

# OPTIMAL SCALARIZATIONS FOR PROVABLE MULTI-OBJECTIVE OPTIMIZATION

**Anonymous authors**

Paper under double-blind review

## ABSTRACT

Linear scalarization is a simple and widely-used technique that can be deployed in any multiobjective setting to combine diverse objectives into one reward function, but such heuristics are not theoretically understood. To that end, we perform a case study of the multiobjective stochastic linear bandits framework with  $k$  objectives and our goal is to provably scalarize and explore a diverse set of optimal actions on the Pareto frontier, as measured by the dominated hypervolume. Even in this elementary convex setting, the choice of scalarizations and weight distribution surprisingly affects performance, and the natural use of linear scalarization with uniform weights is suboptimal due to a non-uniform Pareto curvature. Instead, we suggest the usage of the theoretically-inspired hypervolume scalarizations with non-adaptive uniform weights, showing that it comes with novel hypervolume regret bounds of  $\tilde{O}(dT^{-1/2} + T^{-1/k})$ , with optimal matching lower bounds of  $\Omega(T^{-1/k})$ . We support our theory with strong empirical performance of the hypervolume scalarization that consistently outperforms both the linear and Chebyshev scalarizations in high dimensions.

## 1 INTRODUCTION

Optimization objectives are becoming more complex with many different components that must be combined to perform precise tradeoffs in machine learning models. Starting from standard  $\ell_p$  regularization objectives in regression problems (Kutner et al., 2005) to increasingly multi-component losses used in reinforcement learning (Sutton et al., 1998) and deep learning (LeCun et al., 2015), many of these single-objective problems are phrased as a scalarized form of an inherently multiobjective problem.

In addition, practitioners often vary the weights of the scalarization method, with the main goal of exploring the entire *Pareto frontier*, the set of optimal outputs with no way to improve on all objectives simultaneously. Specifically, for some weights  $\lambda \in \mathbb{R}^k$ , we have scalarization functions  $s_\lambda(y) : \mathbb{R}^k \rightarrow \mathbb{R}$  that convert  $k$  multiple objectives  $F(a) := (f_1(a), \dots, f_k(a))$  over some parameter space  $a \in \mathcal{A} \subseteq \mathbb{R}^d$  into a single-objective scalar. Optimization is then applied to this family of single-objective functions  $s_\lambda(F(x))$  for various  $\lambda$  and since we construct  $s_\lambda$  to be monotonically increasing in all coordinates,  $x_\lambda = \arg \max_{a \in \mathcal{A}} s_\lambda(F(a))$  is on the Pareto frontier and the various choices of  $\lambda$  recovers an approximation to the Pareto frontier (Paria et al., 2018). Due to its simplicity of use, many multi-objective optimization algorithms use a heuristic-based scalarization strategy to pick the scalarizer and weights, which efficiently splits the multi-objective optimization into numerous single "scalarized" optimizations (Roijers et al., 2013).

While there are recent developments in specific multi-objective algorithms tailored to specific settings such as ParEgo (Knowles, 2006) and MOEAD (Zhang and Li, 2007) for black-box optimization or multivariate iteration for reinforcement learning (Yang et al., 2019), the appeal of scalarization still remains as it generalizes to any optimization setting. For example, on the MUJOCO environment, multi-objective reinforcement learning is often done on a combined reward function that simply linearly penalizes the task reward with the negative action norm, as human-like behaviors are usually efficient and low-energy (Abdolmaleki et al., 2021). Another example is performing neural architecture search on loss functions that non-linearly combine possibly conflicting objectives, such as accuracy and network size (Chen et al., 2020).

However, despite the wide usage, the choice of weight distribution, reference point, and even the scalarization functions themselves is diverse and largely various between different papers (Paria et al., 2018; Nakayama et al., 2009; Zhang and Li, 2007). In fact, the popularly used linear scalarization is known to be suboptimal in non-convex settings (Boyd and Vandenberghe, 2004; Emmerich and Deutz, 2018) and cannot recover a concave Pareto frontier. For multi-armed bandits, scalarized knowledge gradient methods empirically perform better with non-linear scalarizations (Yahya et al., 2014). Recently, some works have come up with novel scalarizations that perform better empirically (Aliano Filho et al., 2019; Schmidt et al., 2019) and others have tried to do comparisons between different scalarizations with varying conclusions (Kasimbeyli et al., 2019). Some have also proposed adaptively weighted approaches that have connections to gradient-based multi-objective optimization (Lin et al., 2019), while other works have proposed piecewise linear scalarizations inspired by economics (Busa-Fekete et al., 2017).

In this paper, we seek to understand, from a theoretical perspective, the optimal choice of scalarizations and corresponding weights distribution to perform multiobjective optimization. To perform rigorous analysis, we focus on the classical *stochastic linear bandit* problem in the multiobjective setting (Lattimore and Szepesvári, 2020). For the single objective case, the standard average cumulative regret bound for this problem is  $O(d/\sqrt{T})$ , and this is known to be tight in the worst case (Dani et al., 2008). For the multi-objective case, the notion of optimality becomes varied. Previous work by (Lu et al., 2019) proved Pareto regret bounds of  $O(d\sqrt{T})$ , but that only guarantees recovery of a single point close to the Pareto frontier. Some minimize a notion of distance to the Pareto frontier, such as the  $\ell_\infty$  norm (Auer et al., 2016), although such approaches work in the finite multi-arm bandit setting which mandates at least one pull of each arm. Paria et al. (2018) provides a Bayes regret bound with respect to a scalarization-induced regret, but it is unclear how to choose the right scalarization or weight distribution.

In recent years, a natural and widely used metric to measure progress is the *hypervolume indicator*, which is the volume of the dominated portion of the Pareto set (Zitzler and Thiele, 1999; Shah and Ghahramani, 2016). The hypervolume metric has become a gold standard because it has strict Pareto compliance meaning that if set  $A$  is a subset of  $B$  and  $B$  has at least one Pareto point not in  $A$ , then the hypervolume of  $B$  is greater than that of  $A$ . Zuluaga et al provides sub-linear hypervolume regret bounds; however, they are exponential in  $k$  and its analysis only applies to a specially tailored algorithm that requires an unrealistic classification step (Zuluaga et al., 2013). Most relevant is recent work by (Golovin and Zhang, 2020) that introduces random hypervolume scalarizations and when combined with standard generalization bounds, one can directly derive a  $O(kd/\sqrt{T} + T^{-1/O(k)})$  convergence bound for multiobjective linear bandits by using a linear kernel.

## 1.1 OUR CONTRIBUTIONS

As our theoretical toy model, we consider the classic *stochastic linear bandit* setting. For the single-objective setting, in round  $t = 1, 2, \dots, T$ , the learner chooses an action  $a_t \in \mathbb{R}^d$  from the action set  $\mathcal{A}$  and receives a reward  $y_t = \langle \theta^*, a_t \rangle + \xi_t$  where  $\xi_t$  is i.i.d. 1-sub-Gaussian noise and  $\theta^* \in \mathbb{R}^d$  is the unknown true parameter vector. In the *multi-objective stochastic linear bandit* setting, the learner instead receives a vectorized reward  $y_t = \Theta^* a_t + \xi_t$ , where  $\Theta^* \in \mathbb{R}^{k \times d}$  is now a matrix of  $k$  true parameters and  $\xi_t \in \mathbb{R}^k$  is a vector of independent 1-sub-Gaussian noise. We also denote  $\mathbf{A}_t \in \mathbb{R}^{d \times t}$  to be the history action matrix, whose  $i$ -th column is  $a_i$ , the action taken in round  $i$ . Similarly,  $\mathbf{y}_t$  is defined analogously.

For any scalarization and weight distribution, we propose a new algorithm (Algorithm 1) for multi-objective stochastic linear bandit that combines uniform exploration and exploitation via an UCB approach to provably obtain scalarized Bayes regret bounds, which we then combine with the hypervolume scalarization to derive optimal hypervolume regret bounds. Specifically, for any scalarization  $s_\lambda$ , we show that our algorithm in the linear bandit setting has a scalarized Bayes regret bound of  $\tilde{O}(L_p k^{1/p} d T^{-1/2} + T^{-1/(k+1)})$ , where  $L_p$  is the Lipschitz constant of the  $s_\lambda(\cdot)$  in the  $\ell_p$  norm. By introducing a non-Euclidean analysis, we reduce and even completely remove the dependence on the number of objectives,  $k$ , which had a polynomial dependence in previous regret bounds.

We emphasize that our novel measure of progress is neither the simple regret nor cumulative regret, and is in fact even different from the Bayes regret definition in previous literature. However, it is arguably the most realistic setting of regret as it generalizes to the multi-objective setting in the form

of hypervolume regret, which is a measure of the hypervolume gap between the Pareto frontier and the set of points found by the optimizer. Indeed, by applying our improved scalarized Bayes regret bounds with hypervolume scalarizations, we derive improved hypervolume regret bounds.

