Navigating the Social Welfare Frontier: Portfolios for Multi-objective Reinforcement Learning

Cheol Woo Kim [*1] **Jai Moondra** [*2] **Shresth Verma** [1] **Madeleine Pollack** [3] **Lingkai Kong** [1] **Milind Tambe** [1 4] **Swati Gupta** [3]

Abstract

In many real-world applications of reinforcement learning (RL), deployed policies have varied impacts on different stakeholders, creating challenges in reaching consensus on how to effectively aggregate their preferences. Generalized p-means form a widely used class of social welfare functions for this purpose, with broad applications in fair resource allocation, AI alignment, and decision-making. This class includes well-known welfare functions such as Egalitarian, Nash, and Utilitarian welfare. However, selecting the appropriate social welfare function is challenging for decision-makers, as the structure and outcomes of optimal policies can be highly sensitive to the choice of p. To address this challenge, we study the concept of an α-approximate portfolio in RL, a set of policies that are approximately optimal across the family of generalized p-means for all $p \leq 1$. We propose algorithms to compute such portfolios and provide theoretical guarantees on the trade-offs among approximation factor, portfolio size, and computational efficiency. Experimental results on synthetic and real-world datasets demonstrate the effectiveness of our approach in summarizing the policy space induced by varying p values, empowering decision-makers to navigate this landscape more effectively.

1. Introduction

In this paper, 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, which is often modeled using multi-objective reinforcement learning (MORL), arises in many RL applications, such as fair resource allocation in healthcare (Verma et al., 2024a), cloud computing (Perez et al., 2009; Hao et al., 2023) and communication networks (Wu et al., 2018; Chen et al., 2021). Recently, with the rise of large language models (LLMs), reinforcement learning from human feedback (RLHF) techniques that reflect the preferences of heterogeneous individuals have also been explored (Chakraborty et al., 2024; Zhong et al., 2024; Park et al., 2024).

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 (Yu et al., 2024; Cousins et al., 2024; Verma et al., 2024a; Fan et al., 2023; Zhong et al., 2024; Park et al., 2024; Chakraborty et al., 2024). 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.

In this work, we focus on a class of social welfare functions known as generalized p-means, a widely used class of social welfare functions in algorithmic 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 (Roberts, 1980; Moulin, 2003; Cousins, 2023; Pardeshi et al., 2024).

In practice, the right choice of p is often unclear in advance, and the decision-maker must understand how policies vary with p in order 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

[*]Equal contribution [1]Harvard University, Cambridge, USA [2]Georgia Institute of Technology, Atlanta, USA [3]Massachusetts Institute of Technology, Cambridge, USA [4]Google Deepmind. Correspondence to: Cheol Woo Kim <cwkim@seas.harvard.edu>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

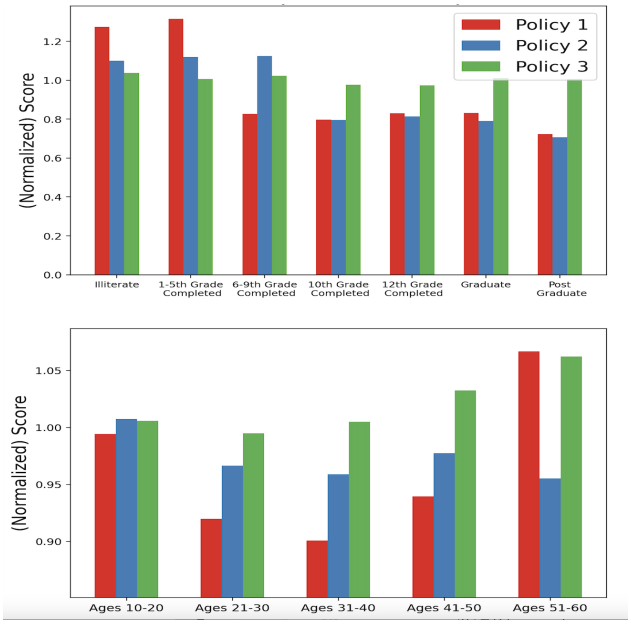


Figure 1. Portfolio of three policies for the healthcare intervention setting (see Section 5 for details), obtained using Algorithm 1 with $\alpha = 0.60$. The bar plots show total rewards induced by each policy across different education (top) and age (bottom) brackets. A "score" of 1.0 corresponds to a baseline policy (not shown in the figure) used for comparison. The three policies impact various education and age brackets differently, offering a theoretically sound and diverse set of options for decision-makers.

function and hence a fixed value of p is given (Hayes et al., 2022).

To address this challenge, we propose a method to compute a small yet representative set of policies 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.

The portfolios provide a structured summary of approximately optimal policies, allowing decision-makers to navigate the implications of different p values efficiently. The decision-maker can review the impact of the choice of p on the structure of policies, on relevant dimensions of interest, and make an informed choice about which policy to deploy.

1.1. Summary of Contributions

In this paper, 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 1) to compute a finite portfolio of policies that is α-approximate for the generalized p-means objectives for any value of $p \leq 1$. We provide theoretical guarantees, including an upper bound on the portfolio size and the number of MDPs solved to optimality, both expressed in terms of the approximation factor α.

2. We introduce a lightweight heuristic BUDGET-CONSTRAINEDPORTFOLIO (Algorithm 3) that reduces computational costs compared to Algorithm p-MEANPORTFOLIO while maintaining high-quality portfolio generation.

3. We theoretically show that the search for an optimal policy for a given p-mean can be efficiently warm-started using the optimal policy for a different p-mean.

4. We evaluate our approach on three different domains, 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 BUDGET-CONSTRAINEDPORTFOLIO constructs portfolios that closely match those produced by Algorithm p-MEANPORTFOLIO, 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. However, this work shifts the focus toward constructing a small set of policies with theoretical guarantees. This approach better captures the trade-offs within the actionable policy space, rather than relying on modeling choices on how these objectives should be formulated.

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 (Verma et al., 2024a). This setting can be captured by restless multi-armed bandits (RMABs) (Whittle, 1988), 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 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 5.

2. Related Work

Social Welfare Function and RL. In RL with multiple stakeholders, various social welfare functions have been explored, including generalized Gini-Welfare (Yu et al., 2024; Cousins et al., 2024), p-means (Verma et al., 2024a; Fan et al., 2023; Cousins et al., 2024), proportional-fairness (Ju et al., 2024), Nash welfare (Mandal and Gan, 2023), among others. (Alamdari et al., 2024) investigated ordinal social welfare functions rather than cardinal ones.

A related line of work has emerged in RLHF for LLMs, where social choice theory has been used to aggregate multiple reward models. For example, approaches leveraging Nash welfare (Zhong et al., 2024), α-fairness (Park et al., 2024), and Egalitarian welfare (Chakraborty et al., 2024) have been proposed. However, in both RLHF and broader multi-stakeholder RL, existing works typically assume that a fixed social welfare function is given and focus on computing the corresponding optimal policy (Hayes et al., 2022). As a result, these methods do not offer guidance for settings where the appropriate choice of social welfare function is unclear in advance.

Portfolios in MORL. In the broader MORL literature, the concept of portfolios of policies is well-established. A common approach is to compute solutions that approximate the Pareto front (Parisi et al., 2014; Van Moffaert and Nowé, 2014; Rădulescu et al., 2019; Hayes et al., 2022). While the Pareto front provides a characterization of trade-offs between conflicting objectives, it does not assume a specific set of social welfare functions. This generality, while powerful, presents challenges in practical multi-stakeholder settings. That is, the Pareto front does not directly correspond to any specific notion of welfare or utility, making it difficult to interpret the societal implications of these policies. Furthermore, without an explicit scalarization function to aggregate preferences, decision-makers must choose among Pareto-efficient policies without clear guidance on how to weigh the trade-offs, which is often critical in real-world applications where diverse preferences must be aggregated into a single deployable policy. Additionally, the Pareto front is typically large, making it prohibitively expensive to compute in practice (Hayes et al., 2022).

A more refined notion of portfolio similar to the one we study is the concept of convex coverage sets. However, this concept is limited to weighted linear combinations of reward functions (Roijers et al., 2013) and does not account for $p-$means. As a result, existing MORL methods, whether based on the Pareto frontier or weighted linear combinations, do not provide guarantees with respect to p-mean welfare functions. We include additional discussion comparing our method with several well-known MORL algorithms proposed in (Yang et al., 2019; Reymond et al., 2022; Alegre et al., 2023; Yu et al., 2024) in Appendix C.

Portfolios in Optimization. In the optimization literature, (Drygala et al., 2024) studied portfolios with stochastic guarantees for fixed objectives. Recent work in approximation algorithms also explores the notions of small-sized portfolios that approximately optimize a class of social welfare functions (Gupta et al., 2023; 2025; Goel and Meyerson, 2006; Golovin et al., 2008; Chakrabarty and Swamy, 2019), by exploiting the combinatorial properties of facility location, scheduling and set cover problems. While these works operate in well-structured combinatorial settings, our work addresses significantly more complex landscape of RL where the policy space is vast and computing optimal policies is inherently challenging. This necessitates novel algorithmic and theoretical advancements to efficiently construct portfolios with strong guarantees.

