w(a))} \geq S^{-a'_i} S^{a_i - i} S^i > S.$$

Since $S/L = \Omega(\log^2 d)$, this means that $x(a')$ is an $\omega(\log d)$-approximation for $\|\cdot\|_{(w(a))}$. That is, any $O(\log d)$-approximate portfolio for $\overline{T}$ for ordered norms must have size $|\overline{T}| \geq 2^L/(2L^2)$. However,

$$\frac{2^L}{2L^2} = 2^{\Theta((\log d)/\log\log d)} \Theta\left(\frac{(\log\log d)^2}{(\log d)^2}\right) = d^{\Theta\left(\frac{1}{\log\log d}\right) - O\left(\frac{\log\log d}{\log d}\right)} = d^{\Omega\left(\frac{1}{\log\log d}\right)}.$$

To prove the second part of the theorem, we claim that in fact even for $\operatorname{conv}(\overline{T})$, we have any $O(\log d)$ portfolio for ordered norms must have size $\geq |\overline{T}| = d^{\Omega(1/\log\log d)}$. Let $x = \sum_{b \in \overline{T}} \lambda_b x(b) \in \operatorname{conv}(\overline{T})$. Fix $a \in \overline{T}$. We will show that for all x such that $1 - \lambda_a > 1/4$, $\|x\|_{(w(a))} = \Theta(S/L)\|x(a)\|_{(w(a))}$. That is, the any $O(\log d)$-approximate minimizer x of $\|\cdot\|_{(w(a))}$ in $\operatorname{conv}(\overline{T})$ must have $\lambda_a \geq \frac{3}{4}$, implying the claim.

Note that for each b, $x(b)^{\downarrow} = x(b)$. Therefore,

$$\begin{aligned}
\|x\|_{w(a)} &= \left(\sum_{b \in \overline{T}} \lambda_b x(b)\right)^{\top} w(a) = \sum_{b \in \overline{T}} \lambda_b \|x(b)\|_{(w(a))} \\
&= \lambda_a \|x(a)\|_{(w(a))} + \sum_{b \neq a} \lambda_b \|x(b)\|_{(w(a))} \\
&\geq \lambda_a L + S \sum_{b \neq a} \lambda_b \geq S(1 - \lambda_a) \geq S/4.
\end{aligned}$$

Where the last inequality follows from the assumption that $1 - \lambda_a \geq 1/4$. Therefore, $\|x\|_{(w(a))} = \Theta(S/L) = \omega(\log d)$. This finishes the proof. □

B.2 Omitted proofs from Section 4.5

Proof of Lemma 4.12. For each $i \in [r], j \in [d]$, by construction we have $\widetilde{A}_{i,j} \leq A_{i,j}$, so that if $x \in \mathcal{P}$, then $Ax \geq \widetilde{A}x \geq \mathbf{1}_r$, i.e., $\widetilde{\mathcal{P}} \subseteq \mathcal{P}$.

We claim that for all $x \in \mathcal{P}$ there is some $\widetilde{x} \in \widetilde{\mathcal{P}}$ such that $\widetilde{x} \preceq (1+\varepsilon)x$. From Lemma 2.1, this claim implies that $\|\widetilde{x}\| \leq (1+\varepsilon)\|x\|$ for all symmetric monotonic norms $\|\cdot\|$, and therefore that $\min_{\widetilde{x}\in\widetilde{\mathcal{P}}} \|\widetilde{x}\| \leq (1+\varepsilon)\min_{x\in\mathcal{P}} \|x\|$. This implies the lemma.

Define $\widetilde{x} = \left(1+\frac{\varepsilon}{2}\right)\left(x + \frac{\varepsilon\|x\|_1}{3d}(1,\ldots,1)\right)$. We have for all $k \in [d]$

$$\|\widetilde{x}\|_{(\mathbf{1}_k)} = \left(1+\frac{\varepsilon}{2}\right)\left(\|x\|_{(\mathbf{1}_k)} + \frac{k\varepsilon\|x\|_1}{3d}\right).$$

However, $\frac{\|x\|_1}{d} \leq \frac{\|x\|_{(\mathbf{1}_k)}}{k}$, so that the above gives us

$$\|\widetilde{x}\|_{(\mathbf{1}_k)} \leq \left(1+\frac{\varepsilon}{2}\right)\left(\|x\|_{(\mathbf{1}_k)} + \frac{\varepsilon\|x\|_{(\mathbf{1}_k)}}{3}\right) = \left(1+\frac{\varepsilon}{2}\right)\left(1+\frac{\varepsilon}{3}\right)\|x\|_{(\mathbf{1}_k)}.$$

For all $\varepsilon \in (0,1]$, $\left(1+\frac{\varepsilon}{2}\right)\left(1+\frac{\varepsilon}{3}\right) \leq 1+\varepsilon$, so that $\widetilde{x} \preceq (1+\varepsilon)x$. Next, we show that $\widetilde{x} \in \widetilde{\mathcal{P}}$. Clearly, $\widetilde{x} \geq x \geq 0$; it remains to show that $\widetilde{A}\widetilde{x} \geq \mathbf{1}_r$.

Fix $i \in [r]$; denote the ith rows of $A, \widetilde{A}$ respectively by $A_i, \widetilde{A}_i$. From the algorithm, for $j \notin B(i)$, we have $\widetilde{A}_{i,j} \geq \frac{1}{1+\frac{\varepsilon}{2}} A_{i,j}$. Therefore,

$$\begin{aligned}
\widetilde{A}_i^\top \widetilde{x} &= \sum_{j\in[d]} \widetilde{A}_{i,j}\widetilde{x}_j = \sum_{j\notin B(i)} \widetilde{A}_{i,j}\widetilde{x}_j && (\widetilde{A}_{i,j} = 0 \,\forall\, j \in B(i)),\\
&\geq \frac{1}{1+\frac{\varepsilon}{2}} \sum_{j\notin B(i)} A_{i,j}\widetilde{x}_j \\
&= \frac{1}{1+\frac{\varepsilon}{2}} \left(\sum_{j\notin B(i)} A_{i,j}\left(1+\frac{\varepsilon}{2}\right)\left(x_j + \frac{\varepsilon\|x\|_1}{3d}\right)\right) \\
&= \sum_{j\notin B(i)} A_{i,j}x_j + \frac{\varepsilon\|x\|_1}{3d}\sum_{j\notin B(i)} A_{i,j}.
\end{aligned}$$

Now, $\sum_{j\notin B(i)} A_{i,j} \geq a_i^* \geq \frac{\mu}{d}\sum_{j\in B(i)} A_{i,j} = \frac{3d}{\varepsilon}\sum_{j\in B(i)} A_{i,j}$. Therefore,

$$\frac{\varepsilon\|x\|_1}{3d}\sum_{j\notin B(i)} A_{i,j} \geq \frac{\varepsilon\|x\|_1}{3d}\cdot\frac{3d}{\varepsilon}\cdot\sum_{j\in B(i)} A_{i,j} \geq \sum_{j\in B(i)} A_{i,j}x_j.$$

Together, this means that $\widetilde{A}_i^\top\widetilde{x} \geq A_i^\top x \geq 1$. Since this holds for all $i \in [r]$, $\widetilde{x} \in \mathcal{P}$. $\square$

B.2.1 Proof of Lemma 4.16

We restate the relevant convex programs and the lemma here for convenience:

$$\min \|x\|_{(w)} \quad \text{s.t.} \quad Ax \geq \mathbf{1}_r, x \in \mathcal{Q}. \tag{primal'}$$

$$\min \|A^\top \lambda\|^*_{(w)} \quad \text{s.t.} \quad \lambda \in \mathbf{\Delta}_r. \tag{dual}$$

Lemma 4.16. *Given a weight vector w, $\|x(w)\|_{(w)}\|A^\top\lambda(w)\|^*_{(w)} = 1$. Further, there is a reduced order ρ such that both $x(w), A^\top\lambda(w)$ satisfy ρ.*

For $j \in [d]$, denote the jth column of A as $A^{(j)}$. $A^{(j)}$ is an r-dimensional vector. Recall that $S_1, \ldots, S_{N^r}$ form a partition of $[d]$ such that for $l \in [N^r]$, and for all $j, j' \in S_l$, $A^{(j)} = A^{(j')}$. Also recall that $\mathcal{Q}$ is the set of all vectors $x \geq 0$ such that $x_j = x_{j'}$ for all $j, j' \in S_l$, for all $l \in [N^r]$. From Lemma 4.13, $x(w) \in \mathcal{Q}$.

First, for all $x \in \mathcal{P}$ and $\lambda \in \mathbf{\Delta}_r$, we get by Ordered Cauchy-Schwarz (Lemma 4.4) that $\|x\|_{(w)}\|A^\top\lambda\|^*_{(w)} \geq \lambda^\top Aw$. Since $x \in \mathcal{P}$, $Ax \geq \mathbf{1}_r$, and since $\lambda \in \mathbf{\Delta}_r$, $\lambda^\top Ax \geq 1$. Now, suppose that there is some $\lambda \in \mathbf{\Delta}_r$ such that $\|x(w)\|_{(w)}\|A^\top\lambda\|^*_{(w)} = 1$, i.e. equality holds. Then, since $\lambda(w) = \arg\min_{\lambda \in \mathbf{\Delta}_r} \|\lambda\|^*_{(w)}$, we get that

$$1 = \|x(w)\|_{(w)}\|A^\top\lambda\|^*_{(w)} \geq \|x(w)\|_{(w)}\|A^\top\lambda(w)\|^*_{(w)} \geq 1.$$

Then equality must hold everywhere, and in particular $\|x(w)\|_{(w)}\|A^\top\lambda(w)\|^*_{(w)} = 1$. Further, from Ordered Cauchy-Schwarz, it is necessary that $x(w), A^\top\lambda(w)$ satisfy some order $\pi \in \text{Perm}(d)$.

From Lemma 4.13, $x(w) \in \mathcal{Q}$, i.e., for all $j, j' \in S_l$, for all $l \in [N^r]$, $x(w)_j = x(w)_{j'}$. Similarly, $(A^\top\lambda(w))_j$ is the dot product of the jth column of A with $\lambda(w)$, and therefore $A^\top\lambda(w) \in \mathcal{Q}$ as well. Since $x, A^\top\lambda(w)$ both satisfy order π, π must induce a reduced order ρ on $S_1, \ldots, S_{N^r}$. This implies the lemma.

It remains to prove that there exists λ such that $\|x(w)\|_{(w)}\|A^\top\lambda\|^*_{(w)} = 1$. Our proof

is along the lines of the proof of strong duality using Slater's conditions [147], although we use the properties of ordered norms at several places. We will need the following two lemmas:

Lemma B.2. *For vector $y \in \mathbb{R}^d$ such that $y_1 \geq \cdots \geq y_d \geq 0$, let $t_1 \leq t_2 \leq \cdots \leq t_T = d$ be indices such that*

$$y_1 = \cdots = y_{t_1} \geq y_{t_1+1} = \cdots = y_{t_2} \geq \cdots \geq y_{t_{T-1}+1} = \cdots = y_{t_T}.$$

Then for any weight vector w, $\|y\|^*_{(w)} = \max_{k \in [d]} \frac{\|y\|_{(\mathbf{1}_k)}}{\|w\|_{(\mathbf{1}_k)}}$ is achieved at some $k \in \{t_1, \ldots, t_T\}$.

Proof. It is sufficient to show that for all $i \in [T]$ and $t_{i-1} \leq k \leq t_i$, we have

$$\max \left\{ \frac{\|y\|_{(\mathbf{1}_{t_{i-1}})}}{\|w\|_{(\mathbf{1}_{t_{i-1}})}}, \frac{\|y\|_{(\mathbf{1}_{t_i})}}{\|w\|_{(\mathbf{1}_{t_i})}} \right\} \geq \frac{\|y\|_{(\mathbf{1}_k)}}{\|w\|_{(\mathbf{1}_k)}}.$$

Denote $z = y_{t_{i-1}+1} = \cdots = y_{t_i}$. Consider $(1-\lambda)\|y\|_{(\mathbf{1}_{t_{i-1}})} + \lambda\|y\|_{(\mathbf{1}_{t_i})}$ for $\lambda = \frac{k-t_{i-1}}{t_i - t_{i-1}}$. Then $\lambda \in [0,1]$, and

$$(1-\lambda)\|y\|_{(\mathbf{1}_{t_{i-1}})} + \lambda\|y\|_{(\mathbf{1}_{t_i})} = \|y\|_{(\mathbf{1}_{t_{i-1}})} + \lambda z(t_i - t_{i-1}) = \|y\|_{(\mathbf{1}_{t_{i-1}})} + (k - t_{i-1})z = \|y\|_{(\mathbf{1}_k)}.$$

Further,

$$\begin{aligned}(1-\lambda)\|w\|_{(\mathbf{1}_{t_{i-1}})} + \lambda\|w\|_{(\mathbf{1}_{t_i})} &= \|w\|_{(\mathbf{1}_{t_{i-1}})} + \lambda(w_{t_{i-1}+1} + \cdots + w_{t_i}) \\ &= \|w\|_{(\mathbf{1}_{t_{i-1}})} + (k - t_{i-1})\frac{w_{t_{i-1}+1} + \cdots + w_{t_i}}{t_i - t_{i-1}}.\end{aligned}$$

Since $w_{t_{i-1}+1} \geq \cdots \geq w_{t_i}$, we get that

$$\frac{w_{t_{i-1}+1} + \cdots + w_{t_i}}{t_i - t_{i-1}} \leq \frac{w_{t_{i-1}+1} + \cdots + w_k}{k - t_{i-1}}.$$

Plugging this back in, we get $(1-\lambda)\|w\|_{(\mathbf{1}_{t_{i-1}})} + \lambda\|y\|_{(\mathbf{1}_{t_i})} \leq \|w\|_{(\mathbf{1}_k)}$. Therefore,

$$\frac{\|y\|_{(\mathbf{1}_k)}}{\|w\|_{(\mathbf{1}_k)}} \leq \frac{(1-\lambda)\|y\|_{(\mathbf{1}_{t_{i-1}})} + \lambda\|y\|_{(\mathbf{1}_{t_i})}}{(1-\lambda)\|w\|_{(\mathbf{1}_{t_{i-1}})} + \lambda\|w\|_{(\mathbf{1}_{t_i})}} \leq \max\left\{\frac{\|y\|_{(\mathbf{1}_{t_{i-1}})}}{\|w\|_{(\mathbf{1}_{t_{i-1}})}}, \frac{\|y\|_{(\mathbf{1}_{t_i})}}{\|w\|_{(\mathbf{1}_{t_i})}}\right\}.$$

□

Lemma B.3. *For $\mu \in \mathbb{R}^r_{\geq 0}$,*

$$\sup_{x\in\mathcal{Q}} \mu^\top Ax - \|x\|_{(w)} = \begin{cases} 0 & \text{if } \|\mu^\top A\|^*_{(w)} \leq 1, \\ \infty & \text{otherwise}. \end{cases}$$

Proof. Denote $y = A^\top\mu$. Then $y \in \mathbb{R}^d$, and $y_j = (A^{(j)})^\top\mu$. If $\|y\|^*_{(w)} \leq 1$, we get from Ordered Cauchy-Schwarz (Lemma 4.4) that

$$y^\top x - \|x\|_{(w)} \leq \|y\|^*_{(w)}\|x\|_{(w)} - \|x\|_{(w)} \leq (\|y\|^*_{(w)} - 1)\|x\|_{(w)} \leq 0.$$

However, $0 \in \mathcal{Q}$. Therefore $y^\top x - \|x\|_{(w)} = 0$ when $x = 0$, so that

$$\sup_{x\in\mathcal{Q}} y^\top x - \|x\|_{(w)} = 0.$$

Now suppose that $\|y\|^*_{(w)} \geq 1$. Note that since $y_j = (A^{(j)})^\top\mu$, for all $j, j' \in S_l$ for some l, we get $y_j = y_{j'}$.

Relabel the indices $[N^r]$ so that for all $j \in S_l$ and $j' \in S_{l+1}$, $y_j \geq y_{j'}$. Further, relabel indices $[d]$ so that $S_1 = \{1, \ldots, |S_1|\}$, $S_2 = \{|S_1|+1, \ldots, |S_1|+|S_2|\}$ etc, i.e.,

$$y_1 = \cdots = y_{|S_1|} \geq y_{|S_1|+1} = \cdots = y_{|S_1|+|S_2|} \geq \cdots \geq y_{d-|S_{N^r}|+1} = \cdots = y_d \geq 0.$$

By the previous lemma $\|y\|^*_{(w)} = \max_{k\in[d]} \frac{\|y\|_{(\mathbf{1}_k)}}{\|w\|_{(\mathbf{1}_k)}}$ achieved at some $k = |S_1| + \cdots + |S_l|$. For brevity, denote this number as k^*.