**Theorem 1** (Informal Restatement of Theorem 12). *Let  $\mathbf{A}_T \subseteq \mathcal{A}$  be the actions generated by  $T$  rounds of Algorithm 1, then our hypervolume regret is bounded by:*

$$\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_T) \leq O \left( d \sqrt{\frac{\log(T)}{T}} + \frac{1}{T^{1/(k+1)}} \right)$$

We note that even for mild values of  $k$ , the dominating term in the convergence bound is the  $O(T^{-1/k})$  term, which follows from controlling the metric entropy for  $L$ -Lipschitz functions in  $\mathbb{R}^k$ . We prove novel lower bounds showing one cannot hope for a better convergence rate due to the exponential nature of our regret. Specifically, we show that the hypervolume regret of any algorithm after  $T$  actions is at least  $\Omega(T^{-1/(k-1)})$ , demonstrating the necessity of the  $O(T^{-1/k})$  term up to small constants in the denominator. As a corollary, we leverage the properties of hypervolume scalarization to show that our scalarized Bayes regret bounds are tight.

**Theorem 2** (Informal Restatement of Theorem 13). *Any algorithm that outputs a set of  $T$  actions  $\mathbf{A}_T$  must suffer hypervolume regret of at least*

$$\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_T) = \Omega(T^{-1/(k-1)})$$

Guided by our theoretical analysis, we empirically evaluate a diverse combination of scalarizations and weight distributions with our proposed algorithm for multiobjective linear bandits. Our experiments show that for some settings of linear bandits, in spite of a convex Pareto frontier, applying linear or Chebyshev scalarizations naively with various weight distributions leads to suboptimal hypervolume progress, especially when the number of objective increase to exceed  $k \geq 5$ . This is because the non-uniform curvature of the Pareto frontier, exaggerated by the curse of dimensionality and combined with a stationary weight distribution, hinders uniform progress in exploring the frontier. Although one can possibly adapt the weight distribution to the varying curvature of the Pareto frontier, we suggest remediating the issue by simply adopting the use of non-linear scalarizations that are more robust to the choice of weight distribution and are theoretically sound.

We acknowledge that while there might be better performing multi-objective algorithms for this specific linear bandit setting, such as Gaussian Processes (Williams and Rasmussen, 2006), we focus only on comparing scalarizing algorithms as to demonstrate the general utility of hypervolume scalarization as the optimal choice in various other settings, such as reinforcement learning. From our analysis on this toy example, we recommend the use of hypervolume scalarizations, even when the optimization is convex, as an efficient non-adaptive method to perform general multiobjective optimization in high dimensions. We believe that this is especially relevant given the modern era of learning algorithms that incorporates multiple objectives such as fairness, privacy, latency.

We summarize our contributions as follows:

- Introduce optimization algorithm for multiobjective linear bandits that achieves  $\tilde{O}(dT^{-1/2} + T^{-1/k})$  hypervolume regret via a novel non-Euclidean regret analysis and metric entropy.
- Establish a tight lower bound on the hypervolume regret and Bayes regret of  $\Omega(T^{-1/k})$  by introducing a packing argument on the Pareto set.
- Empirically study the choice of scalarizations and weight distributions for finding a diverse Pareto frontier in multi-objective linear bandits, promoting the adoption or experimentation of hypervolume scalarizations in other arenas of multi-objective optimization, especially when the number of objectives is high.

## 2 PROBLEM SETTING AND NOTATION

We assume, for sake of normalization, that  $\|\Theta_i^*\| \leq 1$  and that  $\|a_t\| \leq 1$ , where  $\|\cdot\|$  denotes the  $\ell_2$  norm unless otherwise stated. Other norms that are used include the classical  $\ell_p$  norms  $\|\cdot\|_p$  and

matrix norms  $\|x\|_{\mathbf{M}} = x^\top \mathbf{M} x$  for a positive semi-definite matrix  $\mathbf{M}$ . For a scalarization function  $s_\lambda(x)$ , we will define smoothness with respect to the input, and analogously for  $\lambda$ . We say that  $s_\lambda$  is  $L_p$ -Lipschitz with respect to the  $\ell_p$  norm on  $\mathcal{X}$  if for  $x_1, x_2 \in \mathcal{X}$ ,  $|s_\lambda(x_1) - s_\lambda(x_2)| \leq L_p \|x_1 - x_2\|_p$ . We let  $\mathcal{S}_+^{k-1} = \{y \in \mathbb{R}^k \mid \|y\| = 1, y > 0\}$  be the sphere restricted to the positive orthant and by abuse of notation, we also let  $y \sim \mathcal{S}_+^{k-1}$  denote that  $y$  is drawn uniformly on  $\mathcal{S}_+^{k-1}$ . Our usual settings of the weight distribution  $\mathcal{D} = \mathcal{S}_+^{k-1}$  will be uniform, unless otherwise stated.

We want to study the optimal choices of scalarization and weight distribution that generally maximizes the expected rewards of each objective  $y_i = (\Theta_i^* a)$  for  $i = 1, \dots, k$ . Unlike single-objective optimization, multi-objective optimization usually aims to discover the Pareto frontier of the problem. For two outputs  $y, z \in \mathbf{Y} \subseteq \mathbb{R}^k$ , we say that  $y$  is *Pareto-dominated* by  $z$  if  $y_i \leq z_i$  for all  $i$  and there exists  $j$  such that  $y_j < z_j$ . A point is *Pareto-optimal* if no point in the output space  $\mathbf{Y}$  can dominate it. Let  $\mathbf{Y}^*$  denote the set of Pareto-optimal points in  $\mathbf{Y}$ , which is also known as the *Pareto frontier*.

Our main progress metrics for multiobjective optimization is given by the standard hypervolume indicator. For  $S \subseteq \mathbb{R}^k$  compact, let  $\text{vol}(S)$  be the regular hypervolume of  $S$  with respect to the standard Lebesgue measure.

**Definition 3.** For  $Y \subseteq \mathbb{R}^k$ , we define the (dominated) **hypervolume indicator** of  $Y$  with respect to reference point  $z$  as:

$$\mathcal{HV}_z(Y) = \text{vol}(\{x \mid x \geq z, x \text{ is dominated by some } y \in Y\})$$

Therefore, for a finite set  $Y$ ,  $\mathcal{HV}_z(Y)$  can be viewed as the hypervolume of the union of the dominated hyper-rectangles for each point  $y_i \geq z$  that has one corner at  $z$  and the other corner at  $y_i$ . Note that our definition also holds for non-finite set as a limiting integral. We can formally phrase our optimization as trying to rapidly minimize the hypervolume (psuedo-)regret:  $\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_T)$ , which is 0 if and only if  $\mathbf{A}_T$  contains the Pareto frontier.

To study a diversity of scalarizations and weight distributions, an important measure of progress that attempts to capture the requirement of diversity in the Pareto front is scalarized Bayes regret for some scalarization function  $s_\lambda$ . For some fixed scalarization with weights  $\lambda$ ,  $s_\lambda : \mathbb{R}^k \rightarrow \mathbb{R}$ , we can define the instantaneous scalarized (psuedo-)regret as

$$r(s_\lambda, a_t) = \max_{a \in \mathcal{A}} s_\lambda(\Theta^* a) - s_\lambda(\Theta^* a_t)$$

Since the scalarized regret is only a function of a single action  $a_t$ , it fails to capture the variety of solutions that we would optimize for in the Pareto frontier. To capture some notion of diversity, we must define progress with respect to a set of past actions  $\mathbf{A}_t$ . Generalizing the scalarized regret above, we can formulate the Bayes regret as an average of the scalarized regret over some distribution of non-negative weight vectors,  $\lambda \sim \mathcal{D}$ . Specifically, we define the (*scalarized*) *Bayes regret* with respect to a set of actions  $\mathbf{A}_t$  to be:

$$BR(s_\lambda, \mathbf{A}_t) = \mathbf{E}_{\lambda \sim \mathcal{D}} [\max_{a \in \mathcal{A}} s_\lambda(\Theta^* a) - \max_{a \in \mathbf{A}_t} s_\lambda(\Theta^* a)] = \mathbf{E}_{\lambda \sim \mathcal{D}} [\min_{a \in \mathbf{A}_t} r(s_\lambda, a)]$$

Unlike previous notions of Bayes regret in literature, we are actually calculating the Bayes regret of a reward function that is maximized with respect to an entire set of actions  $\mathbf{A}_t$ . Specifically, by maximizing over all previous actions, this captures the notion that during multi-objective optimization our Pareto set is always expanding. We will see later that this novel definition is the right one, as it generalizes to the multi-objective setting in the form of hypervolume regret.

Finally, since our action set  $\mathcal{A}$  is generic, we need some prior knowledge on  $\mathcal{A}$  to allow for efficient implementation of "uniform" exploration. Therefore, we assume that  $\mathcal{A}$  contains an isotropic set of actions and specifically, there is  $\mathcal{E} \subset \mathcal{A}$  with size  $|\mathcal{E}| = O(d)$  such that  $\sum_i e_i e_i^\top \succeq \frac{1}{2} \mathbf{I}$ , where  $\succeq$  denotes the PSD ordering on symmetric matrices. We note that this assumption is not restrictive, as the assumption and analysis can be fully relaxed by the study of optimal design for least squares estimators (Lattimore and Szepesvári, 2020) and the Kiefer-Wolfowitz Theorem (Kiefer and Wolfowitz, 1960), which guarantees the existence and construction of an uniform exploration basis of small size.