3. Preliminaries

In this section, we provide preliminaries on MORL, p-mean social welfare functions, and portfolios.

3.1. Multi-Objective Reinforcement Learning

We consider a multi-objective Markov decision process (MDP), defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \mathbf{R} = (R_i)_{i\in[N]})$, where $\mathcal{S}$ denotes a finite state space, $\mathcal{A}$ denotes a finite action space, and $P : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ 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 N distinct stakeholders, each associated with a reward function $R_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_{>0}$ for $i \in [N]$. A policy $\pi : \mathcal{S} \times [H] \to \Delta(\mathcal{A})$ 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 3.1 (Bounded Reward)**.** There exist strictly positive scalars U, L such that $L \le R_i(\cdot) \le U$ for all $i \in [N]$. We call $\kappa := U/L$ the *condition number* of rewards.

The N stakeholders could represent different entities depending on the application; for example, different con-

stituent demographics of a democracy, stakeholders in a company, or simply just abstract differences in values or preferences.

3.2. Social Welfare Function

We consider the following family of social welfare functions $f(\cdot, p) : \mathbb{R}^N_{>0} \to \mathbb{R}_{>0}$ called the *generalized p-mean*. It is defined for each $p \in [-\infty, 1]$ and strictly each positive vector $\mathbf{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N_{>0}$ as follows:

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

We also provide an illustrative example in Appendix B to provide the intuitive impact of various p-mean objectives. We will use the following *monotonicity* property:

Lemma 3.2 (Theorem 1, Chapter 3.3.1 (Bullen, 2003))**.** *For all strictly positive vectors $\mathbf{x} \in \mathbb{R}^N_{>0}$, and for all $p, q \leq 1$ with $p < q$, we have $f(\mathbf{x}, p) \leq f(\mathbf{x}, q)$.*

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 $G_i(\tau) = \sum_{h=1}^H R_i(s_h, a_h), i \in [N]$, represent the total reward over the trajectory, and $\mathbf{G}(\tau) \in \mathbb{R}^N$ the vector with $G_i(\tau)$ as its ith entry. The aggregation functions are defined as follows:

$$v^{(1)}(\pi, p) = \mathbb{E}_{\tau\sim\pi}\left[f\big(\mathbf{G}(\tau), p\big)\right], \tag{1}$$

$$v^{(2)}(\pi, p) = f\big(\mathbb{E}_{\tau\sim\pi}[\mathbf{G}(\tau)], p\big). \tag{2}$$

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, (Fan et al., 2023) and (Agarwal et al., 2022) 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 Agarwal et al. (2022), Roijers et al. (2013), Rădulescu et al. (2019), Agarwal et al. (2022), and Fan et al. (2023). While these aggregation rules are often studied independently in MORL, our work provides a unified framework applicable to both methods. We show that the proposed algorithms apply to both settings leveraging similar theoretical foundations.

3.3. Portfolio of Policies for All p-means

In this section, we introduce the key definitions for portfolios. 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 3.3 (α-approximation[1])**.** Given a $p \leq 1$, aggregation rule $\ell \in \{1, 2\}$, and an approximation factor $\alpha \in (0, 1)$, a policy $\pi \in \Pi$ is an α-approximation to the aggregation function $v^{(\ell)}(\cdot, p)$ if the value $v^{(\ell)}(\pi, p)$ achieved by π is within factor α of the maximum achievable value of $v^{(\ell)}(\cdot, p)$ by policies in Π, that is,

$$v^{(\ell)}(\pi, p) \geq \alpha \times \max_{\pi'\in\Pi} v^{(\ell)}(\pi', p) = \alpha \times v^{(\ell)}_*(p).$$

When ℓ and Π are clear from context, we say for brevity that π is an α-approximate policy for p.

Definition 3.4 (Portfolio)**.** Given 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 in Π' that is an α-approximation to $v^{(\ell)}(\cdot, p)$.

4. Portfolio Algorithms

In this section, we present our main algorithmic and theoretical results. Before presenting the details of our proposed method, we first define the notion of an oracle, which serves as a key component in our algorithm and analysis. Recall that 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:

$$v^{(\ell)}_*(p) = \max_{\pi\in\Pi} v^{(\ell)}(\pi, p). \tag{3}$$

The algorithms we develop assume that (3) can be solved, and we refer to each instance of solving (3) as an "oracle". This terminology is used to facilitate the discussion of oracle complexity, which quantifies the number of times Problem (3) must be solved to construct a portfolio.

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

[1]The α-approximation we refer to is distinct from the α-fairness mentioned in Section 2. To avoid confusion, from this point onward, all instances of α will exclusively refer to the approximation factor.

Algorithm 1 p-MEANPORTFOLIO$(\mathcal{M}, \Pi, \ell, \alpha)$

input (i) MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \mathbf{R} = (R_i)_{i\in[N]})$ with N reward functions, (ii) feasible set of policies Π for $\mathcal{M}$, (iii) aggregation rule $\ell \in \{1, 2\}$, and (iv) desired approximation factor $\alpha \in (0, 1)$

output α-approximate portfolio Π' for the set of aggregation functions $v^{(\ell)}(\cdot, p), p \le 1$, assuming oracle access to solve problem (3)

1: initialize $p_0 = -\frac{\ln N}{\ln(1/\alpha)}$ and $t = 0$
2: initialize portfolio $\Pi' \leftarrow \emptyset$
3: **while** $p_t < 1$ **do**
4: add $\pi_{p_t} := \arg\max_{\pi\in\Pi} v^{(\ell)}(\pi, p_t)$ to Π'
5: $p_{t+1} =$ LINESEARCH$(v^{(\ell)}, p_t, \Pi, \alpha)$
6: $t \leftarrow t + 1$
7: **return** Π'

Algorithm 2 LINESEARCH$(v^{(\ell)}, p, \Pi, \alpha)$

input (i) aggregation function $v^{(\ell)}$ (ii) some $p \in (-\infty, 1)$ (iii) feasible set Π of policies, and (iv) desired approximation $\alpha \in (0, 1)$

output $b^* > p$ such that $\pi_p := \arg\max_{\pi'\in\Pi} v^{(\ell)}(\pi', p)$ is an α-approximation for $v^{(\ell)}(\cdot, q)$ for all $q \in [p, b^*]$

1: $a \leftarrow p$ and $b \leftarrow 1$
2: initialize $\pi = \arg\max_{\pi'\in\Pi} v^{(\ell)}(\pi', p)$
3: **while** $v^{(\ell)}(\pi, a) < \alpha\, v_*^{(\ell)}(b)$ **do**
4: $q \leftarrow \frac{a+b}{2}$
5: **if** $v^{(\ell)}(\pi, a) \ge \sqrt{\alpha}\, v_*^{(\ell)}(q)$ **then**
6: $a \leftarrow q$
7: **else**
8: $b \leftarrow q$
9: **return** b

for Problem (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, Problem (3) can be solved efficiently by enumerating over the policies $\pi \in \Pi$ and using Monte Carlo simulations to estimate the value $v^{(\ell)}(\pi, p)$.

4.1. Algorithm p-MEANPORTFOLIO

In this section, we introduce p-MEANPORTFOLIO (Algorithm 1), which constructs an α-approximate portfolio Π' for all $p \le 1$. In Theorem 4.1, we establish formal bounds on the portfolio size $|\Pi'|$ and the number of oracle calls to solve (3). The full proof is provided in Appendix A.

Theorem 4.1. *Given an MDP $\mathcal{M}$ with N 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 p-MEANPORTFOLIO returns an α-approximate portfolio of policies Π' for the set of aggregation functions $\{v^{(\ell)}(\cdot, p) : p \le 1\}$. 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*

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

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

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 2). It ensures that π_{p_t} is α-approximate for all $p \in [p_t, p_{t+1}]$, and that π_{p_0} is α-approximate for all $p \in [-\infty, p_0]$.

Line search. The line search subroutine works as follows: given a $p \le 1$, it seeks to find some $b^* \ge p$ such that π_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 $p \le a < b \le 1$ and iteratively refines these bounds.

At each iteration, the algorithm checks whether $v^{(\ell)}(\pi_p, a) \ge \alpha v_*^{(\ell)}(b)$ (line 3). If this condition holds, then by the monotonicity property of p-means (Lemma 3.2), we must have:

$$v^{(\ell)}(\pi_p, b) \ge v^{(\ell)}(\pi_p, a) \ge \alpha\, v_*^{(\ell)}(b).$$

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 $v_*^{(\ell)}(q)$ for $q = \frac{a+b}{2}$. As before, if $v^{(\ell)}(\pi_p, a)$ exceeds α times $v_*^{(\ell)}(q)$, then we know that π_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 A, we can in fact ensure faster convergence by slightly modifying this step. Instead of checking whether $v^{(\ell)}(\pi_p, a) \ge$

$\alpha \times v_*^{(\ell)}(q)$, we check the stronger condition $v^{(\ell)}(\pi_p, a) \geq \sqrt{\alpha} \times v_*^{(\ell)}(q)$. 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 to solve Problem (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 A.8, this implies that after j iterations of LINESEARCH, the ratio $\frac{v_*^{(\ell)}(b)}{v_*^{(\ell)}(a)}$ is upper bounded by $\psi(\kappa, N, \alpha) \times 2^{-j}$, where ψ is some function of condition number κ, dimension N, and approximation factor α. By deriving a lower bound on this ratio, we derive an upper bound on the total number of iterations j.