Define x such that $x_1 = x_2 = \cdots = x_{k^*} = \frac{\alpha}{\|w\|_{(\mathbf{1}_{k^*})}}$ and $x_{k^*+1} = \cdots = x_d = 0$ where α is an arbitrarily large number. Then $x \in \mathcal{Q}$ and $\|x\|_{(w)} = \alpha$. Further,

$$y^\top x = \|y\|_{(\mathbf{1}_{k^*})} \frac{\alpha}{\|w\|_{(\mathbf{1}_{k^*})}}.$$

Since $\frac{\|y\|_{(\mathbf{1}_{k^*})}}{\|w\|_{(\mathbf{1}_{k^*})}} = \|y\|^*_{(w)} > 1$, we get that $y^\top x - \|x\|_{(w)} = \alpha \left(\frac{\|y\|_{(\mathbf{1}_{k^*})}}{\|w\|_{(\mathbf{1}_{k^*})}} - 1 \right)$, which can be arbitrarily large as α grows. This proves the second case as well. □

We proceed to prove that there exists λ such that $\|x(w)\|_{(w)} \|A^\top \lambda\|^*_{(w)} = 1$. Let $\mathcal{A}$ be the set of points $(v_1, \ldots, v_r, t)$ such that there exists an $x \in \mathcal{Q}$ with $v_i \geq 1 - A_i^\top x$ for all $i \in [r]$ and $t \geq \|x\|_{(w)}$. It is easy to check that $\mathcal{A}$ is convex. Next, define $\mathcal{B} = \{(\underbrace{0, \ldots, 0}_{r}, s) : s < \|x(w)\|_{(w)}\}$. Clearly, $\mathcal{B}$ is convex. It is easy to see that $\mathcal{A} \cap \mathcal{B} = \emptyset$. Therefore, there is a separating hyperplane between $\mathcal{A}, \mathcal{B}$, i.e. there exist $\mu \in \mathbb{R}^d, \delta, \alpha \in \mathbb{R}$ such that

$$\mu^\top v + \delta t \geq \alpha \, \forall \, (v, t) \in \mathcal{A}, \tag{B.1}$$

$$\delta s < \alpha \, \forall s < \|x(w)\|_{(w)}. \tag{B.2}$$

The second equation implies that $\delta \geq 0$ since otherwise we can choose s to be arbitrarily small and δs becomes arbitrarily large. Then, we get $\delta \|x(w)\|_{(w)} \leq \alpha$.

Further, by a similar argument, $\mu \geq 0$. Applying Equation B.1 to point $(1 - A_1^\top x, \ldots, 1 - A_r^\top x, \|x\|_{(w)}) \in \mathcal{A}$ we get that for all $x \in \mathcal{Q}$,

$$\sum_{i \in [r]} \mu_i - \mu^\top A x + \delta \|x\|_{(w)} \geq \alpha \geq \delta \|x(w)\|_{(w)}.$$

Case I: $\mu = 0$. Then $\delta \|x\|_{(w)} \geq \alpha \geq \delta \|x(w)\|_{(w)}$. Since not both μ, δ can be zero, $\delta > 0$. Further, $\|x(w)\|_{(w)} > 0$, so if we pick $x = 0 \in \mathcal{Q}$, we get a contradiction.

Case II: $\mu \neq 0$, so we get that all for all $x \in \mathcal{Q}$, $\sum_{i \in [r]} \mu_i - \mu^\top A x + \delta \|x\|_{(w)} \geq \alpha \geq \delta \|x(w)\|_{(w)}$. If $\delta = 0$, then $\sum_i \mu_i - \mu^\top A x \geq 0$ for all $x \in \mathcal{Q}$. Pick arbitrarily large x again,

giving a contradiction. Therefore, $\delta > 0$; assume without loss of generality that it is 1.

That is, for all $x \in \mathcal{Q}$, $\sum_i \mu_i - \mu^\top A x + \|x\|_{(w)} \geq \|x(w)\|_{(w)}$. Taking infimum on the left-hand side and applying Lemma B.3, we get that $\sum_i \mu_i \geq \|x(w)\|_{(w)}$ with $\|\mu^\top A\|^*_{(w)} \leq 1$. Then $\lambda := \frac{\mu}{\sum_i \mu_i} \in \mathbf{\Delta}_r$. Therefore,

$$1 \geq \|\mu^\top A\|^*_{(w)} = \sum_i \mu_i \|\lambda^\top A\|^*_{(w)} \geq \|x(w)\|_{(w)} \|\lambda^\top A\|^*_{(w)}. \quad \square$$

APPENDIX C

OMITTED PROOFS FROM Chapter 6

We include omitted proofs and various lemmas for Chapter 6 here.

C.1 Proof of Theorem 6.1

The following lemma follows from the monotonicity of p-mean functions (Lemma 6.1):

Lemma C.1. $f(p) := \max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]$ *is monotone increasing in* p*.*

Proof. Denote $x^{(p)} = \arg\max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]$. Then, by Lemma 6.1,

$$f(p) = \mathbb{E}[M_p(\mathbf{h}(x^{(p)}))] \leq \mathbb{E}[M_q(\mathbf{h}(x^{(p)}))] \leq f(q). \qquad \square$$

The following two lemmas show that the p-mean for $p = -\infty$ is α-approximated by the p_0-mean where $p_0 = -\frac{\ln d}{\ln(1/\alpha)}$, so we can effectively restrict to $[-p_0, 1]$ when finding portfolios:

Lemma C.2. *Given a vector* $z \in \mathbb{R}^d_{>0}$ *and* $\alpha \in (0,1)$*, define* $p_0 = -\frac{\ln d}{\ln(1/\alpha)}$*. Then,*

$$M_{-\infty}(z) \geq \alpha \cdot M_p(z) \quad \forall\, p \leq p_0.$$

Proof. Suppose $0 < z_1 \leq \ldots \leq z_d$, so that $M_{-\infty}(x) = \min_{j \in [d]} z_j = z_1$. Given $p \leq p_0$, denote $q = -p \geq \frac{\ln d}{\ln(1/\alpha)}$. Then, since

$$M_p(z) = \left(\frac{1}{d} \sum_{j \in [d]} z_j^p \right)^{1/p} = \frac{1}{\left(\frac{1}{d} \sum_{j \in [d]} \frac{1}{z_j^q} \right)^{1/q}} = \frac{1}{\frac{1}{z_1} \left(\frac{1}{d} \sum_{j \in [d]} \left(\frac{z_1}{z_j} \right)^q \right)^{1/q}},$$

we get

$$\frac{1}{M_p(z)} \geq \frac{1}{z_1}\left(\frac{1}{d} \times \left(\frac{z_1}{z_1}\right)^q\right)^{1/q} = \frac{d^{1/p}}{z_1} \geq \frac{d^{1/p_0}}{z_1} = \frac{\alpha}{M_{-\infty}(z)},$$

implying the result. □

Lemma C.3. *Given feasible set $\mathcal{D}$ with random base objectives $h_1, \ldots, h_d : \mathcal{D} \to \mathbb{R}$ and an approximation factor $\alpha \in (0,1)$, denote $p_0 = -\frac{\ln d}{\ln(1/\alpha)}$. Then, $x^{(0)} = \arg\max_{x\in\mathcal{D}} M_{p_0}(\mathbf{h}(x))$ is an α-approximation for all $p \leq p_0$. That is,*

$$\mathbb{E}[M_p(\mathbf{h}(x^{(0)}))] \geq \max_{x\in\mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))] \quad \forall\, p \leq p_0.$$

Proof. From Lemma C.2, for all $p \leq p_0$, we get

$$\begin{aligned}
\mathbb{E}[M_p(\mathbf{h}(x^{(0)}))] &\geq \mathbb{E}[M_{-\infty}(\mathbf{h}(x^{(0)}))] && (Lemma\ C.1)\\
&\geq \alpha \cdot \mathbb{E}[M_{p_0}(\mathbf{h}(x^{(0)}))] && (Lemma\ C.2)\\
&= \alpha \cdot \max_{x\in\mathcal{D}} \cdot \mathbb{E}[M_{p_0}(\mathbf{h}(x))]\\
&\geq \alpha \cdot \max_{x\in\mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]. && (Lemma\ C.1)
\end{aligned}$$

□

The following three lemmas establish guarantees on `LineSearch`. Denote $\mathrm{OPT}_p = \max_{x\in\mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))]$ as the maximum value of the p-mean function over $\mathcal{D}$.

Lemma C.4. *Suppose* `LineSearch` *(Algorithm 8) on input p, α returns $b^* \in [p, 1]$. Then, either $b^* = 1$, or the policy $x := \arg\max_{x'\in\mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x'))]$ satisfies*

$$\frac{\mathrm{OPT}_p}{\sqrt{\alpha}} \leq \mathrm{OPT}_{b^*} \leq \frac{\mathbb{E}[M_{b^*}(\mathbf{h}(x))]}{\alpha}.$$

Proof. If `LineSearch` on input p, α does not return $b^* = 1$, then we must have that

p-MeanPortfolio enters the while loop (step 3) at least once. In particular,

$$\mathrm{OPT}_p = \mathbb{E}[M_p(\mathbf{h}(x))] < \alpha\ \mathrm{OPT}_{b^*}) < \sqrt{\alpha}\ \mathrm{OPT}_{b^*}.$$

This proves the first inequality. For the second inequality, notice that the algorithm terminates only when $\mathbb{E}[M_a(\mathbf{h}(x))] \geq \alpha\ \mathrm{OPT}_{b^*}$. As can be checked, the algorithm maintains the invariant $a \geq p$. Therefore, by Lemma 6.1,

$$\mathbb{E}[M_{b^*}(\mathbf{h}(x))] \geq \mathbb{E}[M_a(\mathbf{h}(x))] \geq \alpha\ \mathrm{OPT}_{b^*}. \qquad \square$$

Lemma C.5. *Algorithm* textttLineSearch *maintains the following invariant at all times:* $v\mathbb{E}[M_a(\mathbf{h}(x))] \geq \sqrt{\alpha}\ \mathrm{OPT}_a$.

Proof. Initially, $p = a$, and therefore $x := \arg\max_{x' \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x')]$ satisfies $\mathbb{E}[M_p(\mathbf{h}(x)] = \mathrm{OPT}_p \geq \sqrt{\alpha}\ \mathrm{OPT}_p$ since $\alpha \in (0, 1)$.

Further, the algorithm only updates $a \leftarrow q$ in step 6 when $\mathbb{E}[M_a(\mathbf{h}(x)] \geq \sqrt{\alpha}\, \mathbb{E}[M_q(\mathbf{h}(x)]$. Since $q \geq a$, we get from Lemma 6.1 that $\mathbb{E}[M_q(\mathbf{h}(x)] \geq \mathbb{E}[M_p(\mathbf{h}(x)]$, thus finishing the proof. $\square$

Lemma C.6. *Algorithm* LineSearch *on input* p, α *returns* $b^* > p$ *such that* $x := \arg\max_{x' \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x')]$ *is an* α*-approximation for all* $q \in [p, b^*]$*, i.e.,* $\mathbb{E}[M_q(\mathbf{h}(x)] \geq \alpha\ \mathrm{OPT}_q$ *for all such* q.

Proof. LineSearch starts with $a \leftarrow p$ and keeps increasing $a \leftarrow q$ whenever $\mathbb{E}[M_a(\mathbf{h}(x)] \geq \sqrt{\alpha}\ \mathrm{OPT}_q$ for $q = \frac{a+b}{2}$. In particular, whenever a is increased, we get that for all $q' \in [a, (a+b)/2]$, from Lemma 6.1,

$$\mathbb{E}[M_{q'}(\mathbf{h}(x)] \geq \mathbb{E}[M_a(\mathbf{h}(x)] \geq \sqrt{\alpha}\ \mathrm{OPT}_{(a+b)/2} \geq \sqrt{\alpha}\ \mathrm{OPT}_{q'} > \alpha\ \mathrm{OPT}_{q'}.$$

That is, at all times, the algorithm maintains the invariant that π is an α-approximation for all $q' \in [p, a]$. Further, the algorithm terminates at $b = b^*$ when $\mathbb{E}[M_a(\mathbf{h}(x)] \geq \mathrm{OPT}_{b^*}$,

i.e., for all $q' \in [a, b^*]$, we get

$$\mathbb{E}[M_{q'}(\mathbf{h}(x)] \geq \mathbb{E}[M_a(\mathbf{h}(x)] \geq \alpha \operatorname{OPT}_{b^*} \geq \alpha \operatorname{OPT}_{q'}. \qquad \square$$

The next lemma bounds the slope of the logarithm of the p-mean function and consequently the slope of OPT_p:

Lemma C.7. *1. For any $z \in \mathbb{R}^d_{>0}$, such that $L \leq z_i \leq U$ for all $i \in [d]$, define $g(z,p) := \ln M_p(z) = \frac{1}{p} \ln \left(\frac{1}{d} \sum_{i \in [d]} z_i^p \right)$. Then, we have for all $p \leq 1$ that*

$$\frac{\mathrm{d}g(z,p)}{\mathrm{d}p} \leq \kappa \ln \kappa,$$

where $\kappa := \frac{U}{L}$.

2. *For any given $x \in \mathcal{D}$ and base functions $h_1, \ldots, h_d$ with condition number κ, we have $\frac{\mathrm{d}(\ln \mathbb{E}[M_p(\mathbf{h}(x)])}{\mathrm{d}p} \leq \kappa \ln \kappa$, where κ is the condition number defined in Assumption 6.1.*

3. *For all distinct $p, q \leq 1$,*

$$\frac{\ln \operatorname{OPT}_q - \ln \operatorname{OPT}_p}{q - p} \leq \kappa \ln \kappa.$$

Proof. **Part 1.** Note that $g(\beta z, p) = \ln \beta + g(z,p)$ for all $z \in \mathbb{R}^d_{>0}$, $p \leq 1$, and $\beta > 0$. Therefore, we can assume without loss of generality that $L = 1$ and $U = \frac{U}{L} = \kappa$.

That is, assume without loss of generality that $1 \leq z_1 \leq \ldots \leq z_d \leq \kappa$. Since $g(z,p) = \frac{-\ln d}{p} + \frac{1}{p} \ln \left(\sum_{i \in [d]} z_i^p \right)$, we get that $pg(z,p) = -\ln d + \ln \left(\sum_{i \in [d]} z_i^p \right)$. Therefore,

$$g(z,p) + p \frac{\mathrm{d}g(z,p)}{\mathrm{d}p} = \frac{\sum_i (\ln z_i) \cdot z_i^p}{\sum_i z_i^p}. \tag{C.1}$$

Since $\ln t$ is concave in t, we get that

$$pg(z,p) = \ln\left(\frac{1}{d}\sum_i z_i^p\right) \geq \frac{1}{d}\sum_i \ln z_i^p = \frac{p}{d}\sum_i \ln z_i.$$

Therefore, we get

$$g(z,p)\begin{cases} \geq \frac{1}{d}\sum_i \ln z_i & \text{if } p \geq 0, \\ \leq \frac{1}{d}\sum_i \ln z_i & \text{if } p < 0. \end{cases} \tag{C.2}$$

Denote $\mu_p = \frac{1}{d}\sum_{i\in[d]} z_i^p$.

Case A: $p \in [0,1]$**.** Plugging this back into Equation C.1, we get

$$p\frac{dg(z,p)}{\mathrm{d}p} \leq \frac{\sum_i (\ln z_i)\cdot z_i^p}{\mu_p \times d} - \frac{1}{d}\sum_i \ln z_i = \frac{1}{d}\left(\sum_i \ln z_i\left(\frac{z_i^p}{\mu_p} - 1\right)\right).$$

Since $1 \leq z_i \leq \kappa$ for all i, we have $0 \leq \ln z_i \leq \ln \kappa$ for all i. Further, since $z_d \geq z_i$, we get that

$$p\,\frac{\mathrm{d}g(z,p)}{\mathrm{d}p} \leq \sum_i \ln z_i\left(\frac{z_i^p}{\mu_p} - 1\right) \leq \ln\kappa \sum_{i: z_i^p \geq \mu_p}\left(\frac{z_d^p}{\mu_p} - 1\right) \leq d\ln\kappa\left(\frac{z_d^p}{\mu_p} - 1\right). \tag{C.3}$$

Now, since μ_p is the mean of $z_i^p, i \in [d]$, we must have that $\mu_p = \theta^p$ for some $1 \leq \theta \leq \kappa$. Substituting this, we get $d \cdot p\,\frac{\mathrm{d}g(z,p)}{\mathrm{d}p} \leq d\cdot(\ln\kappa)\left((z_d/\theta)^p - 1\right)$. Since $\frac{z_d}{\theta} \leq \frac{\kappa}{1} = \kappa$ and the derivative of α^p with respect to α is $p\alpha^{p-1}$, we get

$$\begin{aligned}
((z_d/\theta)^p - 1) &\leq (\kappa^p - 1) \\
&= \int_1^\kappa p\cdot\alpha^{p-1}\,\mathrm{d}\alpha \\
&\leq \int_1^\kappa p\cdot 1^{p-1}\,\mathrm{d}\alpha \quad (\text{since } p-1 \leq 0 \text{ and so } \alpha^{p-1} \text{ is nonincreasing in } p) \\
&\leq p(\kappa - 1) \leq p\kappa. \qquad (\text{since } p \geq 0)
\end{aligned}$$

Plugging this back into Equation C.3, we get

$$p\frac{\mathrm{d}g(z,p)}{\mathrm{d}p} \le p\kappa \ln \kappa,$$

or that $\frac{\mathrm{d}g(z,p)}{\mathrm{d}p} \le \kappa \ln \kappa$ for all $p \in [0,1]$.