## 2.1 SCALARIZATIONS FOR MULTIOBJECTIVE OPTIMIZATION

For multiobjective optimization, we generally only consider *monotone* scalarizers that have the property that if  $y > z$ , then  $s_\lambda(y) > s_\lambda(z)$  for all  $\lambda$ . Note this implies that an unique optimal solution

to the scalarized optimization is on the Pareto frontier. A common scalarization used widely in practice is the linear scalarization:  $s_\lambda(y) = \lambda^\top y$  for some chosen positive weights  $\lambda \in \mathbb{R}^k$ . By Lagrange duality and hyperplane separation of convex sets, one can show that any convex Pareto frontier can be characterized fully by an optimal solution to  $\max_y s_\lambda(y)$  for some weights.

**Proposition 4.** *For any point  $y^*$  on the Pareto frontier of a convex set  $\mathcal{Y}$ , there exists  $\lambda > 0$  such that  $y^* = \arg \max_{y \in \mathcal{Y}} \lambda^\top y$ .*

However, it is known that linear scalarizations cannot recover the non-convex regions of Pareto fronts since the linear structure of the level curves can only be tangent to the Pareto front in the protruding convex regions (see Figure 1). To overcome this drawback, another scalarization that is proposed is the Chebyshev scalarization:  $s_\lambda(y) = \min_i \lambda_i y_i$ . Indeed, one can show that the sharpness of the scalarization, due to its minimum operator, can discover non-convex Pareto frontiers (for more details, see Emmerich and Deutz (2018)).

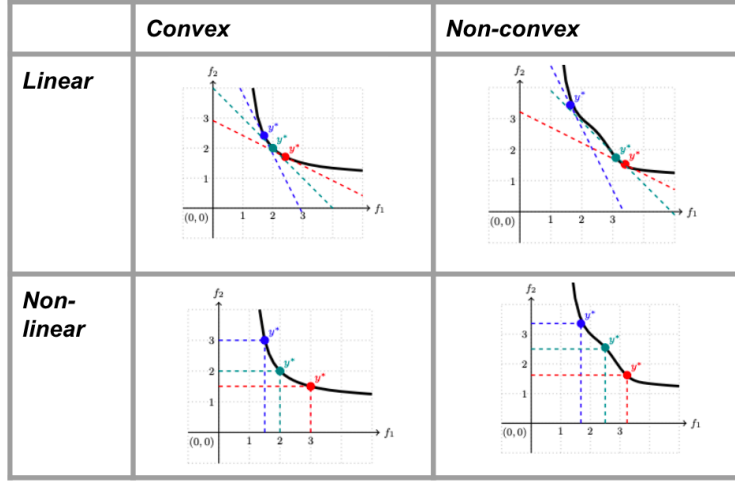


Figure 1: Comparisons of the scalarized minimization solutions with various weights for a multiobjective problem with convex and non-convex Pareto fronts. The colors represent different weights; the dots are scalarized optima and the corresponding dotted lines represent level curves of the linear and non-linear scalarization, respectively. Linear scalarization does not have an optima in the concave region of the Pareto front for any set of weights, but the non-linear scalarization, with its sharper level curves, can discover the whole Pareto front. Plots reworked from Emmerich and Deutz (2018).

**Proposition 5.** *For any point  $y^*$  on the Pareto frontier of any set  $\mathcal{Y}$  that lies in the positive orthant, there exists  $\lambda > 0$  such that  $y^* = \arg \max_{y \in \mathcal{Y}} \min_i \lambda_i y_i$ .*

Next, we introduce a related scalarization known as the hypervolume scalarization,  $s_\lambda(y) = \min_i (y_i / \lambda_i)^k$ . This scalarization has the special property that the expected scalarized value under a uniform weight distribution on  $S_+^{k-1}$  gives the dominated hypervolume, up to a constant scaling factor.

**Lemma 6** (Golovin and Zhang (2020)). *Let  $Y = \{y_1, \dots, y_m\}$  be a set of  $m$  points in  $\mathbb{R}^k$ . Then, the hypervolume of  $Y$  with respect to a reference point  $z$  is given by:*

$$\mathcal{HV}_z(Y) = c_k \mathbf{E}_{\lambda \sim S_+^{k-1}} \left[ \max_{y \in Y} s_\lambda(y - z) \right]$$

where  $s_\lambda(y) = \min_i (\max(0, y_i / \lambda_i))^k$  and  $c_k = \frac{\pi^{k/2}}{2^k \Gamma(k/2 + 1)}$  constant depending only on  $k$ .

Intuitively, this lemma says that the optima of the hypervolume scalarization over some uniform choice of weights will be diverse enough so as to capture the total dominated hypervolume of the Pareto set. Furthermore, note that our scalarization function, similar to the Chebyshev scalarization, is a minimum over coordinates and there is a one-to-one correspondence between the optima of the hypervolume and Chebyshev scalarizations. To see this, note that since  $f(x) = x^k$  is a monotone transform, we can consider an equivalent scalarization (in terms of optima) that is given by  $s_\lambda^{1/k} = \min_i y_i / \lambda_i$ . From this, we make the novel observation that the hypervolume scalarization is similar to the Chebyshev scalarization but with inverse weights, which does affect the optimization performance. For completeness, we show that the optima of hypervolume scalarization also spans the whole Pareto front.

**Lemma 7.** *For any point  $y^*$  on the Pareto frontier of any set  $\mathcal{Y}$  that lies in the positive orthant, there exists  $\lambda > 0$  such that  $y^* = \arg \max_{y \in \mathcal{Y}} \min_i (y_i / \lambda_i)^k$ .*

### 3 MULTIOBJECTIVE STOCHASTIC LINEAR BANDITS

Upon first glance, maximizing for hypervolume in multiobjective linear bandits seems straightforward via the greedy approach of finding the next point that maximizes the hypervolume gain. However, because our observations are inherently noisy, it becomes hard to even statistically infer an unbiased value of the hypervolume measure, let alone trying to optimize for the hypervolume gain. Therefore, even in this simple setting, we turn to scalarized algorithms and present an algorithm that comes with strong convergence guarantees for some generic fixed scalarization  $s_\lambda$ , in terms of the scalarized Bayes regret.

To get an intuition for the final algorithm, first consider the  $k = 1$  case. In that case, our scalarization is essentially no-op and our regret is given by for our sole parameter  $\theta^*$ :  $BR(s_\lambda, \mathbf{A}_t) = \max_{a \in \mathcal{A}} \langle \theta^*, a \rangle - \max_{a \in \mathbf{A}_t} \langle \theta^*, a \rangle = \min_t \langle \theta^*, a^* \rangle - \langle \theta^*, a_t \rangle$

Note that the Bayes regret in the 1-D setting reduces to what is commonly known as the simple regret, although this version of simple regret is even more robust to exploration as it is minimized across all time steps  $T$ . In this setting, also known as pure exploration or best arm identification, it is shown that playing a uniform exploration strategy (or some precomputed  $G$ -optimal design) does quite competitively (Soare et al., 2014). In fact, one can show an information-theoretic lower bound that matches the established upper bound of  $O(d/\sqrt{T})$  that is achieved from uniform exploration (Jedra and Proutiere, 2020), suggesting that the dependence on  $d, T$  cannot be fundamentally improved in our upper bounds.

In higher dimensions, a uniform exploration strategy does not naively work. This is because we cannot simply output one best answer at the end but we need to output a diverse set of optima that closely represent the Pareto frontier. Yet we can take advantage of the non-increasing nature of Bayes regret to show that it often does hurt to explore, and it turns out that a simple strategy of evenly balancing exploration and exploitation is optimal. Furthermore, since each objective is essentially independent and provides a separate observed reward, we provide a careful analysis of the algorithm can provide regret bounds that do not need to inherently scale with  $k$  in the cumulative scalarized regret.

By using the confidence ellipsoids given by the UCB algorithm, we can determine each objective parameter  $\Theta_i^*$ , up to a small error. Then, we can bound the scalarized regret as a function by controlling the smoothness of the scalarization function with respect to small changes in each objective. Naively doing this would be a  $O(k)$  regret bound since each objective has about  $O(d/\sqrt{T})$  uncertainty; however, by using a  $\ell_p$  analysis, we can reduce the dependence on  $k$  when  $p$  is large. Indeed, for the Chebyshev and the hypervolume scalarizations, which have very sharp level sets, it turns out setting  $p = \infty$  gives the best regret analysis and eliminates the dependence on  $k$ .

**Lemma 8.** *Consider running EXPLOREUCB (Algorithm 1) for  $T > \max(k, d)$  iterations and for  $T$  even, let  $a_T$  be the action that maximizes the scalarized UCB in iteration  $T/2$ . Then, with probability at least  $1 - \delta$ , the instantaneous scalarized regret can be bounded by*

$$r(s_\lambda, a_T) \leq 10k^{1/p} L_p d \sqrt{\frac{\log(k/\delta) + \log(T)}{T}}$$

where  $L_p$  is the  $\ell_p$ -Lipschitz constant for  $s_\lambda(\cdot)$ .