4.2. 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 Problem (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 Theorem 4.1 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 BUDGET-CONSTRAINEDPORTFOLIO (Algorithm 3), 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 Π'_t denote our current portfolio. The objective at each step is to identify the worst-case p value where the approximation quality of Π'_t is the lowest by solving the following problem:

$$\mathcal{Q}(\Pi'_t) = \min_{p \leq 1} \frac{\max_{\pi \in \Pi'_t} v^{(\ell)}(\pi, p)}{v_*^{(\ell)}(p)}. \tag{4}$$

However, solving this optimization problem exactly is computationally infeasible, particularly since evaluating the objective for any new p would require an additional oracle call. Instead, BUDGET-CONSTRAINEDPORTFOLIO 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 Problem (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 (4) by decomposing the minimization over the intervals:

$$\mathcal{Q}(\Pi'_t) \approx \min_{l \in [t-1]} \left[\min_{p \in [p_{m(l)}, p_{m(l+1)}]} \frac{\max_{\pi \in \Pi'_t} v^{(\ell)}(\pi, p)}{v_*^{(\ell)}(p)} \right].$$

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 (Theorem 4.1). 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:

$$\begin{aligned}
&\min_{p \in [p_{m(l)}, p_{m(l+1)}]} \frac{\max_{\pi \in \Pi'_t} v^{(\ell)}(\pi, p)}{v_*^{(\ell)}(p)} \\
&\approx \min_{p \in [p_{m(l)}, p_{m(l+1)}]} \frac{v^{(\ell)}(\pi_{p_{m(l)}}, p)}{v_*^{(\ell)}(p)} \\
&\approx \frac{v^{(\ell)}(\pi_{p_{m(l)}}, p_{m(l+1)})}{v_*^{(\ell)}(p_{m(l+1)})} := u(l).
\end{aligned}$$

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 $\frac{v^{(\ell)}(\pi_{p_{m(l)}}, p)}{v_*^{(\ell)}(p)}$ 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 A.8) 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) approximation factor and choose its

Algorithm 3 BUDGETCONSTRAINEDPORTFOLIO

input (i) MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \mathbf{R} = (R_i)_{i\in[N]})$ with N reward functions, (ii) aggregation rule $\ell \in \{1, 2\}$, (iii) feasible set of policies Π for $\mathcal{M}$, (iv) budget K, and (v) initial p_0
output A portfolio Π'

Initialize $\Pi' \leftarrow \emptyset$
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 $\pi_{p_t} := \arg\max_{\pi\in\Pi} v^{(\ell)}(\pi, p_t)$ to Π'
return Π'

midpoint:

$$p_{t+1} = \frac{p_{m(l^*)} + p_{m(l^*+1)}}{2}, \quad l^* = \arg\min_{l\in[t-1]} u(l).$$

4.3. Warm-Start for Computational Efficiency

Both p-MEANPORTFOLIO and BUDGETCONSTRAINEDPORTFOLIO involve repeatedly solving Problem (3) for different p values. To accelerate this process, warm-starting is a natural choice. Specifically, once the optimal policy π_p for $\max_{\pi\in\Pi} v^{(\ell)}(\pi, p)$ is found, it can be used as a warm-starting point for solving $\max_{\pi\in\Pi} v^{(\ell)}(\pi, q), q > p$, if q and p are close. Proposition 4.2 below offers a theoretical justification for this approach by showing that $v^{(\ell)}(\pi_p, q)$ is close to the optimal value $v^{(\ell)}_*(q)$ whenever p and q are close. The proof is deferred to Appendix A.

Proposition 4.2. *Let $q > p$. For $\ell \in \{1, 2\}$, the following inequality holds:*

$$\left| v^{(\ell)}_*(q) - v^{(\ell)}(\pi_p, q) \right| \le (q - p)UH\kappa \ln \kappa,$$

where π_p denotes the policy $\arg\max_{\pi\in\Pi} v^{(\ell)}(\pi, p)$.

5. 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 Appendix E. The code for our experiments can be found at https://github.com/jaimoondra/approximation-portfolios-for-rl/.

5.1. Experimental Setups

For a given α and MDP environment, we first compute a portfolio using p-MEANPORTFOLIO 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 baselines involve randomness, we generate 10 independent baseline portfolios for each setting and report the average performance across these runs. Finally, we run BUDGETCONSTRAINEDPORTFOLIO with budget K.

To evaluate each portfolio Π', we compute its approximation factor $\mathcal{Q}(\Pi')$ as defined in Equation (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 BUDGETCONSTRAINEDPORTFOLIO. This procedure is repeated for varying α values.

5.2. 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 Dietterich (2000); Fan et al. (2022). The taxi environment consists of a grid world where a taxi driver serves passengers across $N = 4$ 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 (Fan et al., 2022). 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 $N = 12$ 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. We consider a real-world healthcare intervention problem, modeled as an RMAB problem described in Section 1.2 (Verma et al., 2024b). ARMMAN (ARMMAN, 2024), 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 $N = 59$ 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.

5.3. Results

Table 1 presents comparisons of the approximation quality of p-MEANPORTFOLIO with random p sampling and random policy sampling baselines, while Table 2 reports the number of oracle calls made by p-MEANPORTFOLIO and BUDGETCONSTRAINEDPORTFOLIO. Note that by design, the number of calls for BUDGETCONSTRAINEDPORTFOLIO 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 ≤ 1.

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 BUDGETCONSTRAINEDPORTFOLIO in most cases.

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, BUDGETCONSTRAINEDPORTFOLIO achieves comparable quality while using significantly fewer oracle calls. This suggests that when computational resources are limited, BUDGETCONSTRAINEDPORTFOLIO serves as a strong alternative with good empirical performance.

Table 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

Diversity. As we have shown in Figure 1 in Section 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 Appendix D, Table 4. 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. Figures 3 and 4 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 ps as well.

6. 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 es-

[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 4.

Table 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

tablished theoretical guarantees on the trade-offs among α, portfolio size, and the number of oracle calls. Additionally, we presented a theoretical analysis of warm-start techniques and 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.

Acknowledgments

We thank NSF AI Institute for Societal Decision Making (AI-SDM) Award No. 2229881, ONR MURI N00014-24-1-2742, and NSF CAREER 2239824 for their support.

Impact Statement

Our work studies the setting where a deployed RL policy impacts multiple stakeholders differently, a scenario that arises in various societal applications, including healthcare and LLMs. By summarizing the space of policies induced under different p-values, our proposed approach aims to assist decision-makers in making informed and equitable choices. Furthermore, this work has the potential to stimulate discussions within the machine learning community about the complexities and pluralities of social welfare notions, and the importance of addressing them. However, our focus is limited to p-means social welfare functions, which may not fully capture all dimensions of fairness in real-world scenarios.

Our experiments use real-world healthcare data provided by the NGO ARMMAN. These experiments are strictly secondary analyses, with no real-world deployment of the proposed algorithm or baseline methods. All data usage complied with ARMMANs ethics board approvals, and data exchange and analysis followed their ethics review committee's guidelines, including privacy, informed consent, and anonymization. All of the analyses is conducted in close collaboration with ARMMAN.

This work belongs to the broader area of algorithmic decision-making, where deployment decisions can have real-world implications. We emphasize the importance of not relying on this study in isolation. Responsible deployment requires comprehensive field testing and human oversight. Moreover, we acknowledge broader challenges inherent in algorithmic decision-making, including potential biases in data, inaccuracies in the underlying mathematical models, accountability issues, and concerns related to consent and privacy. Addressing these challenges is essential to ensuring that algorithmic decisions are ethically sound.

References

Mridul Agarwal, Vaneet Aggarwal, and Tian Lan. Multi-objective reinforcement learning with non-linear scalarization. In Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS '22, page 917, Richland, SC, 2022. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450392136.

Parand A. Alamdari, Soroush Ebadian, and Ariel D. Procaccia. Policy aggregation. In The Thirty-eighth Annual Conference on Neural Information Processing Systems,

2024. URL https://openreview.net/forum?id=ybiUVIxJth.

Lucas N. Alegre, Diederik M. Roijers, Ann Nowé, Ana L. C. Bazzan, and Bruno C. da Silva. Sample-efficient multi-objective learning via generalized policy improvement prioritization. In Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2023.

ARMMAN. Armman: Advancing reduction in mortality and morbidity of mothers, children, and neonates, 2024. URL https://armman.org/.

P.S Bullen. Handbook of Means and Their Inequalities. Mathematics and Its Applications ; 560. Springer Netherlands : Imprint: Springer, Dordrecht, 2nd ed. 2003. edition, 2003.