Case B: $p < 0$**.** As before, plugging Equation C.2 into Equation C.1, we get

$$\begin{aligned} p\frac{\mathrm{d}g(z,p)}{\mathrm{d}p} &\ge \frac{\sum_i (\ln z_i) \cdot z_i^p}{d \times \mu_p} - \frac{1}{d}\sum_i \ln z_i \\ &= \frac{1}{d}\left(\sum_i \ln z_i \left(\frac{z_i^p}{\mu_p} - 1\right)\right) = -\frac{1}{d}\left(\sum_i \ln z_i \left(1 - \frac{z_i^p}{\mu_p}\right)\right). \end{aligned}$$

That is, since $p < 0$,

$$\begin{aligned} \frac{\mathrm{d}g(z,p)}{\mathrm{d}p} &\le \frac{1}{(-p) \times d}\left(\sum_i \ln z_i \left(1 - \frac{z_i^p}{\mu_p}\right)\right) \le \frac{\ln \kappa}{(-p)d} \sum_{i: z_i^p \le \mu_p} \left(1 - \frac{z_i^p}{\mu_p}\right) \\ &\le \frac{\ln \kappa}{(-p)d} \times d\left(1 - \frac{z_d^p}{\mu_p}\right) = \frac{\ln \kappa}{(-p)}\left(1 - \frac{z_d^p}{\mu_p}\right). \end{aligned}$$

Denote $-p = q$; then $q > 0$. As before, $\mu_p = \theta^p$ for some $1 \le \theta \le d$, and so we have

$$\frac{\mathrm{d}g(z,p)}{\mathrm{d}p} \le \frac{\ln \kappa}{q}\left(1 - \frac{\theta^q}{z_d^q}\right) \le \frac{\ln \kappa}{q}\left(1 - \frac{1}{\kappa^q}\right).$$

For $q \ge 1$ (i.e., for $p \le -1$), this is at most $\ln \kappa$. For $q \in [0,1]$, we will bound this like Case A:

$$\begin{aligned} \frac{\mathrm{d}g(z,p)}{\mathrm{d}p} &\le \frac{\ln \kappa}{q}\int_{1/\kappa}^{1} q\alpha^{q-1}\,\mathrm{d}\alpha \\ &\le \ln \kappa \int_{1/\kappa}^{1} \frac{1}{\kappa^{q-1}}\,\mathrm{d}\alpha \\ &\le (\ln \kappa) \times \kappa^{1-q} \le \kappa \ln \kappa. \end{aligned}$$

Part 2. Part 1 implies that

$$\kappa \ln \kappa \geq \frac{\mathrm{d}(\ln M_p(z))}{\mathrm{d}p} = \frac{1}{M_p(z)} \cdot \frac{\mathrm{d}(M_p(z))}{\mathrm{d}p},$$

or that $\frac{\mathrm{d}(M_p(z))}{\mathrm{d}p} \leq (\kappa \ln \kappa) M_p(z)$ for all $z \in [L, U]^d$ and $p \leq 1$.

Given $x \in \mathcal{D}$, denote the d-dimensional random variable $Z = \mathbf{h}(x)$, then by Assumption 6.1 we have that $Z \in [L, U]^d$ always. We prove the result when Z is a discrete random variable; essentially the same proof holds for continuous random variables. By the above, we have that

$$\begin{aligned}
\frac{\mathrm{d}(\ln \mathbb{E}[M_p(Z)])}{\mathrm{d}p} &= \frac{1}{\mathbb{E}[M_p(Z)]} \cdot \frac{\mathrm{d}(\mathbb{E}[M_p(z)])}{\mathrm{d}p} \\
&= \frac{1}{\mathbb{E}[M_p(Z)]} \cdot \sum_z \Pr(Z = z) \frac{\mathrm{d}(\mathbb{E}[M_p(z)])}{\mathrm{d}p} \\
&\leq \frac{1}{\mathbb{E}[M_p(Z)]} \cdot \sum_z \Pr(Z = z) \cdot ((\kappa \ln \kappa) M_p(z)) \\
&= (\kappa \ln \kappa \cdot) \frac{\mathbb{E}[M_p(Z)]}{\mathbb{E}[M_p(z)]} = \kappa \ln \kappa.
\end{aligned}$$

Part 3. Note that $\ln \mathrm{OPT}_p = \ln \max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x))] = \max_{x \in \mathcal{D}} \ln \mathbb{E}[M_p(\mathbf{h}(x))]$. Given distinct $p, q \leq 1$, denote $x_p = \arg\max_{x \in \mathcal{D}} \mathbb{E}[M_p(\mathbf{h}(x)]$. Then, by Part 1,

$$\begin{aligned}
\frac{\ln \mathrm{OPT}_q - \ln \mathrm{OPT}_p}{q - p} &= \frac{(\max_{x \in \mathcal{D}} \ln \mathbb{E}[M_q(\mathbf{h}(x))]) - \ln \mathbb{E}[M_p(\mathbf{h}(x_p))]}{q - p} \\
&\leq \frac{\ln \mathbb{E}[M_q(\mathbf{h}(x_p))] - \ln \mathbb{E}[M_p(\mathbf{h}(x_p))]}{q - p} \leq \kappa \ln \kappa. \qquad \square
\end{aligned}$$

We are ready to prove Theorem 6.1 that gives guarantees for p-`MeanPortfolio` (Algorithm 7).

Proof of Theorem 6.1. We prove that (correctness) the set X of policies output by the algorithm is indeed an α-approximate portfolio, (size bound) $|X| = O\left(\frac{\ln \kappa}{\ln(1/\alpha)}\right)$, and (oracle

complexity) the number of oracle calls to Equation 6.3 is upper bounded by

$$O\left(\frac{(\ln\kappa)^2 \ln\ln d}{\ln(1/\alpha)\ln\ln(1/\alpha)}\right). \tag{C.4}$$

We note that this size bound can be tightened slightly to $O\left(\frac{(\ln\kappa)(\ln\kappa+\ln\ln d)}{\ln(1/\alpha)\ln\ln(1/\alpha)}\right)$; we present the the above cleaner bound for simplicity.

Correctness. Suppose the algorithm returns portfolio $X = \{x_0, x_1, \ldots, x_K\} \subseteq \mathcal{D}$ such that $x_t = \arg\max_{x\in\mathcal{D}} \mathbb{E}[M_{p_t}(\mathbf{h}(x))]$ with $-\frac{\ln d}{\ln(1/\alpha)} = p_0 < p_1 < \ldots < p_K = 1$.

Lemma C.3 shows that x_0 is an α-approximation for all $p \le p_0$. It is therefore sufficient to prove that for all $t \in [0, K-1]$, policy x_t is an α-approximation for all $p \in [p_t, p_{t+1}]$. Since $p_{t+1} = \texttt{LineSearch}(p_t, \alpha)$, Lemma C.6 implies this.

Size bound. Bby definition, $p_{t+1} = \texttt{LineSearch}(p_t, \alpha)$. Therefore, from Lemma C.4, we have that x_t satisfies $\mathrm{OPT}_{p_{t+1}} \ge \frac{\mathrm{OPT}_{p_t}}{\sqrt{\alpha}}$ for all t except possibly $t = K-1$. Therefore,

$$\mathrm{OPT}_{p_K} \ge \mathrm{OPT}_{p_{K-1}} \ge \left(\frac{1}{\alpha}\right)^{(K-1)/2} \mathrm{OPT}_{p_0} = \left(\frac{1}{\alpha}\right)^{(K-1)/2} \mathrm{OPT}_{-\infty}.$$

However, $\mathrm{OPT}_{p_K} \le U$ and $\mathrm{OPT}_{p_0} \ge L$, so that $\frac{\mathrm{OPT}_{p_K}}{\mathrm{OPT}_{p_0}} \le \frac{U}{L} = \kappa$, and therefore,

$$\frac{K-1}{2} \le \log_{(1/\alpha)} \kappa = \frac{\ln\kappa}{\ln(1/\alpha)}.$$

Oracle complexity. To bound the oracle complexity, we will show that each run of `LineSearch` calls the oracle to solve Equation 6.3 at most $O\left(\ln\left(\kappa|p_0|\right)\right)$ times, where $p_0 = -\frac{\ln d}{\ln(1/\alpha)}$ is the first iterate in p-`MeanPortfolio` and $\kappa = U/L$ is the condition number of the rewards. Since `LineSearch` is called at most $O(K) = O\left(\frac{\ln\kappa}{\ln(1/\alpha)}\right)$ times, this implies the bound Equation C.4.

To bound the number of oracle calls in `LineSearch`, we will use Lemma C.7.3 that upper bounds the slope of $\ln \mathrm{OPT}_p$ by $m := \kappa\ln\kappa$. Suppose, for a given $p \ge p_0 = -\frac{\ln d}{\ln(1/\alpha)}$

that `LineSearch` on input p, α finished in j iterations. Then, we have the following for `LineSearch`:

1. $p_0 \leq p \leq a < b \leq 1$ at all times, and
2. except in the last iteration, we have $\mathbb{E}[M_a(\mathbf{h}(x))] < \alpha \ \mathrm{OPT}_b$ (otherwise `Line-Search` terminates by step 3).

Denote by $a^{(j-1)}, b^{(j-1)}$ the value of a, b in iteration $j-1$. Then, since $b-a$ is halved in each iteration, we must have

$$0 < b^{(j-1)} - a^{(j-1)} = (1-p)2^{-(j-1)} \leq (1-p_0)2^{-(j-1)} \leq \frac{4\ln d}{\ln(1/\alpha)} 2^{-j}.$$

However, since the algorithm does not terminate in iteration $j-1$, as discussed, we must also have that

$$\mathbb{E}[M_{a^{(j-1)}}(\mathbf{h}(x))] < \alpha \cdot \mathrm{OPT}_{b^{(j-1)}}. \tag{C.5}$$

From Lemma C.7, we get that

$$\ln \frac{\mathrm{OPT}_{b^{(j-1)}}}{\mathrm{OPT}_{a^{(j-1)}}} \leq (\kappa \ln \kappa)(b^{(j-1)} - a^{(j-1)}) \leq \frac{4\kappa(\ln\kappa)(\ln d)}{\ln(1/\alpha)} 2^{-j}. \tag{C.6}$$

However, from Lemma C.5, we get $\mathbb{E}[M_{a^{(j-1)}}(\mathbf{h}(x))] \geq \sqrt{\alpha}\ \mathrm{OPT}_{a^{(j-1)}}$. Putting these together with Equation C.5 and Equation C.6, we get that

$$\frac{1}{2}\ln(1/\alpha) \leq \frac{4\kappa(\ln\kappa)(\ln d)}{\ln(1/\alpha)} 2^{-j}.$$

Therefore,

$$2^j \leq \frac{8\kappa(\ln\kappa)(\ln d)}{(\ln(1/\alpha))^2}.$$

Equivalently, the number of iterations j in `LineSearch` is bounded by

$$O\left(\ln\left(\frac{\kappa \ln d}{\ln(1/\alpha)}\right)\right) = O\left(\frac{(\ln\kappa)(\ln\ln d)}{\ln\ln(1/\alpha)}\right). \qquad \square$$

APPENDIX D

ADDITIONAL EXPERIMENTAL DETAILS FROM Chapter 6

D.1 Comparison with other MORL Algorithms

[126] optimize a generalized Gini welfare assuming a fixed weight vector. We address a fundamentally different scenario, where decision-makers are uncertain about the appropriate fairness criterion (e.g., the choice of p in p-means or the selection of weights in Gini welfare) in advance. Optimizing a single policy for one fairness parameter can lead to poor outcomes under different criteria (see the approximation qualities of random baselines in our experiments). Our method computes a small, representative set of policies covering the entire spectrum of fairness criteria. This allows decision-makers to select confidently from this set, without worrying about suboptimality under other potential choices of p.

[148] focuses on weighted linear objectives and learns a single policy, whereas we focus on p-means and compute a portfolio (multiple policies). [149] builds portfolios focusing on weighted linear objectives (as opposed to p-means) and does not offer guarantees on portfolio size or oracle complexity. [150] trains a single policy to approximate the Pareto frontier, which does not directly correspond to any social welfare function. To the best of our knowledge, our work is the first to (1) propose a multi-policy approach in p-means and (2) provide theoretical guarantees on portfolio size, approximation quality, and oracle complexity simultaneously.

In Table D.1, we also present comparisons between p-`MeanPortfolio` and the Generalized Policy Improvement (GPI) algorithm of [149], which maintains a set of policies $\Pi' \subseteq \Pi$ and iteratively chooses the weight vector $\mathbf{w}$ with the highest difference between the optimal policy value for $\mathbf{w}$ in Π vs in Π'. Our implementation of GPI uses the differential evolution solver from scipy for this global optimization step. The results in the table indi-

Table D.1: A comparison of actual approximation ratios across various portfolio sizes for p-MeanPortfolio and our implementation of GPI.

Portfolio Size	p-Mean-Portfolio	GPI
Resource Allocation after Natural Disaster		
1	**0.706**	0.514
2	**0.904**	0.520
3	**0.999**	0.520
4	**0.100**	0.520
Healthcare Intervention		
1	**0.924**	0.347
2	**0.982**	0.641
3	**0.982**	0.641
4	**0.982**	0.641
5	**0.993**	0.641
6	**0.999**	0.641
7	**1.000**	0.641

cate that our approach achieves significantly better approximation quality. We omit other details of GPI here and refer the interested reader to [149].

D.2 Additional Experimental Results

Figure D.1 and Figure D.2 illustrate how the optimal policies for different p values in the portfolio lead to varying impacts for the stakeholders. These portfolios are all obtained using p-MeanPortfolio. We also include the values of p chosen by p-MeanPortfolio to construct the portfolio in Table D.2.

D.3 Experimental Details

We include the details of various experiments here.

D.3.1 Taxi Environment

Problem Setting. This environment consists of a taxi agent whose task is to deliver passengers from source to destination. The world consists of a 6x6 grid and based on the en-

Table D.2: The set of values of p chosen by p-`MeanPortfolio` to obtain the corresponding portfolios.

Portfolio size	p Values
Resource Allocation After Natural Disaster	
1	-0.829
2	-4.864, -0.466
3	-11.136, -2.034, 0.621
4	-23.58, -5.15, -0.537, 0.712
Healthcare Intervention	
1	-1.325
2	-2.864, 0.034
3	-4.333, -0.333, 0.333
4	-6.641, -0.433, -0.075, 0.463
5	-9.216, -4.108, -0.437, -0.078, 0.461
6	-13.801, -6.4, -0.594, -0.22, 0.085, 0.542
7	-17.793, -3.698, -0.549, -0.186, -0.038, 0.222, 0.611
Taxi Environment	
1	-2.0
2	-3.89, 0.87
3	-6.21, 0.72, 0.86
8	-13.16, -10.06, -6.17, -4.82, 0.66, 0.7, 0.77, 0.89
10	-27.03, -13.02, -6.25, 0.66, 0.71, 0.78, 0.81, 0.86, 0.89, 0.95

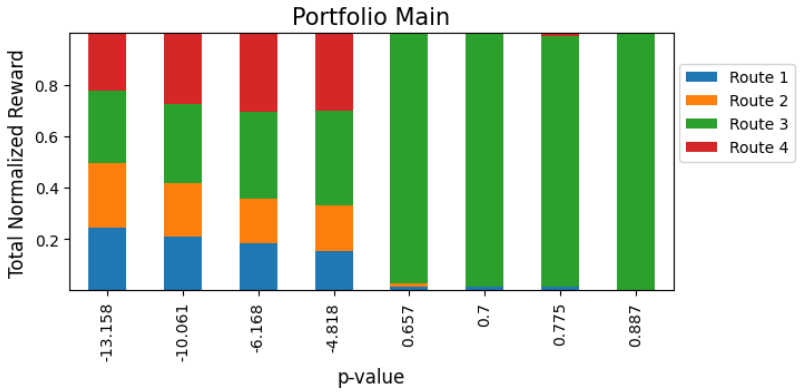


Figure D.1: Normalized total reward for each route under different policies in the portfolio generated by p-`MeanPortfolio` in the taxi environment.