**Algorithm 1:** EXPLOREUCB( $T, \mathcal{D}, s_\lambda$ ): Scalarized UCB for Linear Bandits**Input :** number of maximum actions  $T$ , weight distribution  $\mathcal{D}$ , scalarization  $s_\lambda$ 


---

```

1 Initialize iteration counter  $n = 1$ 
2 repeat
  // Play uniform exploration
3   Play action  $e_i \in \mathcal{E}$  for  $i \equiv n \pmod d$ 
4   Sample  $\lambda \sim \mathcal{D}$  independently
  // Define UCB as in Lemma 15
5   Let  $C_{ti}$  be the confidence ellipsoid for  $\Theta_i$  and let  $UCB_i(a) = \max_{\theta \in C_i} \theta^\top a$ 
  // Play scalarized UCB maximizer
6   Play action that maximizes  $a^* = \operatorname{argmax}_{a \in \mathcal{A}} s_\lambda(UCB_i(a))$ 
7   Increment  $n \leftarrow n + 1$ 
8 until number of actions exceed  $T$ 

```

---

Finally, to connect the expected Bayes regret with the empirical average of scalarized regret, we must show that the empirical running average concentrates uniformly to the mean across all functions of the form  $f(\lambda) = \max_{a \in \mathcal{A}} s_\lambda(\Theta^* a)$ . By appealing to standard bounds on Rademacher complexities for

Lipschitz function classes, we derive a  $O(T^{-1/(k+1)})$  generalization bound.

**Theorem 9.** Assume that for any  $a \in \mathcal{A}$ ,  $|s_\lambda(\Theta^* a)| \leq B$  for some  $B$  and  $s_\lambda$  is  $L_\lambda$ -Lipschitz with respect to the  $\ell_2$  norm in  $\lambda$ . With constant probability, the Bayes regret of running Algorithm 1 at round  $T$  can be bounded by

$$BR(s_\lambda, \mathbf{A}_T) \leq O \left( k^{1/p} L_p d \sqrt{\frac{\log(kT)}{T}} + \frac{BL_\lambda}{T^{1/(k+1)}} \right)$$

### 3.1 CONNECTION TO HYPERVOLUME REGRET

In this section, we apply our general scalarized Bayes regret bounds onto our different choices of scalarizations and for the hypervolume scalarization, we will use its unique property to translate Bayes regret bounds into hypervolume regret bounds. We utilize the fact that if  $s_\lambda$  is differentiable everywhere except for a finite set, bounding Lipschitz constants is equivalent to bounding the dual norm  $\|\nabla s_\lambda\|_q$ , where  $1/p + 1/q = 1$ , which follows from mean value theorem, which we state as Proposition 10.

**Proposition 10.** Let  $f : \mathcal{X} \rightarrow \mathbb{R}$  be a continuous function that is differentiable everywhere except on a finite set, then if  $\|\nabla f(x)\|_q \leq L_p$  for all  $x \in \mathcal{X}$ ,  $f(x)$  is  $L_p$ -Lipshitz with respect to the  $\ell_p$  norm.

Proving smoothness properties of our hypervolume scalarizations for any  $\lambda > 0$  with  $\lambda$  normalized on the unit sphere is non-obvious as  $s_\lambda(y)$  depends inversely on  $\lambda_i$  so when  $\lambda_i$  is small,  $s_\lambda$  might change very fast. However,  $\lambda_i$  being small makes it unlikely that it becomes the minimum coordinate, implying that it is not contributing to the scalarized value or its rate of change. We formalize this intuition below with a given range bound on the values of  $y$ .

**Lemma 11.** Let  $s_\lambda(y) = \min_i (y_i / \lambda_i)^k$  be the hypervolume scalarization with  $\|\lambda\| = 1$  and  $0 < B_l \leq y_i \leq B_u$ . Then, we may bound  $L_p \leq \frac{B_u^k}{B_l k^{k/2-1}}$  and  $L_\lambda \leq \frac{B_u^{k+1}}{B_l k^{(k-1)/2}}$  and  $|s_\lambda| \leq \frac{B_u^k}{k^{k/2}}$ .

**Theorem 12.** Let  $z \in \mathbb{R}^k$  be a reference point such that over all  $a \in \mathcal{A}$ ,  $B = \min_a \Theta^* a - z$  is positive. Then, with constant probability, running Algorithm 1 with  $s_\lambda(y)$  as the hypervolume scalarization and with  $\mathcal{D}$  as the uniform distribution on  $S_+$  gives hypervolume regret bound of

$$\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_T) \leq O \left( c_k \frac{(B+2)^k}{B_l k^{k/2-1}} d \sqrt{\frac{\log(kT)}{T}} + c_k \frac{(B+2)^{2k+1}}{B_l k^{k-1/2} T^{1/(k+1)}} \right)$$

For  $k$  held constant, this becomes

$$\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_T) \leq O \left( d \sqrt{\frac{\log(T)}{T}} + \frac{1}{T^{1/(k+1)}} \right)$$

### 3.2 LOWER BOUND

The dominating factor in convergence upper bound is the  $O(T^{-1/(k+1)})$  dependence that seems to be quite rather slow even for a mild values of  $k$ . This terms comes from the generalization error term when empirically approximating the scalarized Bayes regret in high dimensions with  $T$  samples, in contrast to the noise error term that decays much faster at a rate of  $O(d/\sqrt{T})$ . By constructing a lower bound for with hypervolume scalarizations via a packing argument, we show that this dependence on  $k$  in the convergence term is unavoidable for Bayes regret.

Specifically, we show that for hypervolume regret, any algorithm cannot achieve better than  $O(T^{-1/(k-1)})$  regret, which matches the dominating factor in our algorithm up to a small constant in the denominator. By using hypervolume scalarizations, we conclude by Theorem 12 that this also implies a  $\Omega(T^{-1/(k-1)})$  lower bound on the scalarized Bayes regret.

**Theorem 13.** *There is a setting of objectives  $\Theta^*$  and  $\mathcal{A} = \{a : \|a\| = 1\}$  such that for any actions  $\mathbf{A}_T$ , the hypervolume regret at  $z = 0$  after  $T$  rounds is*

$$\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_T) = \Omega(T^{-1/(k-1)})$$

**Corollary 14.** *Let  $s_\lambda(y)$  be the hypervolume scalarizations with  $\lambda \sim \mathcal{D}$  uniform on  $\mathcal{S}_+^{k-1}$ . Then, there is a setting of objectives  $\Theta^*$  and  $\mathcal{A} = \{a : \|a\| = 1\}$  such that for any actions  $\mathbf{A}_T$ , the scalarized Bayes regret after  $T$  rounds is*

$$BR(s_\lambda, \mathbf{A}_T) = \Omega(T^{-1/(k-1)})$$

## 4 EXPERIMENTS

In this section, we empirically justify our theoretical results by running Algorithm 1 with multiple scalarizations and weight distributions in different settings of the multiobjective stochastic linear bandits. Our empirical results highlight the advantage of the hypervolume scalarization in maximizing the diversity and hypervolume of the resulting Pareto front, especially when there are a mild number of output objectives  $k$ . Our experiments are not meant to show that scalarization is the only or even the best way to solve multiobjective linear bandits; rather, it is a toy example used to demonstrate the importance of choosing optimal scalarizations and weight distributions for solving multiobjective optimization in a variety of settings, such as reinforcement learning.

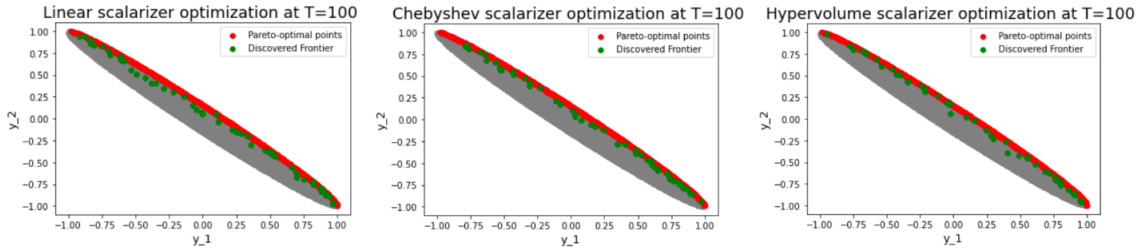


Figure 2: Comparisons of the optimization fronts with anti-correlated  $\theta$ . The green dots represent the frontier points of the 100 points the optimizer has suggested and they are not always Pareto optimal (red). The lack of constant curvature in the convex Pareto front causes the linear scalarization to favor points at the two endpoints of the frontier, while the Chebyshev favors points in the middle of the frontier. The hypervolume scalarizations produce relatively diverse Pareto fronts.

We compare the three widely types of scalarizations that were previously mentioned: the linear, Chebyshev, and the hypervolume scalarization. Note that we use slightly altered form of our hypervolume scalarization as  $s_\lambda(y) = \min_i y_i / \lambda_i$ , which is a simply a monotone transform of the



proposed scalarization and does not inherently affect the optimization. We set our reference point to be  $\mathbf{z} = -\mathbf{2}$  in  $k$  dimension space, since our action set of  $\mathcal{A} = \{a : \|a\| = 1\}$  and our norm bound on  $\Theta^*$  ensures that our rewards are in  $[-1, 1]$ .

In conjunction with the scalarizer, we use our weight distribution  $\mathcal{D} = \mathcal{S}_+$ , which samples vectors uniformly across the unit sphere. In addition, we also compare this with the bounding box distribution methods that were suggested by (Paria et al., 2018), which samples from the uniform distribution from the min to the max each objective and requires some prior knowledge of the range of each objective (Hakanen and Knowles, 2017). Given our reward bounds, we use the bounding box of  $[-1, 1]$  for each of the  $k$  objectives. Following their prescription for weight sampling, we draw our weights for the linear and hypervolume scalarization uniformly in  $[1, 3]$  and take an inverse for the Chebyshev scalarization. We name this the boxed distribution for each scalarization, respectively.