Deeparnab Chakrabarty and Chaitanya Swamy. Approximation algorithms for minimum norm and ordered optimization problems. In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, pages 126–137, 2019.

Souradip Chakraborty, Jiahao Qiu, Hui Yuan, Alec Koppel, Dinesh Manocha, Furong Huang, Amrit Bedi, and Mengdi Wang. MaxMin-RLHF: Alignment with diverse human preferences. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, Proceedings of the 41st International Conference on Machine Learning, volume 235 of Proceedings of Machine Learning Research, pages 6116–6135. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/chakraborty24b.html.

Jingdi Chen, Yimeng Wang, and Tian Lan. Bringing fairness to actor-critic reinforcement learning for network utility optimization. In IEEE INFOCOM 2021 - IEEE Conference on Computer Communications, pages 1–10, 2021. doi: 10.1109/INFOCOM42981.2021.9488823.

Cyrus Cousins. Revisiting fair-pac learning and the axioms of cardinal welfare. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, Proceedings of The 26th International Conference on Artificial Intelligence and Statistics, volume 206 of Proceedings of Machine Learning Research, pages 6422–6442. PMLR, 25–27 Apr 2023. URL https://proceedings.mlr.press/v206/cousins23a.html.

Cyrus Cousins, Kavosh Asadi, Elita Lobo, and Michael Littman. On welfare-centric fair reinforcement learning. Reinforcement Learning Journal, 3:1124–1137, 2024.

Thomas G Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. Journal of artificial intelligence research, 13:227–303, 2000.

Marina Drygala, Silvio Lattanzi, Andreas Maggiori, Miltiadis Stouras, Ola Svensson, and Sergei Vassilvitskii. Data-driven solution portfolios. arXiv preprint arXiv:2412.00717, 2024.

Zimeng Fan, Nianli Peng, Muhang Tian, and Brandon Fain. Welfare and fairness in multi-objective reinforcement learning. arXiv preprint arXiv:2212.01382, 2022.

Ziming Fan, Nianli Peng, Muhang Tian, and Brandon Fain. Welfare and fairness in multi-objective reinforcement learning. In Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS '23, page 19911999, Richland, SC, 2023. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450394321.

Ashish Goel and Adam Meyerson. Simultaneous optimization via approximate majorization for concave profits or convex costs. Algorithmica, 44:301–323, 2006.

Daniel Golovin, Anupam Gupta, Amit Kumar, and Kanat Tangwongsan. All-norms and all-l_p-norms approximation algorithms. In IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (2008). Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2008.

Swati Gupta, Jai Moondra, and Mohit Singh. Which lp norm is the fairest? Approximations for fair facility location across all "p". In Proceedings of the 24th ACM Conference on Economics and Computation, EC '23, page 817, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701047. doi: 10.1145/3580507.3597664. URL https://doi.org/10.1145/3580507.3597664.

Swati Gupta, Jai Moondra, and Mohit Singh. Balancing notions of equity: Trade-offs between fair portfolio sizes and achievable guarantees. In Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2025. URL https://arxiv.org/abs/2311.03230.

Hao Hao, Changqiao Xu, Wei Zhang, Shujie Yang, and Gabriel-Miro Muntean. Computing offloading with fairness guarantee: A deep reinforcement learning method. IEEE Transactions on Circuits and Systems for Video Technology, 33(10):6117–6130, 2023. doi: 10.1109/TCSVT.2023.3255229.

Conor F. Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond,

Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. A practical guide to multi-objective reinforcement learning and planning. Autonomous Agents and Multi-Agent Systems, 36(1):26, April 2022.

Peizhong Ju, Arnob Ghosh, and Ness Shroff. Achieving fairness in multi-agent MDP using reinforcement learning. In The Twelfth International Conference on Learning Representations, 2024. URL https://openreview.net/forum?id=yoVq2BGQdP.

Debmalya Mandal and Jiarui Gan. Socially fair reinforcement learning, 2023. URL https://arxiv.org/abs/2208.12584.

Hervé Moulin. Fair Division and Collective Welfare. The MIT Press, January 2003.

Christos H Papadimitriou and John N Tsitsiklis. The complexity of optimal queuing network control. Mathematics of Operations Research, 24(2):293–305, 1999.

Kanad Shrikar Pardeshi, Itai Shapira, Ariel D. Procaccia, and Aarti Singh. Learning social welfare functions. In The Thirty-eighth Annual Conference on Neural Information Processing Systems, 2024. URL https://openreview.net/forum?id=7O6KtaAr8n.

Simone Parisi, Matteo Pirotta, Nicola Smacchia, Luca Bascetta, and Marcello Restelli. Policy gradient approaches for multi-objective sequential decision making. In 2014 International Joint Conference on Neural Networks (IJCNN), pages 2323–2330, 2014. doi: 10.1109/IJCNN.2014.6889738.

Chanwoo Park, Mingyang Liu, Dingwen Kong, Kaiqing Zhang, and Asuman Ozdaglar. RLHF from heterogeneous feedback via personalization and preference aggregation, 2024. URL https://arxiv.org/abs/2405.00254.

Julien Perez, Cécile Germain-Renaud, Balázs Kégl, and Charles Loomis. Responsive elastic computing. In Proceedings of the 6th International Conference Industry Session on Grids Meets Autonomic Computing, GMAC '09, page 5564, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585789. doi: 10.1145/1555301.1555311. URL https://doi.org/10.1145/1555301.1555311.

Roxana Rădulescu, Patrick Mannion, Diederik M. Roijers, and Ann Nowé. Multi-objective multi-agent decision making: a utility-based analysis and survey. Autonomous Agents and Multi-Agent Systems, 34(1):10, December 2019.

Mathieu Reymond, Eugenio Bargiacchi, and Ann Nowé. Pareto conditioned networks. In Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS '22, page 11101118, Richland, SC, 2022. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450392136.

Kevin W. S. Roberts. Interpersonal Comparability and Social Choice Theory. The Review of Economic Studies, 47(2):421–439, January 1980.

Diederik M. Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. J. Artif. Int. Res., 48(1):67113, October 2013. ISSN 1076-9757.

Kristof Van Moffaert and Ann Nowé. Multi-objective reinforcement learning using sets of pareto dominating policies. J. Mach. Learn. Res., 15(1):34833512, January 2014. ISSN 1532-4435.

Shresth Verma, Gargi Singh, Aditya Mate, Paritosh Verma, Sruthi Gorantla, Neha Madhiwalla, Aparna Hegde, Divy Thakkar, Manish Jain, Milind Tambe, et al. Expanding impact of mobile health programs: Saheli for maternal and child care. AI Magazine, 44(4):363–376, 2023.

Shresth Verma, Niclas Boehmer, Lingkai Kong, and Milind Tambe. Balancing act: Prioritization strategies for LLM-designed restless bandit rewards, 2024a. URL https://arxiv.org/abs/2408.12112.

Shresth Verma, Gargi Singh, Aditya Mate, Paritosh Verma, Sruthi Gorantla, Neha Madhiwalla, Aparna Hegde, Divy Thakkar, Manish Jain, Milind Tambe, and Aparna Taneja. Increasing impact of mobile health programs: Saheli for maternal and child care. Proceedings of the AAAI Conference on Artificial Intelligence, 37(13):15594–15602, Jul. 2024b.

P. Whittle. Restless bandits: Activity allocation in a changing world. Journal of Applied Probability, 25:287–298, 1988. ISSN 00219002. URL http://www.jstor.org/stable/3214163.

Qingqing Wu, Yong Zeng, and Rui Zhang. Joint trajectory and communication design for multi-uav enabled wireless networks. IEEE Transactions on Wireless Communications, 17(3):2109–2121, 2018. doi: 10.1109/TWC.2017.2789293.

Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. Curran Associates Inc., Red Hook, NY, USA, 2019.

Guanbao Yu, Umer Siddique, and Paul Weng. Fair deep reinforcement learning with generalized gini welfare functions. In Francesco Amigoni and Arunesh Sinha, editors, Autonomous Agents and Multiagent Systems. Best and Visionary Papers, pages 3–29, Cham, 2024. Springer Nature Switzerland.

Huiying Zhong, Zhun Deng, Weijie J. Su, Zhiwei Steven Wu, and Linjun Zhang. Provable multi-party reinforcement learning with diverse human feedback, 2024. URL https://arxiv.org/abs/2403.05006.

A. Omitted Proofs

We include omitted proofs and various lemmas here.

A.1. Proof of Theorem 4.1

Our first lemma shows establishes the monotonicity of aggregation functions $v^{(\ell)}$ in p:

Lemma A.1. *For any fixed policy $\pi \in \Pi$, $v^{(\ell)}(\pi, p)$ is monotone increasing in p for $\ell \in \{1, 2\}$.*

Proof. $\ell = 1$: For any $\tau \sim \pi$, Lemma 3.2 implies that $f(\mathbf{G}(\tau), p)$ is monotone increasing in p. Therefore,

$$v^{(1)}(\pi, p) = \mathbb{E}_{\tau\sim\pi}[f(\mathbf{G}(\tau), p)]$$

is monotone increasing in p.