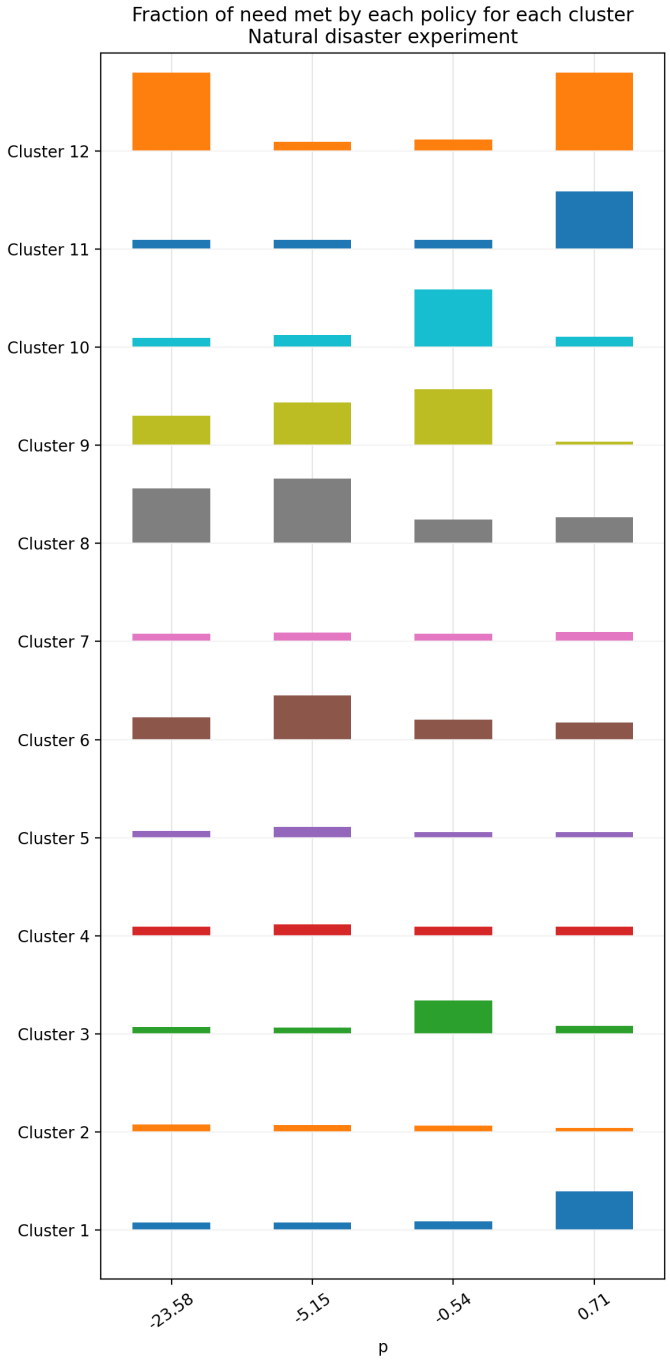


Figure D.2: Fraction of need met for various clusters by various policies in the portfolio obtained by p-`MeanPortfolio` after $H = 4$ intervention steps for the natural disaster experiment. Note that different solutions in the portfolio emphasize different clusters.

vironment setup of [41], we consider 4 source-destination pairs. Whenever a taxi moves to a source point, it can pick a passenger, move to destination and drop the passenger, thus receiving a reward. Since some routes could be easier to serve, the agent has to decide which route to serve more often. Rewards are multi-dimensional, each dimension corresponding to each route. We consider the following source-destination pairs: source coordinates $= ((0,0),(0,5),(3,0),(1,0))$ and destination coordinates $= ((1,5),(5,0),(3,3),(0,3))$.

MDP Structure.

State Space (S)

The state space consists of information about the location of the taxi, the location of passengers, and whether passengers have been picked up by the taxi at the moment.

Action Space (S)

The agent can take one of 6 actions: move north, south, east, west and pick or drop a passenger in the current grid cell.

Reward (R)

The reward function gives a reward of 0 for moving, -10 reward for taking an invalid action of picking or dropping a passenger at the wrong coordinate, and reward 30 reward for dropping a passenger at the right destination.

Learning Policy

Given a scalarization function (or welfare function), the task is to find a policy π^* that maximizined the expected value of scalarized return. Specifically:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau\sim\pi}\left[M_p\big(\mathbf{G}(\tau)\big)\right]. \tag{D.1}$$

We learn this policy using the Welfare Q-Learning algorithm proposed in [41]. For every problem setting, we train the policy for 200 episodes. Every episode is a finite-horizon problem with 1000 timesteps. We consider the discount factor $\gamma = 0.99$.

D.3.2 Natural Disaster

Problem Setting. In this synthetically generated example, suppose that in the wake of a natural disaster, a centralized aid agency must determine how to allocate resources to a set of 12 clusters ($d = 12$). Each cluster is characterized by their population density (high or low), proximity to critical infrastructure (near or far), and the predominant income level of its residents (low, middle, or high).

Table D.3: Clustered population data including density, proximity, income level, total population, and initial need.

Cluster ID	Density	Proximity	Income Level	Total Population	Initial Need
1	High	Far	High-Income	148	150
2	High	Far	Low-Income	307	500
3	High	Far	Middle-Income	616	650
4	High	Near	High-Income	816	300
5	High	Near	Low-Income	1405	1000
6	High	Near	Middle-Income	2782	950
7	Low	Far	High-Income	74	1000
8	Low	Far	Low-Income	203	350
9	Low	Far	Middle-Income	396	300
10	Low	Near	High-Income	36	50
11	Low	Near	Low-Income	113	100
12	Low	Near	Middle-Income	230	100

MDP Formulation and Experimental Protocol. We model post–disaster resource allocation to $d = 12$ clusters as a finite-horizon MDP. Let H denote the time horizon. Throughout, we use k for the allocation increment and K for the per-period budget (these correspond to b and B in the informal description above). All needs and allocations lie on the k-grid.

State space. At time t, the state is $s_t = (s_{t,1}, \ldots, s_{t,d}) \in \mathbb{Z}^d_{\geq 0}$, where $s_{t,i}$ is the unmet need of cluster i (in units of k). We initialize s_0 from Table D.3. Feasible states are those reachable by repeatedly (i) subtracting allocated units and (ii) possibly increasing one unattended cluster by k (see transitions below).

Action space. An action is an allocation vector $a_t = (a_{t,1}, \ldots, a_{t,d})$ with $a_{t,i} \in \{0, k, 2k, \ldots, K\}$ and a per-period budget constraint $\sum_{i=1}^{d} a_{t,i} \leq K$. We impose a *validity constraint*:

$$\text{if } s_{t,i} = 0 \;\Rightarrow\; a_{t,i} = 0,$$

i.e., we never allocate to a cluster whose need is already zero. In implementation, we enumerate the full feasible action space $\mathcal{A} = \{a : \sum_i a_i \leq K\}$ and then state-filter to $\mathcal{A}(s_t) = \{a \in \mathcal{A} : s_{t,i} = 0 \Rightarrow a_i = 0\}$.

Transitions. Given s_t and a_t, define the *interim* post-allocation state

$$x_{t,i} \;=\; \max\{0,\, s_{t,i} - a_{t,i}\}, \qquad x_t = (x_{t,1}, \ldots, x_{t,d}).$$

Let $U_t = \{\, i : \, x_{t,i} > 0 \text{ and } a_{t,i} = 0 \,\}$ be the set of *unattended* clusters that still have unmet need. With probability p no spillover occurs and $s_{t+1} = x_t$. With probability $1 - p$, exactly one unattended cluster $j \in U_t$ is chosen uniformly and its need increases by k:

$$s_{t+1,i} \;=\; \begin{cases} x_{t,i} + k, & i = j, \\ x_{t,i}, & i \neq j. \end{cases}$$

The rationale behind this model behavior is that if need remains unmet, there can possibly be an increase in need over time. For example, lack of resources might lead to an increase in need for medical care, which will compound the total resources needed in the next time period. If $U_t = \emptyset$, then $s_{t+1} = x_t$ deterministically.

Policy menu. We evaluate "reasonable" structured policies that map a state s to a valid action $a \in \mathcal{A}(s)$ using the following priorities:

1. **Need-based:** allocate all K to $\arg\max_i s_i$.
2. **Per-capita need:** allocate all K to $\arg\max_i s_i/\text{pop}_i$.
3. **Population:** allocate all K to $\arg\max_i \text{pop}_i$.
4. **Income-priority:** prioritize Low $\succ$ Middle $\succ$ High income (breaking ties by higher s_i) and allocate all K to the first in this order.
5. **Proximity-priority:** prioritize Near $\succ$ Far (ties by higher s_i) and allocate all K accordingly.
6. **Randomized weighted hybrid (implemented).** Sample weights w_{need}, w_{pc}, w_{inc}, $w_{\text{prox}} \sim \text{Unif}(0,1)$ and normalize to sum to one; score each cluster

$$\text{score}_i \;=\; w_{\text{need}}\, s_i + w_{\text{pc}}\, \frac{s_i}{\text{pop}_i} + w_{\text{inc}} \cdot \mathbf{1}\{\text{income}_i = \text{Low}\} + w_{\text{prox}} \cdot \mathbf{1}\{\text{prox}_i = \text{Near}\}.$$

   Let $i^\star \in \arg\max_i \text{score}_i$. If $s_{i^\star} \geq K$, allocate $a_{i^\star} = K$. Otherwise allocate $a_{i^\star} = k$ and allocate the remaining k (if available) to a uniformly random other cluster with $s_i > 0$; all other coordinates are 0. (Allocations are always in increments of k.)

In our simulation, when a state is first encountered we *sample* one policy from this menu and fix it for that state for the remainder of the rollout. This produces a state–action map that varies across reachable states without solving a global control problem.

Reward and optional allocation bonus. Let need_i^0 be cluster i's initial need. For a transition $s \to s'$ the per-cluster reward is

$$r_i(s, a, s') \;=\; \left(\frac{a_i}{H\,K}\right) \cdot (1 + \beta_i).$$

The optional global equity multiplier $\beta_i \geq 0$ enables *global constraints/bonuses*:

$$\beta_i \;=\; \underbrace{\omega\,\mathbf{1}\{i \in \mathcal{C}\}}_{\text{cluster set bonus}} + , \qquad \beta_i \leftarrow \min\{\beta_i,\ \beta_{\max}\},$$

where ω is a user-set weight, $\mathcal{C}$ a designated set of clusters to prioritize (e.g., $\{2\}$), and $\beta_{\max}$ an optional cap. Setting $\omega = 0$ recovers the unmodified reward.

Expected returns, allocations, and what we report. Given (s, a), we enumerate all feasible s' and their probabilities, and compute per-cluster expected rewards

$$\mathbb{E}[\, r_i(s, a, s')\,] \;=\; \sum_{s'} P(s'|s, a)\, r_i(s, a, s').$$

Over the horizon we accumulate (i) the *expected cumulative reward* $R_i = \sum_{t=0}^{H-1} \Pr(\text{reach } s_t)\ \mathbb{E}[r_i(s_t, a_t, S_{t+1})]$ and (ii) the *expected total allocation* $A_i = \sum_{t=0}^{H-1} \Pr(\text{reach } s_t)\, \mathbb{E}[a_{t,i}]$. We report both R_i and A_i for each cluster.[1]

Rollout and tractability. We avoid Monte Carlo sampling by *enumerating* next states but retain only those with probability $> \varepsilon$ (default $\varepsilon = 10^{-6}$). Newly reached states are added to a frontier with their path probability; we prevent revisiting already-expanded states to keep the graph acyclic in practice. This yields a sparse, tree-like expansion that preserves essentially all probability mass while remaining tractable even with a large discrete action space $\mathcal{A}$ (whose size scales as $O((K/k + 1)^N)$).

Parameterization. Unless otherwise noted, we use the k-grid and budget specified in the problem setting (e.g., $k = 50$, $K = 150$), a spillover parameter $p \in (0, 1)$ controlling unattended-need inflation, and a finite horizon H.

[1] In code, both are computed by weighting each step's contributions by the path probability mass.

D.3.3 Healthcare Intervention

The Healthcare Intervention problem is based on the large-scale mobile-health program run by the NGO ARMMAN [22]. The goal of the program is to maximize engagement of beneficiaries with the program using limited service call interventions. Based on previous works, we model this problem as RMAB problem where we have multiple arms or beneficiaries, a budget on the number of service calls to give every week. For every beneficiary, we have information on their listenership with the voice call. This is considered as an engaging or non-engaging state if the listenership is above or below 30-seconds threshold respectively. One week of time is considered as one timestep. Finally, every week, the action can be to place (active) or not to place a live service call (passive). Additionally, for every beneficiary, we have information on their socio-demographic characteristics such as age, income, education. We use real world data from service quality improvement conducted by ARMMAN in January 2022 [151]. Further, for our experiments, we sample data for 2100 beneficiaries, and run the experiment for $H = 12$ timesteps.

Generation of Policies and Reward Functions. The default reward incentivizes agents to be in engaging state. Thus, agents receive reward 1 if they are in engaging state, and 0 otherwise. However, due to varying policy needs over time or over different geographies, health workers often have to prioritize some beneficiaries more than others. To capture these varying priorities, we leverage the Social Choice Language Model framework [40] to derive reward functions from natural-language commands. Given a command specifying d preferences (each indicating a subgroup to prioritize) this method generates two sets of reward functions:

1. **Individual preferences:** one reward function per preference (total of d reward functions) which we treat as each stakeholder's reward function.

2. **Balancing functions:** a collection of reward functions that trade off among the d

preferences, whose corresponding policies form our feasible set Π.

In our experiments, we set $d = 59$ and construct $|\Pi| = 200$ balancing policies.

Learning Policy. It is computationally intractable to optimally solve the RMAB problem [81]. Thus, we use the commonly used Whittle Index Heuristic [132] to solve RMAB problem for a given single reward function. Specifically, Whittle Index quantifies the reward gain achieved for every arm in every state if we took the active action as compared to if we took the passive action. The policy then chooses the arms with the highest Whittle indices within the budget limit.

Baseline Policy. The baseline policy mentioned in Figure Figure 1.2 is trained on the default reward.

D.3.4 Other Details

Initial p for `BudgetConstrainedPortfolio`. We fix $p_0 = -100$.

Computing actual approximation factor. The approximation factor $\mathcal{Q}(\Pi')$ for any portfolio Π' is computed via a grid search over 1000 points in $[\infty, 1]$ for the natural disaster environment and the healthcare intervention problem. For the taxi environment, we used 50 points due to the computational burden of solving each problem.

Choice of α. For a given portfolio size K, we chose the smallest α in the set $\{0.05, 0.10, \ldots, 0.90, 0.95\} \cup \{0.99\}$ that has the resulting portfolio size K.

APPENDIX E

OMITTED PROOFS FROM Chapter 7

We provide proofs omitted from Chapter 7 here.

E.1 Omitted Proofs from Subsection 7.3.1

To prove Lemma 7.1, we need the following concentration inequality:

Lemma E.1 (Bernstein inequality [152], Theorem 2)**.** *Let $X_1, \ldots, X_N$ be negatively associated zero-mean random variables with $|X_i| \leq 1$ for all $i \in [N]$ with probability 1. Let $z_1, \ldots, z_N$ be nonnegative constants. Then, for all $\delta > 0$,*

$$\Pr\left(\sum_{i\in[N]} z_i X_i \geq \delta\right) \leq \exp\left(-\frac{3\delta^2}{2\delta \max_{i\in[N]} z_i + 6\sum_{i\in[N]} z_i^2 \mathbb{E}[X_i^2]}\right).$$

We prove the Lemma 7.1 now, which bounds the dual block norms of sparse vectors:

Lemma 7.1. *Consider a vector $c \in \{0,1\}^d$ that is S-sparse, i.e., $\|c\|_1 \leq S$, and a block norm $\|\cdot\|_{[n]}$ induced by a random equal n-partition of $[d]$. Then the expected square of the dual norm $\mathbb{E}\left[\left(\|c\|_{[n]}^*\right)^2\right]$ is bounded above by*

$$\mathbb{E}\left[\left(\|c\|_{[n]}^*\right)^2\right] \leq 6\max\left\{\frac{S}{n}, \ln n\right\}.$$

Proof. Recall that given blocks $B_1, \ldots, B_n$ that partition $[d]$, the dual block norm of c is $\max_{j\in[n]} \|c_{B_j}\|_2$, where c_{B_j} is the restriction of c to the coordinates in block B_j. Suppose blocks $B_j, j \in [n]$ are chosen randomly and are of equal size.

Fix some block $j \in [n]$. For $i \in [S]$, let Y_i denote the indicator random variable for whether the ith non-zero coordinate in c lies in B_j. Note that $\mathbb{E}\left[\|c_{B_j}\|_1\right] = \mathbb{E}\left[\sum_{i\in[S]} Y_i\right] =$

$\frac{S}{n}$. Also note that random variables $\{Y_i, i \in [S]\}$ are negatively associated (see [153]). Let $X_i = Y_i - \frac{1}{n}$; then $\mathbb{E}[X_i] = 0$ for each i and the random variables $\{X_i, i \in [S]\}$ are also negatively associated (again follows from [153]).