To highlight the differences between the multiple scalarizations, we configure our linear bandits parameters to be anti-correlated, which creates a convex Pareto front with non-uniform curvature. Note that a perfect anti-correlated Pareto front would be linear, which would cause linear scalarizations to always optimize at the end points. We start with simple  $k = 2$  case and let  $\theta_0$  be random and  $\theta_1 = -\theta_0 + \eta$ , where  $\eta$  is some small random Gaussian perturbation (we set the standard deviation to be about 0.1 times the norm of  $\theta_i$ ). We renormalize after the anti-correlation to ensure  $\|\Theta^*\| = 1$ . We run our algorithm with inherent dimension  $d = 4$  for  $T = 100, 200$  rounds with  $k = 2, 6, 10$ . Since our run is in  $k = 2$  output dimensions, we can fully visualize the results and plot the Pareto optimal points found for various scalarizations (see Figure 2).

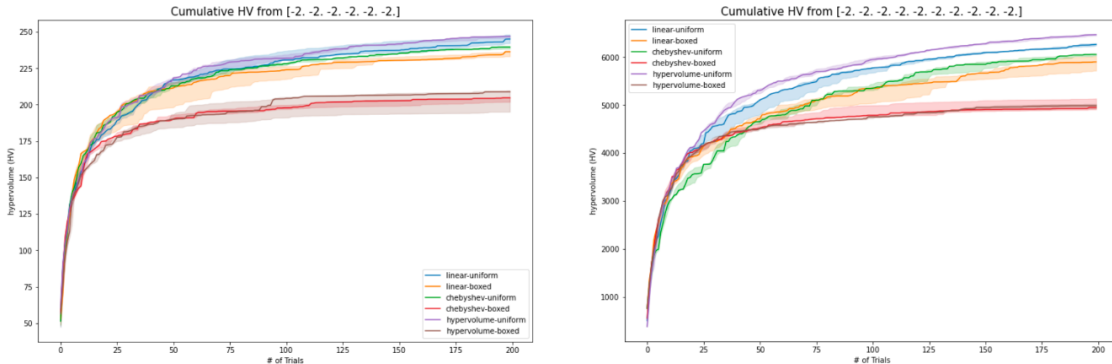


Figure 3: Comparisons of the cumulative hypervolume plots with some anti-correlated  $\theta$ . When the output dimension increase, there is a clearer advantage to using the hypervolume scalarization over the linear and Chebyshev scalarization. We find that the boxed weight distribution does consistently worse than the uniform distribution.

As expected, we find the hypervolume scalarization consistently outperforms the Chebyshev and linear scalarizations, with linear scalarization as the worst performing (see Figure 3). Note that when we increase the output dimension of the problem by setting  $k = 10$ , the hypervolume scalarization shows a more distinct advantage. The boxed distribution approach of (Paria et al., 2018) does not seem to fare well and consistently performs worse than its uniform counterpart. While linear scalarization provides relatively good performance when the number of objective  $k \leq 5$ , it appears that as the number of objectives increase in multi-objective optimization, more care needs to be put into the design of scalarization and their weights due to the curse of dimensionality, since the regions of non-uniformity will exponentially increase. We suggest that as more and more objectives are being added to modern machine learning systems, using smart scalarizations is critical to a uniform exploration of the Pareto frontier.

## REFERENCES

- Michael H Kutner, Christopher J Nachtsheim, John Neter, William Li, et al. *Applied linear statistical models*, volume 5. McGraw-Hill Irwin Boston, 2005.
- Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge, 1998.
- Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.
- Biswajit Paria, Kirthevasan Kandasamy, and Barnabás Póczos. A flexible framework for multi-objective bayesian optimization using random scalarizations. *arXiv preprint arXiv:1805.12168*, 2018.
- Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.
- Joshua Knowles. Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1): 50–66, 2006.
- Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, 11(6):712–731, 2007.
- Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. *Advances in Neural Information Processing Systems*, 32, 2019.
- Abbas Abdolmaleki, Sandy H Huang, Giulia Vezzani, Bobak Shahriari, Jost Tobias Springenberg, Shruti Mishra, Dhruva TB, Arunkumar Byravan, Konstantinos Bousmalis, Andras Gyorgy, et al. On multi-objective policy optimization as a tool for reinforcement learning. *arXiv preprint arXiv:2106.08199*, 2021.
- Zwei Chen, Fengwei Zhou, George Trimonias, and Zhenguo Li. Multi-objective neural architecture search via non-stationary policy gradient. *arXiv preprint arXiv:2001.08437*, 2020.
- Hirofumi Nakayama, Yeboon Yun, and Min Yoon. *Sequential approximate multiobjective optimization using computational intelligence*. Springer Science & Business Media, 2009.
- Stephen Poythress Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, 2004. ISBN 0-521-83378-7.
- Michael TM Emmerich and André H Deutz. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural computing*, 17(3):585–609, 2018.
- Saba Q Yahyaa, Madalina M Drugan, and Bernard Manderick. The scalarized multi-objective multi-armed bandit problem: An empirical study of its exploration vs. exploitation tradeoff. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 2290–2297. IEEE, 2014.
- Angelo Aliano Filho, Antonio Carlos Moretti, Margarida Vaz Pato, and Washington Alves de Oliveira. An exact scalarization method with multiple reference points for bi-objective integer linear optimization problems. *Annals of Operations Research*, pages 1–35, 2019.
- Marie Schmidt, Anita Schöbel, and Lisa Thom. Min-ordering and max-ordering scalarization methods for multi-objective robust optimization. *European Journal of Operational Research*, 275(2):446–459, 2019.
- Rafail Kasimbeyli, Zehra Kamisli Ozturk, Nergiz Kasimbeyli, Gulcin Dinc Yalcin, and Banu Icmen Erdem. Comparison of some scalarization methods in multiobjective optimization. *Bulletin of the Malaysian Mathematical Sciences Society*, 42(5):1875–1905, 2019.

- Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. In *Advances in Neural Information Processing Systems 32*, pages 12037–12047. Curran Associates, Inc., 2019. URL <http://papers.nips.cc/paper/9374-pareto-multi-task-learning.pdf>.
- Róbert Busa-Fekete, Balázs Szörényi, Paul Weng, and Shie Mannor. Multi-objective bandits: Optimizing the generalized gini index. In *International Conference on Machine Learning*, pages 625–634. PMLR, 2017.
- Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.
- Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. 2008.
- Shiyin Lu, Guanghui Wang, Yao Hu, and Lijun Zhang. Multi-objective generalized linear bandits. *arXiv preprint arXiv:1905.12879*, 2019.
- Peter Auer, Chao-Kai Chiang, Ronald Ortner, and Madalina Drugan. Pareto front identification from stochastic bandit feedback. In *Artificial intelligence and statistics*, pages 939–947. PMLR, 2016.
- Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271, 1999.
- Amar Shah and Zoubin Ghahramani. Pareto frontier learning with expensive correlated objectives. In *International conference on machine learning*, pages 1919–1927. PMLR, 2016.
- Marcela Zuluaga, Guillaume Sergent, Andreas Krause, and Markus Püschel. Active learning for multi-objective optimization. In *International Conference on Machine Learning*, pages 462–470, 2013.
- Daniel Golovin and Qiuyu Zhang. Random hypervolume scalarizations for provable multi-objective black box optimization. *arXiv preprint arXiv:2006.04655*, 2020.
- Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.
- Jack Kiefer and Jacob Wolfowitz. The equivalence of two extremum problems. *Canadian Journal of Mathematics*, 12:363–366, 1960.
- Marta Soare, Alessandro Lazaric, and Rémi Munos. Best-arm identification in linear bandits. *Advances in Neural Information Processing Systems*, 27, 2014.
- Yassir Jedra and Alexandre Proutiere. Optimal best-arm identification in linear bandits. *Advances in Neural Information Processing Systems*, 33:10007–10017, 2020.
- Jussi Hakanen and Joshua D Knowles. On using decision maker preferences with parego. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 282–297. Springer, 2017.
- Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, 24:2312–2320, 2011.
- Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.
- Lee-Ad Gottlieb, Aryeh Kontorovich, and Robert Krauthgamer. Adaptive metric dimensionality reduction. *Theoretical Computer Science*, 620:105–118, 2016.

## A MISSING PROOFS

*Proof of Lemma 7.* Let  $\lambda = y^* / \|y^*\|$ . Note that  $\lambda > 0$  since  $y^*$  is in the positive orthant and for the sake of contradiction, assume there exists  $z$  such that  $s_\lambda(z) > s_\lambda(y^*)$ . However, note that for any  $i$ ,  $\frac{z_i}{\lambda_i} \geq \min_i \frac{z_i}{\lambda_i} > \min_i \frac{y_i^*}{\lambda_i} = \frac{y_i^*}{\lambda_i}$ , where the last line follows since  $y_i^* / \lambda_i = \|y^*\|$  for all  $i$  by construction. Therefore, we conclude that  $y^* < z$ , contradicting that  $y^*$  is Pareto optimal.  $\square$

The following lemma about the UCB ellipsoid is borrowed from the original analysis of linear bandits.