$\ell = 2$: Denote $\mathbf{x} = \mathbb{E}_{\tau\sim\pi}[\mathbf{G}(\tau)]$. Then the monotonicity of $v^{(2)}(\pi, p) = f(\mathbf{x}, p)$ follows directly from Lemma 3.2. □

The following corollary follows immediately:

Corollary A.2. *$v_*^{(\ell)}(p) := \max_{\pi\in\Pi} v^{(\ell)}(\pi, p)$ is monotone increasing in p for $\ell \in \{1, 2\}$.*

The following two lemmas show that the p-mean for $p = -\infty$ is α-approximated by the p_0-mean where $p_0 = -\frac{\ln N}{\ln(1/\alpha)}$, so we can effectively restrict to $[-p_0, 1]$ when finding portfolios:

Lemma A.3. *Given a vector $\mathbf{x} \in \mathbb{R}^N_{>0}$ and $\alpha \in (0, 1)$, define $p_0 = -\frac{\ln N}{\ln(1/\alpha)}$. Then,*

$$f(\mathbf{x}, -\infty) \geq \alpha \cdot f(\mathbf{x}, p) \quad \forall\, p \leq p_0.$$

Proof. Suppose $0 < x_1 \leq \ldots \leq x_n$, so that $f(\mathbf{x}, -\infty) = \min_{j\in[N]} x_j = x_1$. Given $p \leq p_0$, denote $q = -p \geq \frac{\ln N}{\ln(1/\alpha)}$. Then, since

$$f(\mathbf{x}, p) = \left(\frac{1}{N}\sum_{j\in[N]} x_j^p\right)^{1/p} = \frac{1}{\left(\frac{1}{N}\sum_{j\in[N]}\frac{1}{x_j^q}\right)^{1/q}} = \frac{1}{\frac{1}{x_1}\left(\frac{1}{N}\sum_{j\in[N]}\left(\frac{x_1}{x_j}\right)^q\right)^{1/q}},$$

we get

$$\frac{1}{f(\mathbf{x}, p)} \geq \frac{1}{x_1}\left(\frac{1}{N} \times \left(\frac{x_1}{x_1}\right)^q\right)^{1/q} = \frac{N^{1/p}}{x_1} \geq \frac{N^{1/p_0}}{x_1} = \frac{\alpha}{f(\mathbf{x}, -\infty)},$$

or that $f(\mathbf{x}, -\infty) \geq \alpha \cdot f(\mathbf{x}, p)$. □

Lemma A.4. *Given an MDP $\mathcal{M}$ with N reward functions, set Π of feasible policies, and an approximation factor $\alpha \in (0, 1)$, denote $p_0 = -\frac{\ln N}{\ln(1/\alpha)}$. Then, for a given aggregation rule $\ell \in \{1, 2\}$, policy $\pi_0 = \arg\max_{\pi\in\Pi} v^{(\ell)}(\pi, p_0)$ is an α-approximation for all $p \leq p_0$. That is,*

$$v^{(\ell)}(\pi_0, p) \geq v_*^{(\ell)}(p) = \max_{\pi\in\Pi} v^{(\ell)}(\pi, p) \quad \forall\, p \leq p_0.$$

Proof. $\ell = 1$: From Lemma A.3, for all $p \leq p_0$, we get

$$\begin{aligned}
v^{(1)}(\pi_0, p) &\geq v^{(1)}(\pi_0, -\infty) && \text{(Lemma A.1)}\\
&= \mathbb{E}_{\tau\sim\pi}[f(\mathbf{G}(\tau), -\infty)] && \text{(eqn.(1))}\\
&\geq \alpha \cdot \mathbb{E}_{\tau\sim\pi}[f(\mathbf{G}(\tau), p_0)] && \text{(Lemma A.3)}\\
&= \alpha \cdot v^{(1)}(\pi_0, p_0) && \text{(eqn.(1))}\\
&= \alpha \cdot \max_{\pi\in\Pi} \cdot v^{(1)}(\pi, p_0)\\
&\geq \alpha \cdot \max_{\pi\in\Pi} v^{(1)}(\pi, p) && \text{(Lemma A.1).}
\end{aligned}$$

The result for $\ell = 2$ follows similarly and is omitted. □

The following three lemmas establish guarantees on LINESEARCH:

Lemma A.5. *Suppose* LINESEARCH *(Algorithm 2) on input $v^{(\ell)}, p, \Pi, \alpha$ returns $b^* \in [p, 1]$. Then, either $b^* = 1$, or the policy $\pi := \arg\max_{\pi \in \Pi} v^{(\ell)}(\pi, p)$ satisfies*

$$\frac{v_*^{(\ell)}(p)}{\sqrt{\alpha}} \le v_*^{(\ell)}(b^*) \le \frac{v^{(\ell)}(\pi, b^*)}{\alpha}.$$

Proof. If LINESEARCH$(v^{(\ell)}, p, \Pi, \alpha)$ does not return $b^* = 1$, then we must have that p-MEANPORTFOLIO enters the while loop (step 2) at least once. In particular,

$$v_*^{(\ell)}(p) = v^{(\ell)}(\pi, p) < \alpha\, v_*^{(\ell)}(b^*) < \sqrt{\alpha}\, v_*^{(\ell)}(b^*).$$

This proves the first inequality. For the second inequality, notice that the algorithm terminates only when $v^{(\ell)}(\pi, a) \ge \alpha\, v_*^{(\ell)}(b^*)$. As can be easily checked, the algorithm maintains the invariant $a \ge p$. Therefore, by Lemma A.1,

$$v^{(\ell)}(\pi, b^*) \ge v^{(\ell)}(\pi, a) \ge \alpha\, v_*^{(\ell)}(b^*).$$

□

Lemma A.6. *Algorithm* LINESEARCH *maintains the following invariant at all times: $v^{(\ell)}(\pi, a) \ge \sqrt{\alpha}\, v_*^{(\ell)}(a)$.*

Proof. Initially, $p = a$, and therefore $\pi := \arg\max_{\pi' \in \Pi} v^{(\ell)}(\pi', p)$ satisfies $v^{(\ell)}(\pi, p) = v_*^{(\ell)}(p) \ge \sqrt{\alpha}\, v_*^{(\ell)}(p)$ since $\alpha \in (0, 1)$.

Further, the algorithm only updates $a \leftarrow q$ in step 2 when $v^{(\ell)}(\pi, a) \ge \sqrt{\alpha}\, v_*^{(\ell)}(q)$. Since $q \ge a$, we get from Lemma A.1 that $v^{(\ell)}(\pi, q) \ge v^{(\ell)}(\pi, a)$, thus finishing the proof. □

Lemma A.7. *Algorithm* LINESEARCH *on input $p, v^{(\ell)}, \Pi, \alpha$ returns $b^* > p$ such that $\pi := \arg\max_{\pi' \in \Pi} v^{(\ell)}(\pi, p)$ is an α-approximation for all $q \in [p, b^*]$, i.e., $v^{(\ell)}(\pi, q) \ge \alpha\, v_*^{(\ell)}(q)$ for all such q.*

Proof. LINESEARCH starts with $a \leftarrow p$ and keeps increasing $a \leftarrow q$ whenever $v^{(\ell)}(\pi, a) \ge \sqrt{\alpha}\, v_*^{(\ell)}(q)$ 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 A.1,

$$v^{(\ell)}(\pi, q') \ge v^{(\ell)}(\pi, a) \ge \sqrt{\alpha}\, v_*^{(\ell)}(\pi, (a+b)/2) \ge \sqrt{\alpha}\, v_*^{(\ell)}(\pi, q') > \alpha\, v_*^{(\ell)}(q').$$

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 $v^{(\ell)}(\pi, a) \ge v_*^{(\ell)}(b^*)$, i.e., for all $q' \in [a, b^*]$, we get

$$v^{(\ell)}(\pi, q') \ge v^{(\ell)}(\pi, a) \ge \alpha\, v_*^{(\ell)}(b^*) \ge \alpha\, v_*^{(\ell)}(q').$$

□

The next lemma bounds the slope of the logarithm of the p-mean function and consequently the slope of $\ln v_*^{(\ell)}(p)$:

Lemma A.8. *1. For any $\mathbf{x} \in \mathbb{R}^N_{>0}$, such that $HL \le x_i \le HU$ for all $i \in [N]$, define $g(\mathbf{x}, p) := \ln f(\mathbf{x}, p) = \frac{1}{p} \ln\left(\frac{1}{N} \sum_{i \in [N]} x_i^p\right)$. Then, if $L \le x_i \le U$ with condition number $\kappa := \frac{U}{L}$, we have*

$$\frac{dg(\mathbf{x}, p)}{dp} \le \kappa \ln \kappa.$$

2. *For any given policy $\pi \in \Pi$ and aggregation rule $\ell \in \{1, 2\}$, we have $\frac{d\left(\ln v^{(\ell)}(\pi, p)\right)}{dp} \le \kappa \ln \kappa$, where κ is the condition number defined in Assumption 3.1.*

3. *For a given aggregation rule $\ell \in \{1, 2\}$, define $w(p) := \ln v_*^{(\ell)}(p)$. Then, for all distinct $p, q \le 1$,*

$$\frac{w(q) - w(p)}{q - p} \le \kappa \ln \kappa.$$

Proof. **Part 1.** Note that $g(\beta\mathbf{x}, p) = \ln\beta + g(\mathbf{x}, p)$ for all $\mathbf{x} \in \mathbb{R}^N_{>0}$, $p \leq 1$, and $\beta > 0$. Therefore, we can assume without loss of generality that $LH = 1$ and $UH = \frac{UH}{LH} = \kappa$.