Setting $z_i = 1$ for all i in Lemma E.1, we get that for all $\delta > 0$

$$\begin{aligned}\Pr\left(\|c_{B_j}\|_1 \geq \frac{S}{n} + \delta\right) &= \Pr\left(\sum_{i\in[S]} Y_i \geq \frac{S}{n} + \delta\right) = \Pr\left(\sum_{i\in[S]} X_i \geq \delta\right) \\ &\leq \exp\left(-\frac{3\delta^2}{2\delta + 6\sum_{i\in[S]}\mathbb{E}[X_i^2]}\right) \\ &= \exp\left(-\frac{3\delta^2}{2\delta + \frac{6S(1-1/n)}{n}}\right) < \exp\left(-\frac{3\delta^2}{2\delta + \frac{6S}{n}}\right).\end{aligned}$$

Choose $\delta = \max\left(4\ln n, 4\sqrt{\frac{S\ln n}{n}}\right)$. When $S \leq n\ln n$, we get

$$\exp\left(-\frac{3\delta^2}{2\delta + \frac{6S}{n}}\right) \leq \exp\left(-\frac{48\ln^2 n}{8\ln n + 6\ln n}\right) \leq \frac{1}{n^2}.$$

When $S > n\ln n$, we get that $\sqrt{\frac{S\ln n}{n}} \leq \frac{S}{n}$, so that

$$\exp\left(-\frac{3\delta^2}{2\delta + \frac{6S}{n}}\right) \leq \exp\left(-\frac{48\frac{S\ln n}{n}}{8\sqrt{\frac{S\ln n}{n}} + \frac{6S}{n}}\right) \leq \exp\left(-\frac{\frac{48S\ln n}{n}}{\frac{14S}{n}}\right) \leq \frac{1}{n^2}.$$

In either case,

$$\Pr\left(\|c_{B_j}\|_1 \geq \frac{S}{n} + \delta\right) \leq \frac{1}{n^2}.$$

By taking a union bound over the n blocks $j = 1, \ldots, n$,

$$\Pr\left(\max_{j\in[n]} \|c_{B_j}\|_1 \geq \frac{S}{n} + \delta\right) \leq \frac{1}{n}. \tag{E.1}$$

Since $c \in \{0,1\}^d$, $\|c_{B_j}\|_2^2 = \|c_{B_j}\|_1$ for all j, and therefore, the expectation of the square

of dual block norm of c is

$$\mathbb{E}\left[\max_{j\in[n]}\|c_{B_j}\|_2^2\right] = \mathbb{E}\left[\max_{j\in[n]}\|c_{B_j}\|_1\right].$$

From Equation E.1, and since $\max_{j\in[n]}\|c_{B_j}\|_1 \le S$ always, we get that

$$\begin{aligned}\mathbb{E}\left[\max_{j\in[n]}\|c_{B_j}\|_2^2\right] &\le \left(1-\frac{1}{n}\right)\left(\frac{S}{n}+\delta\right)+\frac{1}{n}S \\ &\le \frac{2S}{n}+4\max\left(\ln n, \sqrt{\frac{S\ln n}{n}}\right) \\ &\le 6\max\left(\frac{S}{n}, \ln n, \sqrt{\frac{S\ln n}{n}}\right).\end{aligned}$$

Finally, note that $\sqrt{\frac{S\ln n}{n}} = \sqrt{\frac{S}{n}}\sqrt{\ln n} \le \max\left(\frac{S}{n}, \ln n\right)$. □

E.2 Proof of Lemma 7.2

We prove Lemma 7.2 now, which bounds D_n for block norms over the probability simplex.

Proof of Lemma 7.2. Denote $z = x^{(1)}$, First, we compute the gradient $\nabla h_n(z)$. For coordinate i that lies in block B_j, we have

$$\nabla_i h_n(z) = \frac{1}{\gamma_n p_n} \times p_n\|z_{B_j}\|_2^{p_n-1} \times \frac{1}{2\|z_{B_j}\|_2} \times 2z_i = \frac{1}{\gamma_n}\|z_{B_j}\|_2^{p_n-2} z_i.$$

Therefore,

$$\langle \nabla h_n(z), z\rangle = \sum_{j\in[n]}\|z_{B_j}\|_2^{p_n-2}\sum_{i\in B_j}(z_i\cdot z_i) = p_n h_n(z).$$

Consequently,

$$\gamma_n B_{h_n}(x\|z) = \gamma_n h_n(x) - \gamma_n h_n(z) - \gamma_n\langle\nabla h_n(z), x-z\rangle$$

$$= (p_n - 1)\gamma_n h_n(z) + \gamma_n h_n(x) - \gamma_n \langle \nabla h_n(z), x \rangle.$$

However, since $x, z \in \mathbf{\Delta}_d$, we have $\langle \nabla h_n(z), x \rangle \geq 0$ (since each term is nonnegative). Further, since $1 \leq p_n \leq 2$, we have that

$$\gamma_n h_n(x) = \frac{1}{p_n} \sum_{j \in [n]} \|x_{B_j}\|_2^{p_n} \leq \frac{1}{p_n} \sum_{j \in [n]} \|x_{B_j}\|_2 \leq \frac{1}{p_n} \sum_{j \in [n]} \|x_{B_j}\|_1 = \frac{1}{p_n}.$$

Similarly, $\gamma_n h_n(z) \leq \frac{1}{p_n}$. Therefore,

$$\gamma_n B_{h_n}(x \| z) \leq (p_n - 1)\gamma_n h_n(z) + \gamma_n h_n(x) \leq \frac{p_n - 1}{p_n} + \frac{1}{p_n} = 1. \qquad \square$$

E.3 Omitted Proofs from Subsection 7.3.3

We prove Lemma 7.3 and Lemma 7.4 here.

E.3.1 Proof of Lemma 7.3

Similar to the proof of Lemma 7.2, we get that

$$\gamma_S B_{h_S}(z \| x^{(1)}) \leq \gamma_S h_S(z) + (p_S - 1)\, \gamma_S h_S(x^{(1)}) \leq p_S \gamma_S \max_{x \in P} h_S(x).$$

Therefore, $B_{h_S}(z \| x^{(1)}) \leq (p_S \gamma_S \max_{x \in P} h_S(x)) \cdot O(\ln d)$ since $\gamma_S = \frac{1}{e \ln S}$. Therefore, it is sufficient to prove that $\max_{x \in P} h_S(x) \leq \frac{1}{p_S \gamma_S}$.

Since h_S is convex, the maximum occurs at some vertex of P. For each $i \in [d]$, $h_S(\mathbf{e}_i) = \frac{1}{p_S \gamma_S}$. For the remaining vertex $x^{(1)} = A\mathbf{1}_d$, we have (given blocks $B_1, \ldots, B_S$):

$$\begin{aligned} p_S \gamma_S h_S(x^{(1)}) &= \sum_{j \in [S]} \|x_{B_j}\|_2^{p_S} = \sum_{j \in [S]} A^{p_S} \left(\frac{d}{S}\right)^{\frac{p_S}{2}} = S A^{p_S} \left(\frac{d}{S}\right)^{\frac{p_S}{2}} \\ &= (A^2 d S)^{p_S/2} \cdot S^{1 - p_S}. \end{aligned}$$

Since $A = d^{-2/3}$, $S = d^{1/3}$ and $p_S \geq 1$, we have that $(A^2 dS)^{p_S/2} \cdot S^{1-p_S} \leq 1$. This completes the proof.

E.3.2 Proof of Lemma 7.4

We restate the lemma here for convenience.

Lemma 7.4. *The iterates $z^{(1)}, \ldots, z^{(T)} \in \hat{P}$ of OMD with dth block norm satisfy with high probability*

$$z_1^{(t)} \leq \frac{1}{d} + \frac{\sqrt{K}}{R\sqrt{R}}(t-1),$$

where $K = \frac{128}{T} \ln^2(dT)$.

Recall our plan for the proof: we will show that for consecutive iterates $z^{(t)}, z^{(t+1)}$ played by the algorithm, the Bregman divergence $B_{h_d}(z^{(t+1)} \| z^{(t)})$ is sandwiched between

$$\frac{1}{2}\|z^{(t)} - z^{(t+1)}\|_1^2 \leq B_{h_d}(z^{(t+1)} \| z^{(t)}) \leq \frac{K}{2R}. \tag{E.2}$$

Proof of Lemma 7.4 assuming Equation E.2. First, we prove that this L_1 norm bound implies the desired coordinate-wise bound, and then return to the proof of this inequality. Recall that scaled polytope $\hat{P} = \mathrm{conv}(\frac{1}{R}\mathbf{e}_1, \ldots, \frac{1}{R}\mathbf{e}_d, \frac{1}{d}\mathbf{1}_d)$ for $R = d^{1/3}$.

Lemma E.2. *Consider a sequence of points $\{z^{(t)}\}_{t \geq 1}$ in $\hat{P}$ such that $z^{(1)} = \frac{1}{d}\mathbf{1}_d$ and $\|z^{(t+1)} - z^{(t)}\|_1^2 \leq B$ for some $B > 0$. Then these points satisfy*

$$z_1^{(t)} \leq \frac{1}{d} + \frac{\sqrt{B}}{R}(t-1).$$

Proof. Denote $v = \frac{1}{R}\mathbf{e}_1$. Let $\gamma = \frac{v_1 - z_1^{(1)}}{\|v - z^{(1)}\|_1} = \frac{\frac{1}{R} - \frac{1}{d}}{1 + \frac{1}{R} - \frac{2}{d}}$. Define

$$g(z) = z_1 - z_1^{(1)} - \gamma\|z - z^{(1)}\|_1.$$

We will show that $g(z) \leq 0$ for all $z \in \hat{P}$, so that we have $z_1 - z_1^{(1)} \leq \gamma\|z - z^{(1)}\|_1$. In

particular,

$$z_1^{(t)} - z_1^{(1)} \leq \gamma \|z^{(t)} - z^{(1)}\|_1 \leq \gamma \sum_{t'=1}^{t-1} \|z^{(t'+1)} - z^{(t')}\|_1 \leq \gamma(t-1)\sqrt{B}.$$

The result follows by noting that $\gamma = \frac{\frac{1}{R} - \frac{1}{d}}{1 + \frac{1}{R} - \frac{2}{d}} \leq \frac{1}{R}$ for all $R \geq 2$.

It remains to show that $g(z) \leq 0$ for $z \in \hat{P}$. Denote $v^{(i)} = \frac{1}{R}\mathbf{e}_i$, and write $z - z^{(1)} = \sum_{i \in [d]} \lambda_i (v^{(i)} - z^{(1)})$ where each $\lambda_i \geq 0$ and $\sum_{i \in [d]} \lambda_i := \phi \leq 1$. Then $z_1 - z_1^{(1)} = \frac{\lambda_1}{R} - \frac{\phi}{d}$. Further,

$$\begin{aligned}
\|z - z^{(1)}\|_1 &= \left\| \left(\frac{\lambda_1}{R} - \frac{\phi}{d}, \ldots, \frac{\lambda_d}{R} - \frac{\phi}{d} \right) \right\|_1 \\
&= \sum_{i=1}^{d} \left| \frac{\lambda_i}{R} - \frac{\phi}{d} \right| \geq \left(\frac{\lambda_1}{R} - \frac{\phi}{d} \right) + \left| \sum_{i=2}^{d} \left(\frac{\lambda_i}{R} - \frac{\phi}{d} \right) \right| \\
&= \left(\frac{\lambda_1}{R} - \frac{\phi}{d} \right) + \left| \frac{\phi - \lambda_1}{R} - \frac{(d-1)}{d}\phi \right| \\
&\geq \left(\frac{\lambda_1}{R} - \frac{\phi}{d} \right) - \left(\frac{\phi - \lambda_1}{R} - \frac{(d-1)}{d}\phi \right) \\
&= \frac{2}{R}\lambda_1 + \left(1 - \frac{2}{d} - \frac{1}{R} \right) \phi.
\end{aligned}$$

Therefore,

$$\begin{aligned}
g(z) &= z_1 - z_1^{(t)} - \gamma \|z - z^{(t)}\|_1 \\
&\leq \left(\frac{\lambda_1}{R} - \frac{\phi}{d} \right) - \gamma \left(\frac{2}{R}\lambda_1 + \left(1 - \frac{2}{d} - \frac{1}{R} \right) \phi \right) \\
&= \lambda_1 \left(\frac{1 - 2\gamma}{R} \right) - \phi \left(\frac{1}{d} + \gamma \left(1 - \frac{2}{d} - \frac{1}{R} \right) \right).
\end{aligned}$$

Since $\gamma \leq \frac{1}{R} \leq \frac{1}{2}$ and $\lambda_1 \leq \phi$, the above is bounded above by

$$\begin{aligned}
g(z) &\leq \phi \left(\frac{1 - 2\gamma}{R} \right) - \phi \left(\frac{1}{d} + \gamma \left(1 - \frac{2}{d} - \frac{1}{R} \right) \right) \\
&= \phi \left(\frac{1}{R} - \frac{1}{d} - \gamma \left(1 - \frac{2}{d} + \frac{1}{R} \right) \right) = \phi \left(\frac{1}{R} - \frac{1}{d} - \left(\frac{1}{R} - \frac{1}{d} \right) \right) = 0. \qquad \square
\end{aligned}$$

Proof of Lemma 7.4. From Equation E.2, the iterates $z^{(t)}$ of OMD with dth block norm satisfy the conditions of Lemma E.2 with $B = \frac{K}{R}$. The result then follows immediately. $\square$

Proof of Equation E.2. The lower bound in Equation E.2 follows from Theorem 7.1, since B_{h_d} is 1-strongly convex with respect to the L_1 norm in th L_1 norm ball, and $\hat{P}$ is contained in the L_1 norm ball. The upper bound is more involved and uses the structure of loss functions and the polytope.

Specifically, suppose $y^{(t)}$ denotes the intermediate point between iterates $z^{(t)}$ and $z^{(t+1)}$ in Algorithm 10 (i.e., $z^{(t+1)}$ is the minimizer of $B_{h_d}(z\|y^{(t)})$). Therefore, using the generalized Pythagorean theorem for Bregman divergences,

$$B_{h_d}(z^{(t+1)}\|z^{(t)}) \leq B_{h_d}(z^{(t+1)}\|y^{(t)}) + B_{h_d}(z^{(t)}\|y^{(t)}) \leq 2B_{h_d}(z^{(t)}\|y^{(t)}).$$

Therefore, it is sufficient to upper bound $B_{h_d}(z^{(t)}\|y^{(t)})$. However, given $z^{(t)}$, we can explicitly compute $y^{(t)}$ as a function of $z^{(t)}$, dimension d, and the step size η. This is done in Equation E.3 below. Then, we prove the desired upper bound $B_{h_d}(z^{(t)}\|y^{(t)}) \leq \frac{K}{4R}$ in Corollary E.1.