**Lemma 15** (Abbasi-Yadkori et al. (2011)). *Consider the least squares estimator  $\hat{\theta}_t = (\mathbf{M}_t)^{-1} \mathbf{A}_t^\top \mathbf{y}_t$ , where the covariance matrix of the action matrix is  $\mathbf{M}_t = \mathbf{A}_t^\top \mathbf{A}_t + \lambda \mathbf{I}$ , then with probability  $1 - \delta$ ,*

$$\|\hat{\theta}_t - \theta^*\|_{\mathbf{M}_t} \leq \sqrt{\lambda} \|\theta^*\| + \sqrt{2 \log(\frac{1}{\delta}) + d \log(T/\lambda)}$$

*Proof of Lemma 8.* Let  $\hat{\Theta}_T$  be the least squares estimate of the true parameters after observing  $(\mathbf{A}_T, \mathbf{y}_T)$ . Since the noise  $\xi_t$  in each objective is independent and 1-sub-Gaussian, by Lemma 15, if we let  $\mathbf{M}_T = \mathbf{A}_T^\top \mathbf{A}_T + \lambda \mathbf{I}$ , then with regularization  $\lambda = 1$

$$\|\hat{\Theta}_{Ti} - \Theta_i^*\|_{\mathbf{M}_T} \leq 1 + \sqrt{2 \log(k/\delta) + d \log(T)} := D_T$$

holds with probability at least  $1 - \delta/k$ . Note that this describes the confidence ellipsoid,  $C_{Ti} = \{\theta \in \mathbb{R}^d : \|\hat{\Theta}_{Ti} - \theta_i\|_{\mathbf{M}_T} \leq D_T\}$  for  $\Theta_{Ti}$ .

By the definition of the UCB maximization of  $a_t$ , we see  $a_t, \tilde{\Theta}_t = \arg\max_{a \in \mathcal{A}} \max_{\theta_i \in C_{Ti}} s_\lambda(\Theta_i^\top a)$ . Note that since  $\Theta^* \in \mathbf{C}_T$ , we can bound the instantaneous scalarized regret as:

$$r(s_\lambda, a_t) = \max_{a \in \mathcal{A}} s_\lambda(\Theta^* a) - s_\lambda(\Theta^* a_t) \leq s_\lambda(\tilde{\Theta}_t a_t) - s_\lambda(\Theta^* a_t)$$

By the Lipschitz smoothness condition, we conclude that  $r(s_\lambda, a_t) \leq L_p \|(\tilde{\Theta}_t - \Theta^*) a_t\|_p$ .

To bound the desired  $\ell_p$  norm, first note that by triangle inequality,  $\|\tilde{\Theta}_t - \Theta^*\|_{\mathbf{M}_T} \leq 2D_T$ . Since we apply uniform exploration every other step and  $\sum_i e_i e_i^\top \succeq \frac{1}{2} \mathbf{I}$  for  $e_i \in \mathcal{E}$  with size  $|\mathcal{E}| = d$ , we conclude that  $\mathbf{M}_T \succeq \frac{T}{5d} \mathbf{I}$ . Therefore, we conclude that  $\|\hat{\Theta}_{Ti} - \Theta_i^*\| \leq 5\sqrt{d/T} D_T := E_T$  with probability at least  $1 - \delta/k$ . Since  $\|a_t\| \leq 1$ , we conclude by Cauchy-Schwarz, that  $|(\hat{\Theta}_{Ti} - \Theta_i^*) a_t| \leq E_T$ . Together with our Lipschitz condition, we conclude that

$$r(s_\lambda, a_t) \leq k^{1/p} L_p E_T \leq 10k^{1/p} L_p d \sqrt{(\log(k/\delta) + \log(T))/T}$$

$\square$

*Proof of Theorem 9.* For any set of actions  $\mathbf{A} \subseteq \mathcal{A}$ , we define  $f_{\mathbf{A}}(\lambda) = \max_{a \in \mathbf{A}} s_\lambda(\Theta^* a)$ . We let  $\mathcal{F} = \{f_{\mathbf{A}} : \mathbf{A} \subseteq \mathcal{A}\}$  be our class of functions over all possible action sets and for any Bayes regret bounds, we will first demonstrate uniform convergence by bounding the complexity of  $\mathcal{F}$ . Specifically, by generalization bounds from Rademacher complexities Bartlett and Mendelson (2002), over choices of  $\lambda_i \sim \mathcal{D}$ , we know that with probability  $1 - \delta$ , for all  $\mathbf{A}$ , we have the bound

$$\left| \mathbf{E}_{\lambda \sim \mathcal{D}}[f_{\mathbf{A}}] - \frac{1}{m} \sum_{i=1}^m f_{\mathbf{A}}(\lambda_i) \right| \leq R_m(\mathcal{F}) + \sqrt{\frac{8 \ln(2/\delta)}{m}}$$

where  $R_m(\mathcal{F}) = \mathbf{E}_{\lambda_i \sim \mathcal{D}, \sigma_i} \left[ \sup_{f \in \mathcal{F}} \frac{2}{m} \sum_i \sigma_i f(\lambda_i) \right]$ , where  $\sigma_i$  are i.i.d.  $\pm 1$  Rademacher variables.

To bound  $R_m(\mathcal{F})$ , we appeal to Dudley's integral formulation that allows us to use the metric entropy of  $\mathcal{F}$  to bound

$$R_m(\mathcal{F}) \leq \inf_{\alpha > 0} \left( 4\alpha + 12 \int_{\alpha}^{\infty} \sqrt{\frac{\log(\mathcal{N}(\epsilon, \mathcal{F}, \|\cdot\|_2))}{m}} d\epsilon \right)$$

where  $\mathcal{N}$  denotes the standard covering number for  $\mathcal{F}$  under the  $\ell_2$  function norm metric over  $\lambda \in \mathcal{D}$ .

Since  $\mathcal{D}$  is the uniform distribution over  $\mathcal{S}_+$ , this induces a natural  $\ell_{\infty}$  function norm metric on  $\mathcal{F}$  that is  $\|f\|_{\infty} = \sup_{\lambda \in \mathcal{S}_+} |f(\lambda)|$ . Since  $s_{\lambda}(\Theta^* a)$  is  $L_{\lambda}$  Lipschitz with respect to the Euclidean norm in  $\lambda$ . Note that since the maximal operator preserves Lipschitzness,  $f_{\mathbf{A}}(\lambda)$  is also  $L_{\lambda}$ -Lipschitz with respect to  $\lambda \in \mathbb{R}^k$ . Since  $\mathcal{F}$  contains  $L_{\lambda}$ -Lipschitz functions in  $\mathbb{R}^k$ , we can bound the metric entropy via a covering of  $\lambda$  via a Lipschitz covering argument (see Lemma 4.2 of Gottlieb et al. (2016)), so we have

$$\log(\mathcal{N}(\epsilon, \mathcal{F}, \|\cdot\|_2)) \leq \log(\mathcal{N}(\epsilon, \mathcal{F}, \|\cdot\|_{\infty})) \leq (4L_{\lambda}/\epsilon)^k \log(8/k)$$

Finally, we follow the same Dudley integral computation of Theorem 4.3 of Gottlieb et al. (2016) to get that

$$R_m(\mathcal{F}) \leq \inf_{\alpha > 0} \left( 4\alpha + 12 \int_{\alpha}^{\infty} \sqrt{\frac{(4L_{\lambda}/\epsilon)^k \log(8/k)}{m}} d\epsilon \right) = O(L_{\lambda}/m^{1/(k+1)})$$

Therefore, we conclude that with probability at least  $1 - \delta$  over the independent choices of  $\lambda_i \sim \mathcal{D}$ , for all  $\mathbf{A}$ ,

$$\left| \mathbf{E}_{\lambda \sim \mathcal{D}} \left[ \max_{a \in \mathbf{A}} s_{\lambda}(\Theta^* a) \right] - \frac{1}{m} \sum_{i=1}^m \max_{a \in \mathbf{A}} s_{\lambda_i}(\Theta^* a) \right| \leq O\left(\frac{BL_{\lambda}}{m^{1/(k+1)}}\right) + \sqrt{\frac{8 \ln(2/\delta)}{m}}$$

Finally, note that for  $T$  even, with constant probability,

$$\begin{aligned} BR(s_{\lambda}, \mathbf{A}_t) &= \mathbf{E}_{\lambda \sim \mathcal{D}} [r(s_{\lambda}, \mathbf{A}_t)] \\ &= \mathbf{E}_{\lambda \sim \mathcal{D}} [\max_{a \in \mathbf{A}} s_{\lambda}(\Theta^* a) - \max_{a \in \mathbf{A}_T} s_{\lambda}(\Theta^* a)] \\ &\leq \frac{1}{T/2} \sum_{i=1}^{T/2} \left[ \max_{a \in \mathbf{A}} s_{\lambda_i}(\Theta^* a) - \max_{a \in \mathbf{A}_T} s_{\lambda_i}(\Theta^* a) \right] + O\left(\frac{BL_{\lambda}}{T^{1/(k+1)}}\right) \\ &\leq \frac{1}{T/2} \sum_{i=1}^{T/2} \left[ \max_{a \in \mathbf{A}} s_{\lambda_i}(\Theta^* a) - s_{\lambda_i}(\Theta^* a_{2i}) \right] + O\left(\frac{L_{\lambda}}{T^{1/(k+1)}}\right) \\ &\leq \frac{1}{T/2} \sum_{i=1}^{T/2} r(s_{\lambda_i}, a_{2i}) + O\left(\frac{BL_{\lambda}}{T^{1/(k+1)}}\right) \\ &\leq O\left(k^{1/p} L_p d \sqrt{\frac{\log(kT)}{T}} + \frac{BL_{\lambda}}{T^{1/(k+1)}}\right) \end{aligned}$$

where the last line used Lemma 8 with  $\delta = 1/T^2$  and applied a union bound over all  $O(T)$  iterations.  $\square$

**Lemma 16.** Let  $s_{\lambda}(y) = \lambda^{\top} y$  be the linear scalarization with  $\|\lambda\| \leq 1$  and  $\|y\|_{\infty} \leq 1$ . Then, we may bound  $L_p \leq \max(1, k^{1/2-1/p})$  and  $L_{\lambda} \leq \sqrt{k}$  and  $|s_{\lambda}| \leq \sqrt{k}$ .