That is, assume without of generality that $1 \leq x_1 \leq \ldots \leq x_N \leq \kappa$. Since $g(\mathbf{x}, p) = \frac{-\ln N}{p} + \frac{1}{p}\ln\left(\sum_{i\in[N]} x_i^p\right)$, we get that $pg(\mathbf{x}, p) = -\ln N + \ln\left(\sum_{i\in[N]} x_i^p\right)$. Therefore,

$$g(\mathbf{x}, p) + p\frac{dg(\mathbf{x}, p)}{dp} = \frac{\sum_i (\ln x)\cdot x_i^p}{\sum_i x_i^p}. \tag{5}$$

Since $\ln t$ is concave in t, we get that

$$pg(\mathbf{x}, p) = \ln\left(\frac{1}{N}\sum_i x_i^p\right) \geq \frac{1}{N}\sum_i \ln x_i^p = \frac{p}{N}\sum_i \ln x_i.$$

Therefore, we get

$$g(\mathbf{x}, p)\begin{cases} \geq \frac{1}{N}\sum_i \ln x_i & \text{if } p \geq 0, \\ \leq \frac{1}{N}\sum_i \ln x_i & \text{if } p < 0. \end{cases} \tag{6}$$

Denote $\mu_p = \frac{1}{N}\sum_{i\in[N]} x_i^p$.

Case A: $p \in [0, 1]$**.** Plugging this back into eqn. (5), we get

$$p\frac{dg(\mathbf{x}, p)}{dp} \leq \frac{\sum_i (\ln x_i)\cdot x_i^p}{N\mu_p} - \frac{1}{N}\sum_i \ln x_i = \frac{1}{N}\left(\sum_i \ln x_i\left(\frac{x_i^p}{\mu_p} - 1\right)\right).$$

Since $1 \leq x_i \leq \kappa$ for all i, we have $0 \leq \ln x_i \leq \ln\kappa$ for all i. Further, since $\mathbf{x}_N \geq x_i$, we get that

$$Np\,\frac{dg(\mathbf{x}, p)}{dp} \leq \sum_i \ln x_i\left(\frac{x_i^p}{\mu_p} - 1\right) \leq \ln\kappa \sum_{i: x_i^p \geq \mu_p}\left(\frac{x_N^p}{\mu_p} - 1\right) \leq N\ln\kappa\left(\frac{x_N^p}{\mu_p} - 1\right). \tag{7}$$

Now, since μ_p is the mean of $x_i^p, i \in [N]$, we must have that $\mu_p = \theta^p$ for some $1 \leq \theta \leq \kappa$. Substituting this, we get $Np\,\frac{dg(\mathbf{x},p)}{dp} \leq N(\ln\kappa)\left((x_N/\theta)^p - 1\right)$. Since $\frac{x_N}{\theta} \leq \frac{\kappa}{1} = \kappa$ and the derivative of α^p with respect to α is $p\alpha^{p-1}$, we get

$$\begin{aligned}
((x_N/\theta)^p - 1) &\leq (\kappa^p - 1) \\
&= \int_1^\kappa p\cdot\alpha^{p-1}\,\mathrm{d}\alpha \\
&\leq \int_1^\kappa p\cdot 1^{p-1}\,\mathrm{d}\alpha && \text{(since } p-1 \leq 0 \text{ and so } \alpha^{p-1} \text{ is nonincreasing in } p\text{)} \\
&\leq p(\kappa - 1) \leq p\kappa. && \text{(since } p \geq 0\text{)}
\end{aligned}$$

Plugging this back into eqn. (7), we get

$$p\frac{dg(\mathbf{x}, p)}{dp} \leq p\kappa\ln\kappa,$$

or that $\frac{dg(\mathbf{x},p)}{dp} \leq \kappa\ln\kappa$ for all $p \in [0, 1]$.

Case B: $p < 0$**.** As before, plugging bound (6) into eqn. (5), we get

$$p\frac{dg(\mathbf{x}, p)}{dp} \geq \frac{\sum_i (\ln x_i)\cdot x_i^p}{N\mu_p} - \frac{1}{N}\sum_i \ln x_i = \frac{1}{N}\left(\sum_i \ln x_i\left(\frac{x_i^p}{\mu_p} - 1\right)\right) = -\frac{1}{N}\left(\sum_i \ln x_i\left(1 - \frac{x_i^p}{\mu_p}\right)\right).$$

That is, since $p < 0$,

$$\frac{dg(\mathbf{x}, p)}{dp} \leq \frac{1}{(-p)N}\left(\sum_i \ln x_i\left(1 - \frac{x_i^p}{\mu_p}\right)\right) \leq \frac{\ln R}{(-p)N}\sum_{i: x_i^p \leq \mu_p}\left(1 - \frac{x_i^p}{\mu_p}\right) \leq \frac{\ln R}{(-p)N}\times N\left(1 - \frac{x_N^p}{\mu_p}\right) = \frac{\ln\kappa}{(-p)}\left(1 - \frac{x_N^p}{\mu_p}\right).$$

Denote $-p = q$; then $q > 0$. As before, $\mu_p = \theta^p$ for some $1 \le \theta \le N$, and so we have

$$\frac{dg(\mathbf{x},p)}{dp} \le \frac{\ln\kappa}{q}\left(1 - \frac{\theta^q}{x_N^q}\right) \le \frac{\ln\kappa}{q}\left(1 - \frac{1}{\kappa^q}\right).$$

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 in a manner similar to Case A:

$$\begin{aligned}\frac{dg(\mathbf{x},p)}{dp} &\le \frac{\ln\kappa}{q}\int_{1/\kappa}^{1} q\alpha^{q-1}\,\mathrm{d}\alpha \\ &\le \ln\kappa \int_{1/\kappa}^{1} \frac{1}{\kappa^{q-1}}\,\mathrm{d}\alpha \\ &\le (\ln\kappa)\times\kappa^{1-q} \le \kappa\ln\kappa.\end{aligned}$$

Part 2. $\ell = 1$. Given any trajectory τ, for the vector $\mathbf{G}(\tau) \in [LH, UH]^N$, we get from Part 1 that

$$\frac{d\left(\ln f(\mathbf{G}(\tau),p)\right)}{dp} \le \kappa\ln\kappa.$$

Consequently, for any policy π, we get for $v^{(1)}(\pi,p) = \mathbb{E}_{\tau\sim\pi}\left[f(\mathbf{G}(\tau),p)\right]$ using linearity of expectation that

$$\frac{d\left(\ln v^{(1)}(\pi,p)\right)}{p} = \mathbb{E}_{\tau\sim\pi}\left[\frac{d\left(\ln f(\mathbf{G}(\tau),p)\right)}{dp}\right] \le \mathbb{E}_{\tau\sim\pi}\left[\kappa\ln\kappa\right] = \kappa\ln\kappa.$$

$\ell = 2$. Follows by taking $\mathbf{x} = \mathbb{E}_{\tau\sim\pi}[\mathbf{G}(\tau)]$ in Part 1.

Part 3. Note that $w(p) = \ln v_*^{(\ell)}(p) = \ln\max_{\pi\in\Pi} v^{(\ell)}(\pi,p) = \max_{\pi\in\Pi}\ln v^{(\ell)}(\pi,p)$. Given distinct $p, q \le 1$, denote $\pi_p = \arg\max_{\pi\in\Pi}\ln v^{(\ell)}(\pi,p)$. Then, by Part 1,

$$\frac{w(q)-w(p)}{q-p} = \frac{\left(\max_{\pi\in\Pi}\ln v^{(\ell)}(\pi,q)\right) - \ln v^{(\ell)}(\pi_p,p)}{q-p} \le \frac{\ln v^{(\ell)}(\pi_p,q) - \ln v^{(\ell)}(\pi_p,p)}{q-p} \le \kappa\ln\kappa. \qquad \square$$

We are ready to prove Theorem 4.1 that gives guarantees for p-MEANPORTFOLIO (Algorithm 1).