First, let us compute the step size η of OMD with dth block norm. Recall that $P = \mathrm{conv}(\mathbf{\Delta}_d, A\mathbf{1}_d)$, and the scaled polytope $\hat{P} = \frac{1}{Ad}P$. Recall also that we denote the rescaling parameter $R = Ad = d^{1/3}$ and the scaled starting point $z^{(1)} = \frac{1}{R}(A\mathbf{1}_d) = \frac{1}{d}\mathbf{1}_d$. Similar to the proof of Lemma 7.3 above, the diameter $\hat{D}_d$ of $\hat{P}$ is bounded above by $\sqrt{\frac{1}{\gamma_d}} = \sqrt{e \ln d}$. Next, $\hat{G}_d := \max_{x\in\mathcal{K}, t\in[T]} \|\nabla \hat{f}^{(t)}(x)\|_\infty = R$. Therefore, the step size

$$\eta \leq \frac{\hat{D}_d}{\hat{G}_d\sqrt{T}} \leq \frac{\sqrt{e \ln d}}{R\sqrt{T}}.$$

Next, we derive an explicit expression for the (unconstrained) update rule for OMD with

dth block norm. Given iterate $z \in \mathbb{R}^d_{\geq 0}$ and some set $C \subseteq [d]$ of coordinates, define

$$y_i = \begin{cases} \left(z_i^{\frac{1}{\ln d}} + \frac{\eta R}{e \ln d}\right)^{\ln d} & \text{if } i \in C, \\ z_i & \text{if } i \notin C. \end{cases} \tag{E.3}$$

Recall that the scaled loss function $\hat{f}^{(t)}(z) = -R\langle c^{(t)}, z\rangle$, where $c^{(t)}$ is 0-1 vector with exactly S non-zero coordinates, defined as follows: $c_1^{(t)} = 1$ always, and the remaining $S-1$ non-zero coordinates of $c^{(t)}$ are chosen uniformly at random from the remaining $d-1$ coordinates. Denote by $C^{(t)} = \{i \in [d] : c_i^{(t)} = 1\}$ the non-zero coordinates of $c^{(t)}$. To obtain the update rule, recall that $h_d(x) = \frac{1}{\gamma_d p_d} \sum_{i\in[d]} x_i^{p_d}$, where $p_d = 1 + \frac{1}{\ln d}$ and $\gamma_d = \frac{1}{e \ln d}$. The algorithm moves from some iterate $z^{(t)}$ to point $y^{(t)}$ and then to $z^{(t+1)} := \arg\min_{z\in\hat{P}} B_{h_d}(z\|y^{(t)})$. Then, it can be verified through a straightforward calculation using Algorithm 10 that the update rule is given precisely by the above Equation E.3 when $C = C^{(t)}$, $z = z^{(t)}$, and $y = y^{(t)}$. For convenience, we denote

$$\eta_0 = \eta R \leq \sqrt{\frac{e \ln d}{T}}.$$

Note that since $h_d(y) = \frac{e \ln d}{1+\frac{1}{\ln d}} \sum_{i\in[d]} y_i^{1+\frac{1}{\ln d}}$, we have $\nabla h(y) = (e \ln d) y^{1/\ln d}$. Thus,

$$\begin{aligned} B_{h_d}(z\|y) &= h_d(z) - h_d(y) - \langle \nabla h_d(y), z - y\rangle \\ &= \frac{e \ln d}{1 + \frac{1}{\ln d}} \sum_{i\in[d]} \left(z_i^{1+\frac{1}{\ln d}} - y_i^{1+\frac{1}{\ln d}}\right) + e \ln d \sum_{i\in[d]} (y_i - z_i) y_i^{\frac{1}{\ln d}}. \end{aligned}$$

The rest of the plan is as follows: given z and y as defined in Equation E.3, Lemma E.3 bounds the coordinate-wise components of $B_{h_d}(z\|y)$, and Lemma E.4 bounds $B_{h_d}(z\|y)$ in terms of z. Corollary E.1 gives the final desired bound in Equation E.2, using a concentration bound from Lemma E.5.

Lemma E.3. *For any $z \in \mathbb{R}_{\geq 0}$ and y defined in Equation E.3, for all $i \in [d]$,*

$$\frac{e \ln d}{1 + \frac{1}{\ln d}} \left(z_i^{1+\frac{1}{\ln d}} - y_i^{1+\frac{1}{\ln d}} \right) + e \ln d (y_i - z_i) y_i^{\frac{1}{\ln d}} \leq \frac{\eta_0^2}{2e} y_i^{1-\frac{1}{\ln d}}.$$

Proof. Denote the left-hand side by λ. Denote $a = z_i^{\frac{1}{\ln d}}$ and $\delta = \frac{\eta_0}{e \ln d}$. Then $y_i^{\frac{1}{\ln d}} = a + \delta$. Then

$$\begin{aligned}
\frac{\lambda}{e \ln d} &= \frac{1}{1 + \frac{1}{\ln d}} \left(a^{1+\ln d} - (a+\delta)^{1+\ln d} \right) + \left((a+\delta)^{\ln d} - a^{\ln d} \right)(a+\delta) \\
&= \frac{a^{1+\ln d}}{1 + \ln d} \left(\left(1 + \frac{\delta}{a} \right)^{1+\ln d} - (1 + \ln d) \left(1 + \frac{\delta}{a} \right) + \ln d \right) \\
&= \frac{a^{1+\ln d}}{1 + \ln d} \cdot \frac{1 + \ln d}{a} \int_0^\delta \left(\left(1 + \frac{\mu}{a} \right)^{\ln d} - 1 \right) \, \mathrm{d}\mu \\
&\leq a^{\ln d} \int_0^\delta (\ln d) \frac{\mu}{a} \left(1 + \frac{\mu}{a} \right)^{-1+\ln d} \, \mathrm{d}\mu.
\end{aligned}$$

The last inequality holds since $(1+u)^p \leq pu(1+x)^{p-1} \; \forall \; p \geq 1, u \geq 0$. Further,

$$\begin{aligned}
& a^{\ln d} \int_0^\delta (\ln d) \frac{\mu}{a} \left(1 + \frac{\mu}{a} \right)^{-1+\ln d} \, \mathrm{d}\mu \\
\leq & (\ln d) a^{-1+\ln d} \left(1 + \frac{\delta}{a} \right)^{-1+\ln d} \int_0^\delta \mu \, \mathrm{d}\mu \\
= & \frac{(\ln d) \delta^2 (a+\delta)^{-1+\ln d}}{2}.
\end{aligned}$$

Therefore,

$$\lambda \leq \frac{e (\ln d)^2 \delta^2 (a+d)^{-1+\ln d}}{2} = \frac{\eta_0^2}{2e} y_i^{1-\frac{1}{\ln d}}.$$

□

Lemma E.4. *Consider $z \in \mathbb{R}^d_{\geq 0}$ with $\sum_{i \in C} z_i \geq \frac{1}{d^2}$ and y as defined in Equation E.3. If $\eta_0 \leq \frac{1}{2e^2}$, then*

$$B_{h_d}(z \| y) \leq 2\eta_0^2 \sum_{i \in C} z_i.$$

Proof.

$$B_{h_d}(z\|y) = \frac{e\ln d}{1+\frac{1}{\ln d}}\sum_{i\in[d]}\left(z_i^{1+\frac{1}{\ln d}} - y_i^{1+\frac{1}{\ln d}}\right) + e\ln d\sum_{i\in[d]}(y_i - z_i)y_i^{\frac{1}{\ln d}}.$$

The terms corresponding to $i \notin C$ cancel out. By Lemma E.3 therefore,

$$B_{h_d}(z\|y) \le \frac{\eta_0^2}{2e}\sum_{i\in C} y_i.$$

Partition $C = L \cup (C\setminus L)$, where $L = \{i \in C : z_i \ge \frac{1}{d^3}\}$ is the set of all 'large' coordinates. For all $i \in L$,

$$\begin{aligned}
y_i^{1-\frac{1}{\ln d}} &= z_i\left(1 + \frac{\eta_0}{e z_i^{1/\ln d}\ln d}\right)^{\ln d - 1} \\
&\le z_i \exp\left(\frac{\eta_0}{e z_i^{1/\ln d}}\right) && (1+u \le \exp(u)) \\
&\le z_i \exp\left(\eta_0 e^2\right) && (z_i \ge 1/d^3) \\
&\le z_i\left(1 + 2\eta_0 e^2\right) \le 2z_i. && (\eta_0 \le 1/2e^2)
\end{aligned}$$

Thus, $\sum_{i\in L} y_i^{1-\frac{1}{\ln d}} \le 2\sum_{i\in C} z_i$. We now add those $i \in C \setminus L$:

$$\begin{aligned}
\sum_{i\in C\setminus L} y_i^{1-\frac{1}{\ln d}} &= \sum_{i\in C\setminus L}\left(z_i^{\frac{1}{\ln d}} + \frac{\eta_0}{e\ln d}\right)^{\ln d - 1} \\
&\le \sum_{i\in C\setminus L}\left(\frac{1}{e^3} + \frac{\eta_0}{e\ln d}\right)^{\ln d - 1} && (z_i \le d^3) \\
&= \sum_{i\in C\setminus L}\frac{e}{d^3}\left(1 + \frac{e^2\eta_0}{\ln d}\right)^{\ln d - 1} && (z_i \le d^3) \\
&\le \sum_{i\in C\setminus L}\frac{e}{d^3}\exp\left(\eta_0 e^2\right) \\
&\le d \times \frac{e}{d^3} \times (1 + 2\eta_0 e^2) \le 2e\sum_{i\in C} z_i.
\end{aligned}$$

Therefore, we have

$$B_{h_d}(z\|y) \le \frac{\eta_0^2}{2e} \times 4e \sum_{i\in C} z_i \le 2\eta_0^2 \sum_{i\in C} z_i. \qquad \square$$

Since C is a randomly chosen set of coordinates for iterates of the algorithm, $\sum_{i\in C} z_i$ is a random quantity. Our next lemma establishes concentration on $\sum_{i\in C} z_i$:

Lemma E.5. *Given dimension $d > 0$ and positive integer $S \le d$, consider a random set $C \subseteq [d]$ with $|C| = S$ defined as follows: with probability 1, we have $1 \in C$ and the other $S-1$ elements of C are chosen uniformly at random from among the remaining $d-1$ coordinates. Given $z \in \mathbb{R}^d_{\ge 0}$ such that $\|z\|_\infty = z_1$, consider random variable $U = \sum_{i\in C} z_i$. Then, for all $\beta \ge e$,*

$$\Pr\left(U \ge (4\ln\beta)\max\left\{z_1, \frac{S}{d}\|z\|_1\right\}\right) \le \frac{1}{\beta}.$$

Proof. Denote $C' = C \setminus \{1\}$ and $U' = \sum_{i\in C'} z_i$ so that $U = z_1 + \hat{U}$. For $i \in [2,d]$, define the indicator random variable X_i' for whether $i \in C$. Then $\hat{U} = \sum_{i=2}^d X_i' z_i$ and $\mathbb{E}X_i' = \frac{S-1}{d-1}$. Therefore, $\mathbb{E}\hat{U} = \frac{S-1}{d-1}\sum_{i=2}^d z_i$.

Define $X_i = X_i' - \frac{S-1}{d-1}$. Then (1) $\mathbb{E}X_i = 0$, (2) $\hat{U} - \mathbb{E}\hat{U} = \sum_{i=2}^d X_i z_i$, (3) $|X_i| \le 1 - \frac{S-1}{d-1} \le 1$, and (4) $\mathbb{E}[X_i^2] = \frac{S-1}{d-1}\left(1 - \frac{S-1}{d-1}\right) \le \frac{S-1}{d-1}$. Theorefore, by Lemma E.1,

$$\begin{aligned}
\Pr\left(\hat{U} - \mathbb{E}\hat{U} \ge \delta\right) &= Pr\left(\sum_{i=2}^d z_i X_i \ge \delta\right) \\
&\le \exp\left(-\frac{3\delta^2}{2\delta\max_{i\in[2,d]} z_i + 6\sum_{i=2}^d z_i^2 \mathbb{E}[X_i^2]}\right) \\
&\le \exp\left(-\frac{3\delta^2}{2\delta\max_{i\in[2,d]} z_i + \frac{6(S-1)}{d-1}\sum_{i=2}^d z_i^2}\right) && \left(\mathbb{E}[X_i^2] \le \frac{S-1}{d-1}\right) \\
&\le \exp\left(-\frac{3\delta^2}{2\delta\max_{i\in[2,d]} z_i + \frac{6(S-1)}{d-1}\left(\max_{i\in[2,d]} z_i\right)\sum_{i=2}^d z_i}\right) \\
&\le \exp\left(-\frac{3\delta^2}{2\delta z_1 + 6 z_1 \mathbb{E}U'}\right) && (\|z\|_\infty = z_1)
\end{aligned}$$

$$\leq \exp\left(-\frac{3\delta^2}{2\max\{2\delta z_1, 6z_1\mathbb{E}U'\}}\right)$$
$$= \exp\left(-\min\left\{\frac{3\delta}{4z_1}, \frac{\delta^2}{4z_1\mathbb{E}U'}\right\}\right).$$

Choose $\delta = (2\ln\beta)\max\{z_1, \mathbb{E}U'\} \geq (2\ln\beta)\max\left\{z_1, \sqrt{z_1\mathbb{E}U'}\right\}$. Then $\min\left\{\frac{3\delta}{4z_1}, \frac{\delta^2}{4z_1\mathbb{E}U'}\right\} \geq \min\left\{\frac{3}{2}\ln\beta, \ln\beta\right\} = \ln\beta$. Therefore,

$$\Pr\left(\hat{U} - \mathbb{E}\hat{U} \geq \delta\right) \leq \exp\left(-\min\left\{\frac{3\delta}{4z_1}, \frac{\delta^2}{4z_1\mathbb{E}U'}\right\}\right) \leq \exp(-\ln\beta) = \frac{1}{\beta}.$$

Consequently,

$$\Pr(U \geq z_1 + \mathbb{E}U' + (2\ln\beta)\max\{z_1, \mathbb{E}U'\}) \leq \frac{1}{\beta}.$$

However, since $\beta \geq e$, we have $z_1 + \mathbb{E}U' + (2\ln\beta)\max\{z_1, \mathbb{E}U'\} \leq (4\ln\beta)\max\{z_1, \mathbb{E}U'\}$. Since $\mathbb{E}U' = \frac{S-1}{d-1}\sum_{i=2}^{d} z_i \leq \frac{S}{d}\sum_{i=2}^{d} z_i \leq \frac{S}{d}\|z\|_1$, we get the result. □

We get the following result as a corollary, which also proves that Equation E.2 is true with high probability:

Corollary E.1. *Consider an iterate $z^{(t)} \in \hat{P}$ of OMD with dth block norm and unconstrained update point $y^{(t)}$ as defined in Equation E.3. Then, with probability $\geq 1 - \frac{1}{dT}$,*

$$B_{h_d}(z^{(t)}\|y^{(t)}) \leq 8\eta_0^2\ln(dT)\max\left\{z_1, \frac{S}{d}\|z\|_1\right\}.$$

In particular, with probability $\geq 1 - \frac{1}{d}$, for any iterate $z^{(t)} \in \hat{P}$ of OMD with dth block norm, we have

$$B_{h_d}(z^{(t)}\|y^{(t)}) \leq \frac{K}{4R}.$$

Proof. (First part) First, we show that the conditions of Lemma E.4 are met for $z = z^{(t)}$, so that $B_{h_d}(z^{(t)}\|y^{(t)}) \leq 2\eta_0^2\sum_{i\in C^{(t)}} z_i^{(t)}$.

By symmetry, the minimum L_∞ norm of any point in $\hat{P} = \operatorname{conv}\left(\frac{1}{R}\mathbf{e}_1, \ldots, \frac{1}{R}\mathbf{e}_d, \frac{1}{d}\mathbf{1}_d\right)$

is $\frac{1}{Rd} \geq \frac{1}{d^2}$. Since $1 \in C^{(t)}$ for all t, the algorithm increases the coordinate $z^{(t)}$ at all times t, and therefore by induction on t, we get $\|z^{(t)}\|_\infty = z_1^{(t)}$. Thus, $\sum_{i \in C^{(t)}} z_i^{(t)} \geq z_1^{(t)} \geq \frac{1}{d^2}$. Further, $\eta_0 \leq \sqrt{\frac{e \ln d}{T}} \leq \frac{1}{2e^2}$ for all $T \geq 50 \ln d$. Therefore, by Lemma E.4, we have

$$B_{h_d}(z^{(t)} \| y^{(t)}) \leq 2\eta_0^2 \sum_{i \in C^{(t)}} z_i^{(t)}.$$

By Lemma E.5, for $\beta = \ln(dT)$, we then have that

$$\Pr\left(B_{h_d}(z^{(t)} \| y^{(t)}) \geq (2\eta_0^2) \cdot (4 \ln dT) \max\left\{ z_1^{(t)}, \frac{S}{d} \|z^{(t)}\|_1 \right\} \right) \leq \frac{1}{dT}.$$

(Second part) We have $\eta_0^2 \leq \frac{e \ln d}{T} \leq \frac{4 \ln(dT)}{T}$. Further, since $\hat{P} = \operatorname{conv}\left(\frac{1}{R}\mathbf{e}_1, \ldots, \frac{1}{R}\mathbf{e}_d, \frac{1}{d}\mathbf{1}_d\right)$, we have (1) $z_1^{(t)} \leq \frac{1}{R}$, and (2) $\frac{S}{d}\|z^{(t)}\|_1 \leq \frac{S}{d} \cdot 1 = \frac{d^{1/3}}{d} \leq \frac{1}{d^{1/3}} = \frac{1}{R}$, so that $\max\left\{ z_1^{(t)}, \frac{S}{d}\|z^{(t)}\|_1 \right\} \leq \frac{1}{R}$. Therefore, with probability $\geq 1 - \frac{1}{dT}$, $B_{h_d}(z^{(t)} \| y^{(t)}) \leq \frac{32 \ln^2(dT)}{RT} = \frac{K}{4R}$. Taking a union bound over all $t \in [T]$ implies the result. □

E.4 Proof of Theorem 7.6

Let $\ell^* = \arg\min_{\ell \in [N]} \sum_{t \in [T]} f^{(t)}(X_\ell^{(t)})$, and denote $f_{\ell^*}^{(t)} = f_*^{(t)}$ for all t. Define *weights* $w_\ell^{(t)}$ inductively as follows: $w_\ell^{(1)} = \frac{1}{N}$ for all N, and $w_\ell^{(t+1)} = w_\ell^{(t)} \exp\left(-\frac{\varepsilon(f^{(t)}(X_\ell^{(t)}) - f^{(t)}(X_*^{(t)}))}{\rho} \right)$ for all t, where $X_\ell^{(t)}$ is the iterate of the ℓth mirror map at time t.