**Lemma 17.** Let  $s_{\lambda}(y) = \min_i \lambda_i y_i$  be the Chebyshev scalarization with  $\|\lambda\| \leq 1$  and  $\|y\|_{\infty} \leq 1$ . Then, we may bound  $L_p \leq 1$  and  $L_{\lambda} \leq 1$  and  $|s_{\lambda}| \leq 1/\sqrt{k}$ .

*Proof of Lemma 16.* Since  $\nabla_{\lambda} s_{\lambda}(x) = y$ , we use Proposition 10 to bound  $L_{\lambda} \leq \max_y \|y\| \leq \sqrt{k} \|y\|_{\infty} = \sqrt{k}$ . Similarly, since  $\nabla_y s_{\lambda}(y) = \lambda$ , we may bound for  $p \leq 2$ ,  $L_p \leq \|\lambda\|_q \leq \|\lambda\| \leq 1$

for  $1/p + 1/q = 1$  and for  $p \geq 2$ , we may use Holder's inequality to bound  $L_p \leq \|\lambda\|_q \leq k^{1/q-1/2} \|\lambda\| \leq k^{1/2-1/p}$ .

To bound the absolute value of  $s_\lambda$ , note  $s_\lambda(y) = \lambda^\top y \leq \sqrt{k}$  for all since  $\|y\|_2 \leq \sqrt{k} \|y\|_\infty \leq \sqrt{k}$ .  $\square$

*Proof of Lemma 17.* For a specific  $\lambda, y$ , let  $i^*$  be the optimal index of the minimization. Then, the gradient  $\nabla_\lambda s_\lambda(x)$  is simply zero in every coordinate except at  $i^*$ , where it is  $y_{i^*}$ . Therefore, since we can only have a finite number of discontinuities due to monotonicity, we use Proposition 10 to bound  $L_\lambda \leq y_{i^*} \leq 1$ . Similarly, since  $\nabla_y s_\lambda(y)$  has only one non-zero coordinate except at  $i^*$ , which is  $\lambda_{i^*}$ , we may bound for  $L_q \leq \lambda_{i^*} \leq 1$ .

To bound the absolute value of  $s_\lambda$ , note that there must exist  $\lambda_i < 1/\sqrt{k}$  as  $\|\lambda\| \leq 1$ . Thus,  $\min_i \lambda_i y_i < 1/\sqrt{k}$  for  $\|y\|_\infty \leq 1$ .  $\square$

*Proof of Lemma 11.* For a specific  $\lambda, y$ , let  $i^*$  be the optimal index of the minimization. Then, the gradient  $\nabla_\lambda s_\lambda(x)$  is simply zero in every coordinate except at  $i^*$ , which in absolute value is  $k(y_{i^*}/\lambda_{i^*})^k(1/\lambda_{i^*})$ .

Let  $j$  be the index such that  $\lambda_j$  is maximized and since  $\|\lambda\| = 1$ , we know that  $\lambda_j \geq 1/\sqrt{k}$ . Therefore, we see that since  $y_{i^*}/\lambda_{i^*} \leq y_j/\lambda_j \leq y_j/\sqrt{k}$ , we conclude that  $1/\lambda_{i^*} \leq (B_u/B_l)/\sqrt{k}$ .

Therefore, using Proposition 10, we have

$$L_\lambda \leq k(y_{i^*}/\lambda_{i^*})^k(1/\lambda_{i^*}) \leq k(B_u/\sqrt{k})^k \frac{(B_u/B_l)}{\sqrt{k}} = \frac{B_u^{k+1} k^{(k+1)/2}}{B_l k^{(k-1)/2}}$$

And similarly, since  $\nabla_y s_\lambda(y)$  has only one non-zero coordinate except at  $i^*$ , which is  $k(y_{i^*}/\lambda_{i^*})^{k-1}(1/\lambda_{i^*})$ , we may bound for

$$L_q \leq k(y_{i^*}/\lambda_{i^*})^{k-1}(1/\lambda_{i^*}) \leq k(B_u/\sqrt{k})^{k-1} \frac{(B_u/B_l)}{\sqrt{k}} \leq \frac{B_u^k}{B_l k^{k/2-1}}$$

To bound the absolute value of  $s_\lambda$ , note that  $s_\lambda(y) \leq (\frac{y_j}{\lambda_j})^k \leq \frac{B_u^k}{k^{k/2}}$ .  $\square$

*Proof of Theorem 12.* Note that by Lemma 6, we connect the Bayes regret to the hypervolume regret for  $\mathcal{D}$ :

$$\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_t) = c_k \mathbf{E}_{\lambda \sim \mathcal{D}} [\max_{a \in \mathcal{A}} s_\lambda(\Theta^* a) - \max_{a \in \mathbf{A}_T} s_\lambda(\Theta^* a)]$$

where  $s_\lambda(y) = \min_i (y_i - z_i/\lambda_i)^k$ .

Note that since  $\|\Theta^* a\|_\infty \leq 1$  for all  $a \in \mathcal{A}$  and  $B$  is maximal, we have  $B \leq \Theta^* a - z \leq B + 2$ . Therefore, we conclude by Lemma 11 that  $s_\lambda$  is Lipschitz with

$$L_p \leq \frac{(B+2)^k}{B k^{k/2-1}}, L_\lambda \leq \frac{(B+2)^{k+1}}{B k^{(k-1)/2}}, |s_\lambda| \leq \frac{(B+2)^k}{k^{k/2}}$$

Finally, we combine this with Theorem 9 with  $p = \infty$  as the optimal choice of  $p$  (since  $L_p$  does not depend on  $p$ ) to get our desired bound on hypervolume regret.  $\square$

*Proof of Theorem 13.* We let  $\mathcal{A} = \{a : \|a\| \leq 1\}$  be the unit sphere and  $\Theta_i^* = e_i$  be the unit vector directions. Note that in this case the Pareto frontier is exactly  $\mathcal{S}_+^{k-1}$ .

Consider a uniform discretization of the Pareto front by taking an  $\epsilon$  grid with respect to each angular component with respect to the polar coordinates. Let  $p_1, \dots, p_m$  be the center (in terms of each of the  $k-1$  angular dimensions) in the  $m = \Theta((1/\epsilon)^{k-1})$  grid elements. We consider the output

$\mathbf{y}_T = \Theta^* \mathbf{A}_T$  and assume that for some grid element  $i$ , it contains none of the  $T$  outputs  $\mathbf{y}_T$ . Since our radial component  $r = 1$ , by construction of our grid in the angular component, we deduce that  $\min_t \|y_t - p_i\|_\infty > \epsilon/10$  by translating polar to axis-aligned coordinates.

Let  $\epsilon' = \epsilon/10$ . Assume also that  $\frac{1}{k} < p_i < 1 - \frac{1}{k}$ . Next, we claim that the hypercube from  $p_i - \epsilon'/k^2$  to  $p_i$  is not dominated by any points in  $\mathbf{Y}_T$ . Assume otherwise that there exists  $y_t$  such that  $y_t \geq p_i - \epsilon'/k^2$ . Now, this combined with the fact that since  $\min_t \|y_t - p_i\|_\infty > \epsilon'$  implies that there must exist a coordinate such that  $y_{tj} \geq p_j + \epsilon'$ .

However, this implies that

$$\sum_{i=1}^k y_{ti}^2 \geq \sum_{i \neq j} (p_i - \epsilon'/k^2)^2 + (p_j + \epsilon')^2 \geq \sum_i p_i^2 - 2(\epsilon'/k^2) \sum_{i \neq j} p_i + 2\epsilon' p_j > 1$$

where the last inequality follows since  $\sum_i p_i < 1/\sqrt{k}$  and  $p_j > \frac{1}{k}$  by assumption. However, this contradicts that  $\|y_t\| \leq 1$ , so it follows that  $p_i - \epsilon'$  is not dominated.

Therefore, for any grid element such that  $p_i > 1/k$ , if there is no  $y_t$  in the grid, we must have a hypervolume regret of at least  $\Omega(\epsilon'^k) = \Omega(\epsilon^k)$  be simply consider the undominated hypervolume from  $p_i$  to  $p_i - \epsilon'$ , which lies entirely within the grid element. In fact, since there are  $\Theta((1/\epsilon)^{k-1})$  such grid elements satisfying  $p_i > 1/k$ , we see that if  $T < O((1/\epsilon)^{k-1})$ , by pigeonhole, there must be a hypervolume regret of at least  $\Omega((1/\epsilon)^{k-1} \epsilon^k) = \Omega(\epsilon)$ .