Proof of Theorem 4.1. We prove that (correctness) the set Π' of policies output by the algorithm is indeed an α-approximate portfolio, (size bound) $|\Pi'| = O\left(\frac{\ln\kappa}{\ln(1/\alpha)}\right)$, and (oracle complexity) the number of oracle calls to Problem (3) is upper bounded by

$$O\left(\frac{(\ln\kappa)^2\ln\ln N}{\ln(1/\alpha)\ln\ln(1/\alpha)}\right). \tag{8}$$

We note that this size bound can be tightened slightly to $O\left(\frac{(\ln\kappa)(\ln\kappa+\ln\ln N)}{\ln(1/\alpha)\ln\ln(1/\alpha)}\right)$; we present the the above cleaner bound for simplicity.

Correctness. Suppose the algorithm returns $\Pi' = \{\pi_0, \pi_1, \ldots, \pi_K\}$ such that $\pi_t = \arg\max_{\pi\in\Pi} v^{(\ell)}(\pi, p_t)$ with $-\frac{\ln N}{\ln(1/\alpha)} = p_0 < p_1 < \ldots < p_K = 1$.

Lemma A.4 shows that π_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 π_t is an α-approximation for all $p \in [p_t, p_{t+1}]$. Since $p_{t+1} = \textsc{LineSearch}(v^{(\ell)}, p_t, \Pi, \alpha)$, Lemma A.7 implies this.

Size bound. Note that by definition, $p_{t+1} = \textsc{LineSearch}(v^{(\ell)}, p_t, \Pi, \alpha)$. Therefore, from Lemma A.5, we have that π_t satisfies $v_*^{(\ell)}(p_{t+1}) \ge \frac{v_*^{(\ell)}(p_t)}{\sqrt{\alpha}}$ for all t except possibly $t = K-1$. Therefore,

$$v_*^{(\ell)}(p_K) \ge v_*^{(\ell)}(p_{K-1}) \ge \left(\frac{1}{\alpha}\right)^{(K-1)/2} v_*^{(\ell)}(p_0) = \left(\frac{1}{\alpha}\right)^{(K-1)/2} v_*^{(\ell)}(-\infty).$$

However, $v_*^{(\ell)}(p_K) \le HU$ and $v_*^{(\ell)}(p_0) \ge HL$, so that $\frac{v_*^{(\ell)}(p_K)}{v_*^{(\ell)}(p_0)} \le \frac{U}{L} = \kappa$, 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 Problem (3) at most $O\left(\ln\left(\kappa|p_0|\right)\right)$ times, where $p_0 = -\frac{\ln N}{\ln(1/\alpha)}$ 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 (8).

To bound the number of oracle calls in LINESEARCH, we will use Lemma A.8.3 that upper bounds the slope of $w(p) := \ln v_*^{(\ell)}(p)$ by $m := \kappa \ln \kappa$. Suppose, for a given $p \geq p_0 = -\frac{\ln N}{\ln(1/\alpha)}$ 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 $v(\pi, a) < \alpha\, v_*(b)$ (otherwise LINESEARCH terminates by step 2).

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

$$0 < b^{(j-1)} - a^{(j-1)} = (1-p)2^{-(j-1)} \leq (1-p_0)2^{-(j-1)} \leq \frac{4\ln N}{\ln(1/\alpha)} 2^{-j}.$$

However, since the algorithm does not terminate in step $j-1$, as discussed, we must also have that

$$v^{(\ell)}(\pi, a^{(j-1)}) < \alpha \cdot v_*^{(\ell)}(b^{(j-1)}). \tag{9}$$

From Lemma A.8, we get that

$$\ln \frac{v_*^{(\ell)}(b^{(j-1)})}{v_*^{(\ell)}(a^{(j-1)})} \leq (\kappa \ln \kappa)(b^{(j-1)} - a^{(j-1)}) \leq \frac{4\kappa(\ln\kappa)(\ln N)}{\ln(1/\alpha)} 2^{-j}. \tag{10}$$

However, from Lemma A.6, we get $v^{(\ell)}(\pi, a^{(j-1)}) \geq \sqrt{\alpha}\, v_*^{(\ell)}(a^{(j-1)})$. Putting these together with eqns. (9) and (10), we get that

$$\frac{1}{2}\ln(1/\alpha) \leq \frac{4\kappa(\ln\kappa)(\ln N)}{\ln(1/\alpha)} 2^{-j}.$$

Therefore,

$$2^j \leq \frac{8\kappa(\ln\kappa)(\ln N)}{(\ln(1/\alpha))^2}.$$

Equivalently, the number of iterations j in LINESEARCH is bounded by

$$O\left(\ln\left(\frac{\kappa \ln N}{\ln(1/\alpha)}\right)\right) = O\left(\frac{(\ln\kappa)(\ln\ln N)}{\ln\ln(1/\alpha)}\right).$$

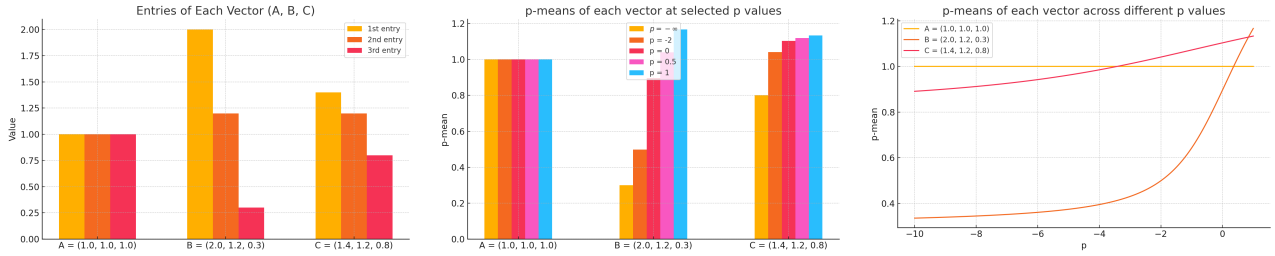
□

A.2. Proof of Proposition 4.2

We prove the proposition assuming the aggregation function $v^{(1)}(\pi, p)$. The same proof applies for $v^{(2)}(\pi, p)$.

Proof.

$$\begin{aligned}
&\max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[f(\mathbf{G}(\tau), q)\right] - \mathbb{E}_{\tau\sim\pi_p}\left[f(\mathbf{G}(\tau), q)\right] \\
&\leq \max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[f(\mathbf{G}(\tau), q)\right] - \mathbb{E}_{\tau\sim\pi_p}\left[f(\mathbf{G}(\tau), p)\right] \\
&= \max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[f(\mathbf{G}(\tau), q)\right] - \max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[f(\mathbf{G}(\tau), p)\right] \\
&\leq \max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[f(\mathbf{G}(\tau), q) - f(\mathbf{G}(\tau), p)\right] \\
&\leq \max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[(q-p)\max_{r\in(p,q)} \frac{\mathrm{d}f(\mathbf{G}(\tau), r)}{\mathrm{d}r}\right] \\
&\leq \max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[(q-p)\kappa\ln\kappa HU\right] \\
&= (q-p)UH\kappa\ln\kappa.
\end{aligned}$$



(a) Entries of A, B, C (b) $p-$means at selected p values. (c) $p-$means across different p values.

Figure 2. Side-by-side comparison of entries and p-means for vectors A, B, C.

The first inequality follows from Lemma 3.2, which ensures monotonicity in the p parameter. The equality comes directly from the definition of π_p, as it maximizes $v^{(1)}(\pi, p)$. The second inequality uses the properties of the maximum function. The next line applies the mean-value theorem to the function $f(\mathbf{G}(\tau), p)$ with respect to the variable p. The subsequent bound follows from the boundedness of the reward function and Lemma A.8. Finally, the terms involving κ are replaced with L and U for conciseness. □

B. Generalized $p-$Means: Illustrative Example

To clarify the meaning behind p-means, we constructed an illustrative example in Figure 2. In this example, three vectors A, B and C represent different policies, where each vector's ith entry is the total reward for the ith stakeholder. Here, A is a balanced policy, B is unbalanced with the largest sum, and C lies in between. Figure 2 (c) plots the p-mean values of these vectors, showing that varying p leads to different optimal policy.

C. Comparison with other MORL Algorithms

(Yu et al., 2024) 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.

(Yang et al., 2019) focuses on weighted linear objectives and learns a single policy, whereas we focus on p-means and compute a portfolio (multiple policies). (Alegre et al., 2023) builds portfolios focusing on weighted linear objectives (as opposed to $p-$means) and does not offer guarantees on portfolio size or oracle complexity. (Reymond et al., 2022) 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 3, we also present comparisons between p-MEANPORTFOLIO and the Generalized Policy Improvement (GPI) algorithm of (Alegre et al., 2023), 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 indicate that our approach achieves significantly better approximation quality. We omit other details of GPI here and refer the interested reader to (Alegre et al., 2023).

D. Additional Experimental Results

Figures 3 and 4 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 4.