Define potential at time t as $\phi^{(t)} = \sum_{\ell \in [N]} w_\ell^{(t)} = \|w^{(t)}\|_1$. Note that $p_\ell^{(t)} \propto w_\ell^{(t)}$ and therefore $p_\ell^{(t)} = \frac{w_\ell^{(t)}}{\|w^{(t)}\|_1} = \frac{w_\ell^{(t)}}{\phi^{(t)}}$. Then,

$$\phi^{(t+1)} = \sum_{\ell \in [N]} w_\ell^{(t+1)} = \sum_{\ell \in [N]} w_\ell^{(t)} \exp\left(-\frac{\varepsilon(f^{(t)}(X_\ell^{(t)}) - f^{(t)}(X_*^{(t)}))}{\rho} \right).$$

Further, $f^{(t)}(X_\ell^{(t)}) - f^{(t)}(X_*^{(t)}) \leq \rho$ by definition and $\varepsilon \in [0, 1]$. Since $\exp(u) \leq 1 + u + u^2$

for all $u \in [-1, 1]$, we get

$$\begin{aligned}
\phi^{(t+1)} &\le \sum_{\ell \in [N]} w_\ell^{(t)} \left(1 - \frac{\varepsilon(f^{(t)}(X_\ell^{(t)}) - f^{(t)}(X_*^{(t)}))}{\rho} + \frac{\varepsilon^2 (f^{(t)}(X_\ell^{(t)}) - f^{(t)}(X_*^{(t)}))^2}{\rho^2}\right) \\
&\le \sum_{\ell \in [N]} w_\ell^{(t)} \left(1 - \frac{\varepsilon(f^{(t)}(X_\ell^{(t)}) - f^{(t)}(X_*^{(t)}))}{\rho} + \varepsilon^2\right) \\
&= (1+\varepsilon^2)\phi^{(t)} - \frac{\varepsilon}{\rho} \sum_\ell w_\ell^{(t)} (f^{(t)}(X_\ell^{(t)}) - f^{(t)}(X_*^{(t)})) \\
&= (1+\varepsilon^2)\phi^{(t)} - \frac{\varepsilon \phi^{(t)}}{\rho} \sum_\ell p_\ell^{(t)} (f^{(t)}(X_\ell^{(t)}) - f^{(t)}(X_*^{(t)})).
\end{aligned}$$

Recall that the algorithm plays $x^{(t)} = \sum_\ell p_\ell^{(t)} X_\ell^{(t)}$. Since $f^{(t)}$ is convex, $f^{(t)}(x^{(t)}) \le \sum_\ell p_\ell^{(t)} f^{(t)}(X_\ell^{(t)})$. Therefore,

$$\phi^{(t+1)} \le \phi^{(t)} \left((1+\varepsilon^2) - \frac{\varepsilon(f^{(t)}(x^{(t)}) - f^{(t)}(X_*^{(t)}))}{\rho}\right).$$

Using $1 + u \le \exp(u)$ for all $u \in \mathbb{R}$,

$$\phi^{(t+1)} \le \phi^{(t)} \exp\left(\varepsilon^2 - \frac{\varepsilon(f^{(t)}(x^{(t)}) - f^{(t)}(X_*^{(t)}))}{\rho}\right).$$

Therefore,

$$\begin{aligned}
\frac{\phi^{(T+1)}}{\phi^{(1)}} &\le \exp\left(T\varepsilon^2 - \frac{\varepsilon \sum_{t \in [T]} (f^{(t)}(x^{(t)}) - f^{(t)}(X_*^{(t)}))}{\rho}\right) \\
&= \exp\left(T\varepsilon^2 - \frac{\varepsilon(\texttt{regret}(T) - \texttt{regret}_*(T))}{\rho}\right).
\end{aligned}$$

Note that $\phi^{(1)} = 1$ and $\phi^{(T+1)} \ge w_*^{(T+1)} = w_*^{(1)} = \frac{1}{N}$, so that $-\ln N \le T\varepsilon^2 - \frac{\texttt{regret}(T) - \texttt{regret}_*(T)}{\rho}$. Rearranging,

$$\texttt{regret}(T) \le \texttt{regret}_*(T) + \rho\left(\frac{\ln N}{\varepsilon} + T\varepsilon\right).$$

By assumption, $T \geq \ln N$. Choose $\varepsilon = \sqrt{\frac{\ln N}{T}} \leq 1$ to get

$$\texttt{regret}(T) \leq \texttt{regret}_*(T) + 2\rho\sqrt{\ln N}\sqrt{T}. \qquad \square$$

REFERENCES

[1] Aristotle, *Nicomachean Ethics*, trans. by T. Irwin. Hackett Publishing, 1985, Originally published in the 4th century BCE.

[2] J. Rawls, "A theory of justice," in *Applied ethics*, Routledge, 2017, pp. 21–29.

[3] J. Lamont and C. Favor, "Distributive Justice," in *The Stanford Encyclopedia of Philosophy*, E. N. Zalta, Ed., Winter 2017, Metaphysics Research Lab, Stanford University, 2017.

[4] D. K. Foley, *Resource allocation and the public sector*. Yale University, 1966.

[5] J. F. Nash, "The Bargaining Problem," *Econometrica*, vol. 18, no. 2, pp. 155–162, 1950, Publisher: [Wiley, Econometric Society].

[6] A. Sen, *Inequality reexamined*. Oxford university press, 1992.

[7] J. Kleinberg, "An Impossibility Theorem for Clustering," *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, 2002.

[8] A. Kumar and J. Kleinberg, "Fairness measures for resource allocation," in *Symposium on Foundations of Computer Science (FOCS)*, ISSN: 0272-5428, Nov. 2000, pp. 75–85.

[9] A. Goel and A. Meyerson, "Simultaneous Optimization via Approximate Majorization for Concave Profits or Convex Costs," *Algorithmica*, vol. 44, no. 4, pp. 301–323, Apr. 2006.

[10] J. Kleinberg, "Inherent Trade-Offs in Algorithmic Fairness," in *2018 ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, Jun. 2018, p. 40, ISBN: 978-1-4503-5846-0.

[11] A. Chouldechova, "Fair Prediction with Disparate Impact: A Study of Bias in Recidivism Prediction Instruments," vol. 5, no. 2, pp. 153–163, Jun. 2017.

[12] V. V. Vazirani, *Approximation Algorithms*. Berlin, Heidelberg: Springer, 2003.

[13] D. P. Williamson and D. B. Shmoys, *The Design of Approximation Algorithms*. Cambridge: Cambridge University Press, 2010.

[14] G. W. Evans, "An Overview of Techniques for Solving Multiobjective Mathematical Programs," *Management Science*, vol. 30, no. 11, pp. 1268–1282, Nov. 1984.

[15] M. Masin and Y. Bukchin, "Diversity Maximization Approach for Multiobjective Optimization," *Operations Research*, vol. 56, no. 2, pp. 411–424, Apr. 2008.

[16] F. Grandoni, R. Ravi, M. Singh, and R. Zenklusen, "New approaches to multi-objective optimization," *Mathematical Programming*, vol. 146, no. 1, pp. 525–554, Aug. 2014.

[17] Y. Tian *et al.*, "Evolutionary Large-Scale Multi-Objective Optimization: A Survey," *ACM Comput. Surv.*, vol. 54, no. 8, 174:1–174:34, Oct. 2021.

[18] A. Herzel, S. Ruzika, and C. Thielen, "Approximation Methods for Multiobjective Optimization Problems: A Survey," *INFORMS Journal on Computing*, vol. 33, no. 4, pp. 1284–1299, Oct. 2021.

[19] C. H. Papadimitriou and M. Yannakakis, "Multiobjective query optimization," in *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, ser. PODS, May 2001, pp. 52–59, ISBN: 978-1-58113-361-5.

[20] D. Chakrabarty and C. Swamy, "Approximation algorithms for minimum norm and ordered optimization problems," in *Symposium on Theory of Computing (STOC) 2019*, Jun. 2019, pp. 126–137, ISBN: 978-1-4503-6705-9.

[21] S. Verma *et al.*, "Increasing impact of mobile health programs: Saheli for maternal and child care," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 13, pp. 15 594–15 602, Jul. 2024.

[22] ARMMAN, *Armman: Advancing reduction in mortality and morbidity of mothers, children, and neonates*, 2024.

[23] D. Golovin, A. Gupta, A. Kumar, and K. Tangwongsan, "All-Norms and All-Lp-Norms Approximation Algorithms," *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, 2008.

[24] A. K. Chandra and C. K. Wong, "Worst-Case Analysis of a Placement Algorithm Related to Storage Allocation," *SIAM Journal on Computing*, vol. 4, no. 3, pp. 249–263, Sep. 1975.

[25] R. L. Graham, "Bounds for certain multiprocessing anomalies," *The Bell System Technical Journal*, vol. 45, no. 9, pp. 1563–1581, 1966.

[26] A. Blum, P. Chalasani, D. Coppersmith, B. Pulleyblank, P. Raghavan, and M. Sudan, "The minimum latency problem," in *Symposium on Theory of Computing (STOC)*, 1994, pp. 163–171, ISBN: 978-0-89791-663-9.

[27] M. Farhadi, A. Toriello, and P. Tetali, "The Traveling Firefighter Problem," in *Applied and Computational Discrete Algorithms (ACDA)*, 2021, pp. 205–216.

[28] Y. Azar and S. Taub, "All-Norm Approximation for Scheduling on Identical Machines," in *Algorithm Theory - SWAT*, 2004, pp. 298–310.

[29] C. Papadimitriou and M. Yannakakis, "On the approximability of trade-offs and optimal access of Web sources," in *Proceedings 41st Annual Symposium on Foundations of Computer Science (FOCS)*, ISSN: 0272-5428, Nov. 2000, pp. 86–92.

[30] C. Bazgan, S. Ruzika, C. Thielen, and D. Vanderpooten, "The Power of the Weighted Sum Scalarization for Approximating Multiobjective Optimization Problems," *Theory of Computing Systems*, vol. 66, no. 1, pp. 395–415, Feb. 2022, arXiv:1908.01181 [cs].

[31] C. Glaßer, C. Reitwießner, H. Schmitz, and M. Witek, "Approximability and Hardness in Multi-objective Optimization," in *Programs, Proofs, Processes*, F. Ferreira, B. Löwe, E. Mayordomo, and L. Mendes Gomes, Eds., Berlin, Heidelberg: Springer, 2010, pp. 180–189, ISBN: 9783642139628.

[32] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma, "Fairness-Aware Classifier with Prejudice Remover Regularizer," in *Machine Learning and Knowledge Discovery in Databases*, P. A. Flach, T. De Bie, and N. Cristianini, Eds., ser. Lecture Notes in Computer Science, Springer, 2012, pp. 35–50, ISBN: 978-3-642-33486-3.

[33] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork, "Learning Fair Representations," in *Proceedings of the 30th International Conference on Machine Learning*, PMLR, May 2013, pp. 325–333.

[34] A. Ageev, Y. Ye, and J. Zhang, "Improved combinatorial approximation algorithms for the k-level facility location problem," *SIAM Journal on Discrete Mathematics*, vol. 18, no. 1, pp. 207–217, 2004.

[35] M. Ghadiri, S. Samadi, and S. Vempala, "Socially fair k-means clustering," in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, ser. FAccT '21, Virtual Event, Canada: Association for Computing Machinery, 2021, pp. 438–448, ISBN: 9781450383097.

[36] E. Chlamtáč, Y. Makarychev, and A. Vakilian, "Approximating fair clustering with cascaded norm objectives," in *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2022, pp. 2664–2683. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611977073.104.

[37] D. Chakrabarty and C. Swamy, "Interpolating between k-Median and k-Center: Approximation algorithms for ordered k-Median," in *International Colloquium on*

Automata, Languages, and Programming (ICALP), ISSN: 1868-8969, vol. 107, 2018, 29:1–29:14.

[38] K. Patton, M. Russo, and S. Singla, "Submodular Norms with Applications To Online Facility Location and Stochastic Probing," in *APPROX*, vol. 275, 2023, 23:1–23:22, ISBN: 978-3-95977-296-9.

[39] S. Barman, U. Bhaskar, A. Krishna, and R. G. Sundaram, "Tight approximation algorithms for p-mean welfare under subadditive valuations," *arXiv preprint arXiv:2005.07370*, 2020.

[40] S. Verma, N. Boehmer, L. Kong, and M. Tambe, *Balancing act: Prioritization strategies for LLM-designed restless bandit rewards*, 2024. arXiv: 2408.12112 [cs.LG].

[41] Z. Fan, N. Peng, M. Tian, and B. Fain, "Welfare and fairness in multi-objective reinforcement learning," *arXiv preprint arXiv:2212.01382*, 2022.

[42] C. Cousins, K. Asadi, E. Lobo, and M. Littman, "On welfare-centric fair reinforcement learning," *Reinforcement Learning Journal*, vol. 3, pp. 1124–1137, 2024.

[43] J. Kwon and V. Perchet, "Gains and losses are fundamentally different in regret minimization: The sparse case," *Journal of Machine Learning Research*, vol. 17, no. 227, pp. 1–32, 2016.

[44] S. Arora, E. Hazan, and S. Kale, "The multiplicative weights update method: A meta-algorithm and applications," *Theory of computing*, vol. 8, no. 1, pp. 121–164, 2012.

[45] T. Erven, W. M. Koolen, S. Rooij, and P. Grünwald, "Adaptive hedge," *Advances in Neural Information Processing Systems*, vol. 24, 2011.

[46] T. Van Erven and W. M. Koolen, "Metagrad: Multiple learning rates in online learning," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[47] F. Orabona and D. Pál, "Coin betting and parameter-free online learning," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[48] M. Drygala, S. Lattanzi, A. Maggiori, M. Stouras, O. Svensson, and S. Vassilvitskii, "Data-Driven Solution Portfolios," in *16th Innovations in Theoretical Computer Science Conference (ITCS 2025)*, R. Meka, Ed., ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 325, Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025, 46:1–46:15, ISBN: 9783959773614.

[49] A. Agarwal, O. Dekel, and L. Xiao, "Optimal algorithms for online convex optimization with multi-point bandit feedback.," in *COLT*, 2010, pp. 28–40.

[50] E. Elkind, J. Lang, and A. Saffidine, "Condorcet winning sets," *Social Choice and Welfare*, vol. 44, no. 3, pp. 493–517, 2015.

[51] M. Charikar, A. Lassota, P. Ramakrishnan, A. Vetta, and K. Wang, "Six Candidates Suffice to Win a Voter Majority," in *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, ser. STOC '25, New York, NY, USA: Association for Computing Machinery, Jun. 2025, pp. 1590–1601, ISBN: 9798400715105.

[52] R. K. Ahuja and J. B. Orlin, "Inverse optimization," *Operations research*, vol. 49, no. 5, pp. 771–783, 2001.

[53] A. Blum, J. Jackson, T. Sandholm, and M. Zinkevich, "Preference elicitation and query learning," *Journal of Machine Learning Research*, vol. 5, no. Jun, pp. 649–667, 2004.

[54] G. H. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities*. Cambridge: Cambridge University Press, 1952.

[55] D. S. Johnson, "Approximation algorithms for combinatorial problems," in *Symposium on Theory of Computing (STOC)*, Apr. 1973, pp. 38–49.

[56] U. Feige, L. Lovász, and P. Tetali, "Approximating Min Sum Set Cover," *Algorithmica*, vol. 40, no. 4, pp. 219–234, Dec. 2004.

[57] E. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization* (Wiley Series in Discrete Mathematics & Optimization). Hoboken, New Jersey, USA: John Wiley and Sons, 1991.

[58] M. Goemans and J. Kleinberg, "An improved approximation ratio for the minimum latency problem," *Mathematical Programming*, vol. 82, no. 1, pp. 111–124, Jun. 1998.

[59] S. Gupta, J. Moondra, and M. Singh, "Which Lp norm is the fairest? Approximations for fair facility location across all "p"," in *Economics and Computation (EC) 2023*, Jul. 2023, p. 817, ISBN: 9798400701047.

[60] J. Bentham, *The principles of morals and legislation*. Clarendon Press, 1879.

[61] M. Fleurbaey and F. Maniquet, *A theory of fairness and social welfare*. Cambridge University Press, Jan. 2010.

[62] D. Golovin, A. Gupta, A. Kumar, and K. Tangwongsan, "All-Norms and All-L_p-Norms Approximation Algorithms," in *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, 2008, pp. 199–210, ISBN: 978-3-939897-08-8.