Table 3. 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

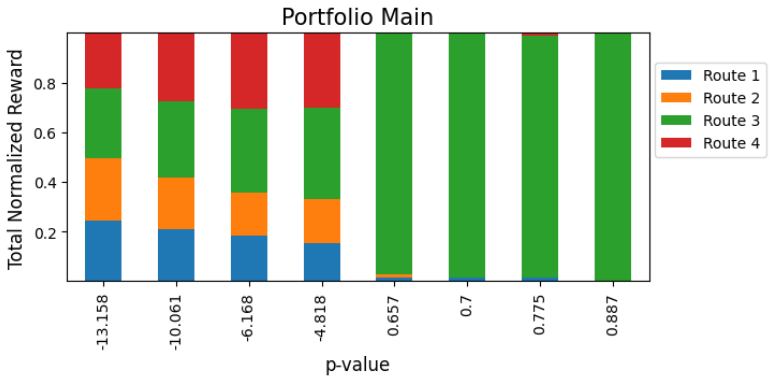


Figure 3. Normalized total reward for each route under different policies in the portfolio generated by p-MEANPORTFOLIO in the taxi environment.

E. Experimental Details

We include the details of various experiments here.

E.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 environment setup of (Fan et al., 2022), 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: $source\_coords = [[0,0],[0,5],[3,0],[1,0]]$, $destination\_coords = [[1,5],[5,0],[3,3],[0,3]]$

MDP Structure

STATE SPACE (S)

The state space consists of information on the location of the taxi, location of passengers and whether passengers have been picked up by the taxi at the moment.

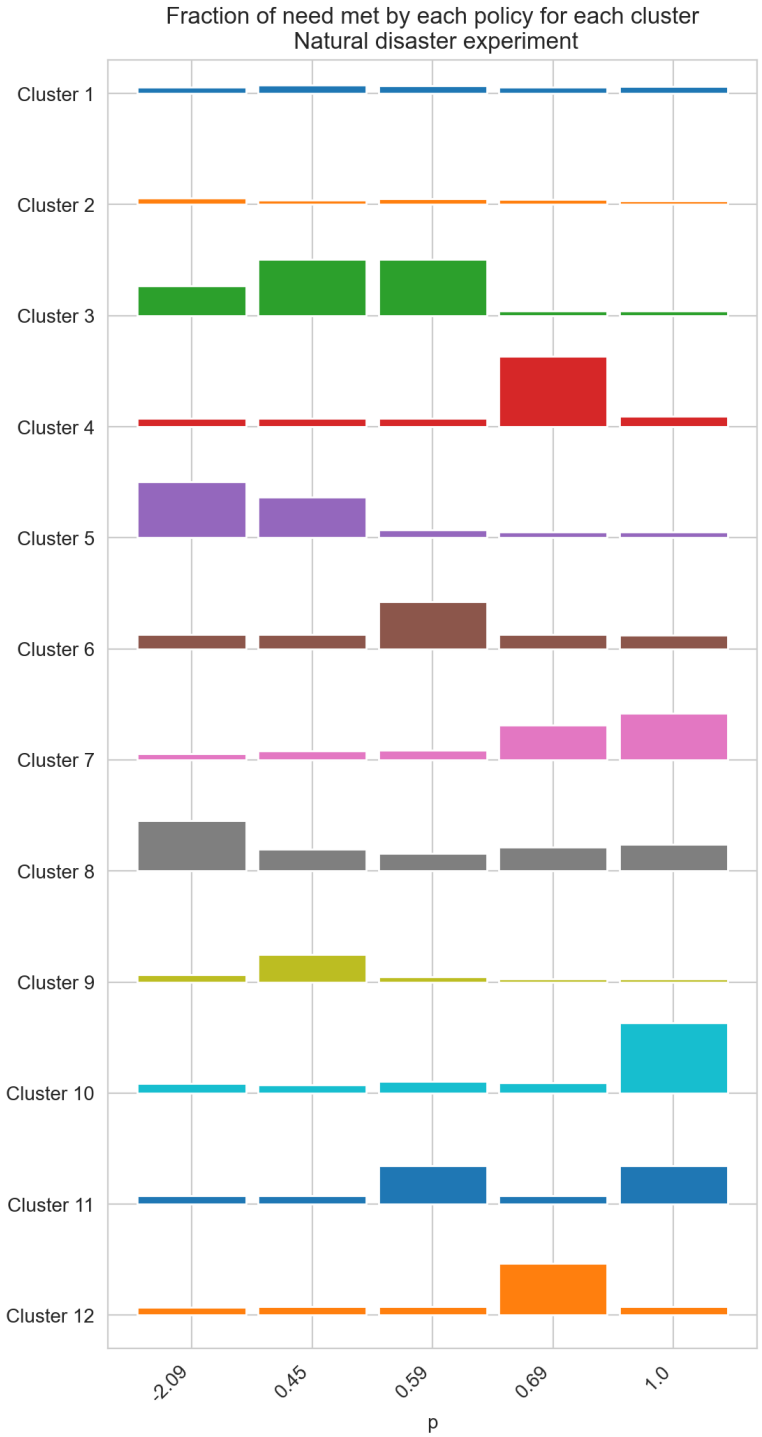


Figure 4. Fraction of need meet for various clusters by various policies in the portfolio obtained by p-MEANPORTFOLIO after $H = 4$ intervention steps for the natural disaster experiment. Note that different solutions in the portfolio emphasize different clusters.

Table 4. 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	-15.29, -2.054, 0.517, 0.758
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

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 reward 0 for moving, -10 reward for taking invalid action of picking or dropping a passenger at wrong coordinating, and 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:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau\sim\pi}\left[f\big(\mathbf{G}(\tau), p\big)\right]. \tag{11}$$

We learn this policy using the Welfare Q-Learning algorithm proposed in (Fan et al., 2022). For every problem setting, we train the policy for 200 episodes. Every episode is a finite horizon problem with 1000 timesteps. We consider discount factor gamma as 0.99.

E.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 ($N = 12$). 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).

MDP Structure
The problem is formulated as an MDP $\mathcal{M}$, with the following components:

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

Table 5. Clustered population data including density, proximity, income level, total population, and initial need.

STATE SPACE (S)
The state $s = (s_1, s_2, \ldots, s_N)$ represents the current resource needs of the N clusters. Each s_c denotes the total remaining need for cluster c. The state space is discretized, with each s_c taking values in increments of b in $[0, B]$. In our setting, $b = 50$ and $B = 150$.

ACTION SPACE (A)
Actions $a = (a_1, a_2, \ldots, a_N)$ represent the allocation of resources to clusters, where: $a_c \in \{0, b, 2b, \ldots, B\}$ represents the resources allocated to cluster c and the total allocation $\sum_{c=1}^{N} a_c$ cannot exceed the total budget B.

TRANSITION PROBABILITY MATRIX (TPM, $P(s'|s, a)$)
The state transitions depend on the current state s and the action a. If $a_c \geq s_c$, $s'_c = 0$ with probability 1 (this cluster's need is fully met). If $a_c < s_c$ the unmet need will reduce by the allocation, that is $s'_c = s_c - a_c$, with 70% probability. However, there is a 30% probability that the cluster will observe an increase in need by b units, meant to reflect ballooning needs when immediate action is not taken.

GENERATION OF POLICIES & REWARD FUNCTIONS
We consider combinations of "reasonable" policies a decision-maker might take in this setting. First, a valid policy would not allocate resources to a zero-need cluster. Secondly, we consider the following priorities that a decision-maker might have when allocating funds among nonzero unmet need clusters: (1) lowest income, (2) highest population, (3) highest unmet need, (4) highest unmet need per capita, (5) high population density regions, and (6) far from critical infrastructure. In every time period, for every feasible state, we apply one of these policies, a random convex weighting of these policies, or a randomized priority to generate 10,000 different feasible policies.

The expected reward accrued by each cluster is the average of the sum of the expected fraction of the initial need met under a policy and the fraction of the expected total allocation awarded to that cluster over the horizon.

E.3. Healthcare Intervention

The Healthcare Intervention problem is based on the large-scale mobile-health program run by the NGO ARMMAN (ARMMAN, 2024). 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 (Verma et al., 2023). Further, for our experiments, we sample data for 2100 beneficiaries, and run the experiment for $H = 12$ timesteps.

Generation of Policies & 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 (Verma et al., 2024a) to derive reward functions from natural-language commands. Given a command specifying Npreferences (each indicating a subgroup to prioritize) this method generates two sets of reward functions:

1. **Individual preferences:** one reward function per preference (total of N reward functions) which we treat as each stakeholder's reward function.
2. **Balancing functions:** a collection of reward functions that trade off among the N preferences, whose corresponding policies form our feasible set Π.

In our experiments, we set $N = 59$ and construct $|\Pi| = 200$ balancing policies as feasible set.

Learning Policy

It is computationally intractable to optimally solve the RMAB problem (Papadimitriou and Tsitsiklis, 1999). Thus, we use the popular Whittle Index Heuristic (Whittle, 1988) 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 highest whittle indices within the budget limit.

Baseline Policy

The baseline policy mentioned in Figure 1 is trained on the default reward.

E.4. Other Details

Initial p for BUDGET-CONSTRAINEDPORTFOLIO

For BUDGET-CONSTRAINEDPORTFOLIO, 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 $\alpha \in \{0.05, 0.10, 0.15, \ldots, 0.90, 0.95, 0.99\}$ among which the resulting portfolio has size K.