[63] S. Gupta, A. Jalan, G. Ranade, H. Yang, and S. Zhuang, "Too many fairness metrics: Is there a solution?" In *Ethics of Data Science Conference EDSC*, 2020.

[64] V. (Chen and J. N. Hooker, "A Just Approach Balancing Rawlsian Leximax Fairness and Utilitarianism," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, ser. AIES '20, New York, NY, USA: Association for Computing Machinery, Feb. 2020, pp. 221–227, ISBN: 9781450371100.

[65] D. Chakrabarty and C. Swamy, "Approximation algorithms for minimum norm and ordered optimization problems," in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, 2019, pp. 126–137.

[66] R. Abelson and R. Robbins, *The powerful companies driving local drugstores out of business*, Oct. 2024.

[67] W. B. Group, *Multidimensional poverty measure*, Dec. 2023.

[68] P. Dutko, M. Ploeg, and T. Farrigan, "Characteristics and influential factors of food deserts," *Economic Research Report*, vol. 140, 2012.

[69] C. Calderon and L. Servén, "The effects of infrastructure development on growth and income distribution," *Social Science Research Network*, 2004.

[70] S. John *et al.*, "Balancing mission and margins: What makes healthy community food stores successful," *Int J Environ Res Public Health*, 2022.

[71] D. Shmoys, É. Tardos, and K. Aardal, "Approximation algorithms for facility location problems (extended abstract)," in *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, ser. STOC '97, El Paso, Texas, USA, 1997, pp. 265–274, ISBN: 0897918886.

[72] D. Hochbaum, "Heuristics for the fixed cost median problem," *Mathematical programming*, vol. 22, pp. 148–162, 1982.

[73] M. Charikar and S. Guha, "Improved combinatorial algorithms for the facility location and k-median problems," in *In Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, 1999, pp. 378–388.

[74] S. Li, "A 1.488 approximation algorithm for the uncapacitated facility location problem," *Information and Computation*, vol. 222, pp. 45–58, 2013, 38th International Colloquium on Automata, Languages and Programming (ICALP 2011).

[75] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness," in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ser. ITCS, Cambridge, Massachusetts: Association for Computing Machinery, 2012, pp. 214–226, ISBN: 9781450311151.

[76] F. Chierichetti, R. Kumar, S. Lattanzi, and S. Vassilvitskii, "Fair Clustering Through Fairlets," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[77] L. Celis, A. Mehrotra, and N. Vishnoi, "Interventions for ranking in the presence of implicit bias," in *Proceedings of the 2020 conference on Fairness, Accountability, and Transparency*, 2020, pp. 369–380.

[78] J. Byrka, K. Fleszar, B. Rybicki, and J. Spoerhase, "Bi-Factor Approximation Algorithms for Hard Capacitated k-Median Problems," in *Proceedings of the 2015 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, ser. Proceedings, Society for Industrial and Applied Mathematics, Dec. 2014, pp. 722–736, ISBN: 9781611973747.

[79] M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt, "Bicriteria Network Design Problems," *Journal of Algorithms*, vol. 28, no. 1, pp. 142–171, Jul. 1998.

[80] M. Sviridenko, "An improved approximation algorithm for the metric uncapacitated facility location problem," in *International Conference on Integer Programming and Combinatorial Optimization*, Springer, 2002, pp. 240–257.

[81] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of optimal queuing network control," *Mathematics of Operations Research*, vol. 24, no. 2, pp. 293–305, 1999.

[82] G. Cornuejols, G. Nemhauser, and L. Wolsey, "The uncapicitated facility location problem," Cornell University Operations Research and Industrial Engineering, Tech. Rep., 1983.

[83] S. Guha and S. Khuller, "Greedy strikes back: Improved facility location algorithms," in *Journal of Algorithms*, 1998, pp. 649–657.

[84] M. Korupolu, C. Plaxton, and R. Rajaraman, "Analysis of a local search heuristic for facility location problems," in *In Proceedings of the 9th Annual ACM-SIAM Symposium On Discrete Algorithms*, 1998, pp. 1–10.

[85] K. Jain, M. Mahdian, and A. Saberi, "A new greedy approach for facility location problems," in *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, ser. STOC, Montreal, Quebec, Canada, 2002, pp. 731–740, ISBN: 1581134959.

[86] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani, "Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP," *J. ACM*, vol. 50, no. 6, pp. 795–824, 2003.

[87] F. Chudak and D. Shmoys, "Improved approximation algorithms for the uncapacitated facility location problem," *SIAM Journal on Computing*, vol. 33, no. 1, pp. 1–25, 2003.

[88] J. Byrka, "An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem," in *APPROX '07/RANDOM '07*, Princeton, NJ, USA, 2007, pp. 29–43, ISBN: 9783540742074.

[89] J. Lin and J. Vitter, "Approximation algorithms for geometric median problems," *Information Processing Letters*, vol. 44, no. 5, pp. 245–249, 1992.

[90] S. Arora, P. Raghavan, and S. Rao, "Approximation schemes for euclidean k-medians and related problems," in *Proceedings of the thirtieth annual ACM Symposium on Theory of Computing*, 1998, pp. 106–113.

[91] M. Charikar, S. Guha, É. Tardos, and D. Shmoys, "A constant-factor approximation algorithm for the k-median problem," in *Proceedings of the thirty-first annual ACM Symposium on Theory of Computing*, 1999, pp. 1–10.

[92] S. Li and O. Svensson, "Approximating k-median via pseudo-approximation," in *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, ser. STOC '13, Palo Alto, California, USA, 2013, pp. 901–910, ISBN: 9781450320290.

[93] V. Cohen-Addad, H. Esfandiari, V. Mirrokni, and S. Narayanan, "Improved approximations for euclidean k-means and k-median, via nested quasi-independent sets," in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, 2022, pp. 1621–1628.

[94] D. Chakrabarty, M. Negahbani, and A. Sarkar, "Approximation algorithms for continuous clustering and facility location problems," *30th Annual European Symposium on Algorithms*, vol. 244, 2022.

[95] M. Truelove, "Measurement of spatial equity," *Environment and Planning C: Government and Policy*, vol. 11, no. 1, pp. 19–34, 1993. eprint: https://doi.org/10.1068/c110019.

[96] E. Talen, "Visualizing fairness: Equity maps for planners," *Journal of the American Planning Association*, vol. 64, no. 1, pp. 22–38, 1998. eprint: https://doi.org/10.1080/01944369808975954.

[97] C. Jung, S. Kannan, and N. Lutz, *A center in your neighborhood: Fairness in facility location*, 2019. arXiv: 1908.09041 [cs.DS].

[98] M. Abbasi, A. Bhaskara, and S. Venkatasubramanian, "Fair clustering via equitable group representations," in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, ser. FAccT, Virtual Event, Canada, 2021, ISBN: 9781450383097.

[99] S. M. Lee and L. J. Moore, "Multi-Criteria School Busing Models," *Management Science*, vol. 23, no. 7, pp. 703–715, Mar. 1977.

[100] D. Bertsimas, A. Delarue, and S. Martin, "Optimizing schools' start time and bus routes," *Proceedings of the National Academy of Sciences*, vol. 116, no. 13, pp. 5943–5948, Mar. 2019.

[101] D. Bertsimas, A. Delarue, W. Eger, J. Hanlon, and S. Martin, "Bus Routing Optimization Helps Boston Public Schools Design Better Policies," *INFORMS Journal on Applied Analytics*, vol. 50, no. 1, pp. 37–49, Jan. 2020.

[102] F. J. Fabozzi, H. M. Markowitz, and F. Gupta, "Portfolio selection," *Handbook of finance*, vol. 2, pp. 3–13, 2008.

[103] C. Papahristodoulou and E. Dotzauer, "Optimal portfolios using linear programming models," *Journal of the Operational Research Society*, vol. 55, no. 11, pp. 1169–1177, Nov. 2004.

[104] S. Esmaeili, B. Brubach, A. Srinivasan, and J. Dickerson, "Fair Clustering Under a Bounded Cost," in *Advances in Neural Information Processing Systems*, vol. 34, Curran Associates, Inc., 2021, pp. 14 345–14 357.

[105] R. Jiang and Y. Guan, "Risk-Averse Two-Stage Stochastic Program with Distributional Ambiguity," *Operations Research*, vol. 66, no. 5, pp. 1390–1405, Oct. 2018.

[106] D. B. Shmoys and E. Tardos, "An approximation algorithm for the generalized assignment problem," *Mathematical Programming*, vol. 62, no. 1, pp. 461–474, Feb. 1993.

[107] USDA, *USDA ERS - Go to the Atlas*, 2023.

[108] S. Gupta, J. Moondra, and M. Singh, "Balancing Notions of Equity: Trade-offs Between Fair Portfolio Sizes and Achievable Guarantees," *Proceedings of the 2025*

Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1136–1165, Jan. 2025.

[109] G. S. Nikolić, B. R. Dimitrijević, T. R. Nikolić, and M. K. Stojcev, "A Survey of Three Types of Processing Units: CPU, GPU and TPU," Jun. 2022, pp. 1–6.

[110] V. Manshadi and S. Rodilitz, "Online Policies for Efficient Volunteer Crowdsourcing," in *Economics and Computation (EC) 2020*, Jul. 2020, pp. 315–316.

[111] M. Locke, A. Ellis, and J. D. Smith, "Hold on to what you've got: The volunteer retention literature," *Voluntary Action*, vol. 5, no. 3, pp. 81–99, 2003.

[112] M. Knapp, V. Koutsogeorgopoulou, and J. D. Smith, *Who volunteers and why?: The key factors which determine volunteering*. UK: Volunteer Centre UK, 1995.

[113] R. O. Winder, "Partitions of n-space by hyperplanes," *SIAM Journal on Applied Mathematics*, vol. 14, no. 4, pp. 811–818, 1966.

[114] W. Wang, B. Liang, and B. Li, "Multi-Resource Fair Allocation in Heterogeneous Cloud Computing Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 10, pp. 2822–2835, 2015.

[115] U. Feige, "A threshold of ln n for approximating set cover," *Journal of the ACM*, vol. 45, no. 4, pp. 634–652, Jul. 1998.

[116] R. L. Graham, "Bounds on Multiprocessing Timing Anomalies," *SIAM Journal on Applied Mathematics*, vol. 17, no. 2, pp. 416–429, Mar. 1969.

[117] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan, "Algorithms for facility location problems with outliers," *Symposium on Discrete Algorithms (SODA)*, pp. 642–651, 2001.

[118] C. W. Kim *et al.*, "Navigating the Social Welfare Frontier: Portfolios for Multi-objective Reinforcement Learning," in *International Conference on Machine Learning (ICML)*, arXiv:2502.09724 [cs], Jul. 2025.

[119] J. Perez, C. Germain-Renaud, B. Kégl, and C. Loomis, "Responsive elastic computing," in *Proceedings of the 6th International Conference Industry Session on Grids Meets Autonomic Computing*, ser. GMAC '09, Barcelona, Spain: Association for Computing Machinery, 2009, pp. 55–64.

[120] H. Hao, C. Xu, W. Zhang, S. Yang, and G.-M. Muntean, "Computing offloading with fairness guarantee: A deep reinforcement learning method," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 10, pp. 6117–6130, 2023.

[121] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-uav enabled wireless networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109–2121, 2018.

[122] J. Chen, Y. Wang, and T. Lan, "Bringing fairness to actor-critic reinforcement learning for network utility optimization," in *IEEE Conference on Computer Communications*, 2021, pp. 1–10.

[123] S. Chakraborty *et al.*, "MaxMin-RLHF: Alignment with diverse human preferences," in *Proceedings of the 41st International Conference on Machine Learning*, vol. 235, PMLR, 2024, pp. 6116–6135.

[124] H. Zhong, Z. Deng, W. J. Su, Z. S. Wu, and L. Zhang, *Provable multi-party reinforcement learning with diverse human feedback*, 2024. arXiv: 2403.05006 [cs.LG].

[125] C. Park, M. Liu, D. Kong, K. Zhang, and A. Ozdaglar, *RLHF from heterogeneous feedback via personalization and preference aggregation*, 2024. arXiv: 2405.00254 [cs.AI].

[126] G. Yu, U. Siddique, and P. Weng, "Fair deep reinforcement learning with generalized gini welfare functions," in *Autonomous Agents and Multiagent Systems. Best and Visionary Papers*, Cham: Springer Nature Switzerland, 2024, pp. 3–29.

[127] K. W. S. Roberts, "Interpersonal Comparability and Social Choice Theory," *The Review of Economic Studies*, vol. 47, no. 2, pp. 421–439, Jan. 1980.

[128] H. Moulin, *Fair Division and Collective Welfare*. The MIT Press, Jan. 2003.

[129] C. Cousins, "Revisiting fair-pac learning and the axioms of cardinal welfare," in *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, F. Ruiz, J. Dy, and J.-W. van de Meent, Eds., ser. Proceedings of Machine Learning Research, vol. 206, 2023, pp. 6422–6442.

[130] K. S. Pardeshi, I. Shapira, A. D. Procaccia, and A. Singh, "Learning social welfare functions," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[131] C. F. Hayes *et al.*, "A practical guide to multi-objective reinforcement learning and planning," *Autonomous Agents and Multi-Agent Systems*, vol. 36, no. 1, p. 26, Apr. 2022.

[132] P. Whittle, "Restless bandits: Activity allocation in a changing world," *Journal of Applied Probability*, vol. 25, pp. 287–298, 1988.

[133] P. Bullen, *Handbook of Means and Their Inequalities* (Mathematics and Its Applications), 2nd. Springer Netherlands, 2003.

[134] M. Agarwal, V. Aggarwal, and T. Lan, "Multi-objective reinforcement learning with non-linear scalarization," in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, ser. (AAMAS), Virtual Event, New Zealand: International Foundation for Autonomous Agents and Multiagent Systems, 2022, pp. 9–17.

[135] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *J. Artif. Int. Res.*, vol. 48, no. 1, pp. 67–113, Oct. 2013.

[136] R. Rădulescu, P. Mannion, D. M. Roijers, and A. Nowé, "Multi-objective multi-agent decision making: A utility-based analysis and survey," *Autonomous Agents and Multi-Agent Systems*, vol. 34, no. 1, p. 10, Dec. 2019.

[137] T. G. Dietterich, "Hierarchical reinforcement learning with the maxq value function decomposition," *Journal of artificial intelligence research*, vol. 13, pp. 227–303, 2000.

[138] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ser. ICML'03, Washington, DC, USA: AAAI Press, Aug. 2003, pp. 928–935, ISBN: 978-1-57735-189-4.

[139] S. Shalev-Shwartz, *Online Learning and Online Convex Optimization*. Foundations and Trends in Machine Learning, 2012.

[140] E. Hazan, "Introduction to Online Convex Optimization," *Foundations and Trends® in Optimization*, vol. 2, no. 3-4, pp. 157–325, Aug. 2016.

[141] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization*, 2001.

[142] J. C. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari, "Composite objective mirror descent.," in *COLT*, vol. 10, 2010, pp. 14–26.

[143] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization.," *Journal of machine learning research*, vol. 12, no. 7, 2011.

[144] A. Rakhlin, K. Sridharan, and A. B. Tsybakov, "Empirical entropy, minimax regret and minimax risk," 2017.

[145] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization.," *Journal of machine learning research*, vol. 12, no. 7, 2011.

[146] R. P. Dilworth, "A decomposition theorem for partially ordered sets," in *The Dilworth Theorems: Selected Papers of Robert P. Dilworth*, Springer, 1990, pp. 7–12.

[147] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge: Cambridge University Press, 2004.

[148] R. Yang, X. Sun, and K. Narasimhan, "A generalized algorithm for multi-objective reinforcement learning and policy adaptation," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

[149] L. N. Alegre, D. M. Roijers, A. Nowé, A. L. C. Bazzan, and B. C. da Silva, "Sample-efficient multi-objective learning via generalized policy improvement prioritization," in *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2023.

[150] M. Reymond, E. Bargiacchi, and A. Nowé, "Pareto conditioned networks," in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, ser. (AAMAS), Virtual Event, New Zealand: International Foundation for Autonomous Agents and Multiagent Systems, 2022, pp. 1110–1118.

[151] S. Verma *et al.*, "Expanding impact of mobile health programs: Saheli for maternal and child care," *AI Magazine*, vol. 44, no. 4, pp. 363–376, 2023.

[152] P. Bertail and S. Clémençon, "Bernstein-type exponential inequalities in survey sampling: Conditional poisson sampling schemes," *Bernoulli*, vol. 25, no. 4B, pp. 3527–3554, 2019.

[153] K. Joag-Dev and F. Proschan, "Negative association of random variables with applications," *The Annals of Statistics*, pp. 286–295, 1983.

VITA

Jai Moondra is a discrete optimization researcher, occasional cook, and longtime poetry fan. While studying computer science in college, he became interested in both the theory and practice of algorithm design, and has continued chasing problems at the intersection of algorithms and optimization.