Therefore, for any  $1/2 > \epsilon > 0$ ,  $\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_T) < \epsilon$  implies that  $T = \Omega((1/\epsilon)^{k-1})$ . Rearranging shows that

$$\mathcal{HV}_z(\Theta^* \mathcal{A}) - \mathcal{HV}_z(\Theta^* \mathbf{A}_T) = \Omega(T^{-1/(k-1)})$$

□

## B CODE

```
# -*- coding: utf-8 -*-
"""Multiobjective Linear Bandits

Automatically generated by Colaboratory.

Original file is located at
    https://colab.corp.google.com/drive/1CD7ek1DV4f3FNzoO7H1rmb0kOkMvx80Z
"""

from absl import flags
from google3.file.recordio.python import recordio
import google3.pyglib.gfile as gfile
import itertools
import dataclasses
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
import math
import pandas as pd
import re
import os

from google3.learning.vizier import benchmark_v2
from google3.learning.vizier.benchmark_v2 import config_pb2
from google3.learning.vizier.benchmark_v2 import analysis
```

```

#@title Simple LinUCB implementation { display-mode: "form" }
import numpy as np

class LinUCB:
    """Simple LinUCB implementation."""

    def __init__(self, dimension: int,
                  max_inst_regret: float = 2.,
                  regularizer: float = 1.,
                  var_noise: float = 1.,
                  max_parameter_norm: float = 1.,
                  failure_probabilty: float = 0.1):
        assert regularizer >= 0.

        self.dimension = dimension
        self._regularizer = regularizer
        self._var_noise = var_noise
        self._max_parameter_norm = max_parameter_norm
        self._max_inst_regret = max_inst_regret
        self._failure_probability = failure_probabilty

        self.reset()

    def reset(self):
        self._covariance_inv = np.eye(self.dimension) * self._regularizer
        self._reward_scaled_features = np.zeros(self.dimension)
        self._parameter_estimate = np.zeros(self.dimension)
        self._num_observations = 0
        self._last_conf_radius = None
        self._conf_ellipsoid_width_raw = self._conf_ellipsoid_rhs()

    def add_observation(self,
                       action,
                       reward: float,
                       context=None):
        """add an observation (action, reward) to the learner.

        This function updates the covariance matrix and parameter estimate.

        Args:
            action: action that was taken
            reward: achieved reward
            context: context for this observation
            blamed: whether this algorithm is to be blamed for this observation.
        """

        self._num_observations += 1

        # Sherman Morrison update of covariance matrix
        y = np.dot(self._covariance_inv, action)
        self._covariance_inv -= np.outer(y, y) / (1 + np.inner(action, y))
        # regression target
        self._reward_scaled_features += reward * action
        # parameter estimate
        self._parameter_estimate = np.dot(self._covariance_inv,
                                           self._reward_scaled_features)
        self._conf_ellipsoid_width_raw = self._conf_ellipsoid_rhs()

```



```

def ucb(self, actions, confidence_scale=1.0):
    """computes the upper-confidence bound for a batch of actions.

    Args:
        actions: A x d matrix
        confidence_scale: scaling factor in front of the bonus prescribed by theory

    Returns:
        upper-confidence bound for each A action, A vector
    """
    rewards = np.dot(actions, self._parameter_estimate)
    var = np.sum(actions * np.dot(actions, self._covariance_inv), axis=1)
    beta = self._conf_ellipsoid_width_raw * confidence_scale
    return rewards, beta * np.sqrt(var)

def choose_action(self, actions, confidence_scale=1.):
    """chooses the action that maximizes the upper confidence bound.

    Args:
        actions: A x d matrix, possible actions to choose from
        confidence_scale: scaling factor in front of the bonus prescribed by theory

    Returns:
        chosen action (d vector)
    """

    rewards, conf_b = self.ucb(actions, confidence_scale)
    action_index = randargmax(rewards + conf_b)
    return actions[action_index, :]

def _conf_ellipsoid_rhs(self):
    # from Abbassi-Yadkori et al. 2011
    beta_t = np.sqrt(self._regularizer) * self._max_parameter_norm
    log_dets = -np.log(self._failure_probability)
    log_dets -= self.dimension / 2 * np.log(self._regularizer)
    log_dets -= np.linalg.slogdet(self._covariance_inv)[1] / 2
    beta_t += np.sqrt(2 * self._var_noise * log_dets)

    return beta_t

def randargmax(b):
    """takes the argmax but randomly picks from the set of maximizers."""
    return np.random.choice(np.flatnonzero(b == b.max()))

def linear_scalarizer(acquisitions, weights):
    sum = 0.0
    for acquisition, weight in zip(acquisitions, weights):
        sum += weight * acquisition
    return sum

def chebyshev_scalarizer(acquisitions, weights):
    min = np.inf
    for acquisition, weight in zip(acquisitions, weights):
        min = np.minimum((acquisition - reference) * weight, min)
    return min

def hypervolume_scalarizer(acquisitions, weights):
    min = np.inf
    for acquisition, weight in zip(acquisitions, weights):

```

```

    min = np.minimum((acquisition - reference) / weight, min)
    return min

def uniform_weights():
    weights = np.random.normal(size=num_metrics)
    return abs(weights) / np.linalg.norm(weights)

def boxed_linear_weights():
    # Use bounding box.
    u = np.random.uniform(low=1.0, high=3.0, size=num_metrics)
    return u / np.linalg.norm(u, ord=1, keepdims=True)

def boxed_chebyshev_weights():
    lmda = boxed_linear_weights()
    lmda_prime = 1.0 / lmda
    return lmda_prime / np.linalg.norm(lmda_prime, ord=1, keepdims=True)

dim = 5
num_metrics = 16
thetas = np.random.normal(size=(num_metrics, dim))

# Anti-correlate thetas.
thetas[0, :] = - thetas[1, :] + np.random.normal(size=thetas[1, :].shape, scale=0.01)
for i in range(int(num_metrics / 2)):
    index = int(2 * i)
    thetas[index, :] = - thetas[index + 1, :] + np.random.normal(size=thetas[index + 1, :].shape, scale=0.01)
# Renormalize to make sure |theta| = 1
thetas = thetas / np.linalg.norm(thetas, axis=1, keepdims=True)
reference = -2

def run_linear_bandit(thetas, scalarizer, weight_generator, num_rounds=100):
    algs = [LinUCB(dimension=dim) for theta in thetas]

    expected_rewards = np.empty(shape=(num_rounds, num_metrics))
    actions = np.random.normal(size=(1000, dim))
    actions = actions / np.linalg.norm(actions, axis=1, keepdims=True)
    for t in range(num_rounds):
        index = np.random.choice(len(actions))
        action = actions[index]
        expected_reward = np.inner(thetas, action)
        rewards = expected_reward + np.random.normal(size=expected_reward.shape)
        for alg, reward in zip(algs, rewards):
            alg.add_observation(action, reward)

        weights = weight_generator()
        acquisitions = scalarizer([sum(alg.ucb(actions)) for alg in algs], weights)
        assert len(acquisitions) == len(actions)
        index = randargmax(acquisitions)

        action = actions[index]
        expected_reward = np.inner(thetas, action)
        rewards = expected_reward + np.random.normal(size=expected_reward.shape)
        for alg, reward in zip(algs, rewards):
            alg.add_observation(action, reward)

        expected_rewards[t] = expected_reward
    return actions, expected_rewards

```

```

scalarizations = {'linear-uniform': (linear_scalarizer, uniform_weights),
                  'linear-boxed': (linear_scalarizer, boxed_linear_weights),
                  'chebyshev-uniform': (chebyshev_scalarizer, uniform_weights),
                  'chebyshev-boxed': (chebyshev_scalarizer, boxed_chebyshev_weights),
                  'hypervolume-uniform': (hypervolume_scalarizer, uniform_weights),
                  'hypervolume-boxed': (hypervolume_scalarizer, boxed_linear_weights)}

from vizier.pyvizier import multimetric
from vizier.pyvizier.m multimetric import xla_pareto

pareto_algo = xla_pareto.JaxParetoOptimalAlgorithm()

import matplotlib.pyplot as plt

num_rounds=100
actions, linear_rewards = run_linear_bandit(thetas, scalarizer=hypervolume_scalarizer)
all_values = np.inner(actions, thetas)
all_frontier = all_values[pareto_algo.is_pareto_optimal(all_values)]
frontier = linear_rewards[pareto_algo.is_pareto_optimal(linear_rewards)]
plt.scatter(all_values[:,0], all_values[:,1],color='grey')
plt.scatter(all_frontier[:,0], all_frontier[:,1], color='red',label='Pareto-optimal p
plt.scatter(frontier[:,0], frontier[:,1],color='green',label='Discovered Frontier',s
plt.xlabel('y_1',fontsize=13)
plt.ylabel('y_2',fontsize=13)
plt.title(f'Hypervolume scalarizer optimization at T={num_rounds} ',fontsize=18)
plt.legend()

origin = -2 * np.ones(shape=num_metrics)
num_repeats = 5
fig, ax = plt.subplots(1, 1, figsize=(12, 8))

for key, (scalarizer, weight_generator) in scalarizations.items():
    hvs = []
    for _ in range(num_repeats):
        _, linear_rewards = run_linear_bandit(
            thetas, scalarizer, weight_generator, num_rounds=200)
        front = multimetric.ParetoFrontier(
            points=linear_rewards,
            origin=origin,
            cum_hypervolume_base=xla_pareto.jax_cum_hypervolume_origin)
        hvs.append(front.hypervolume(is_cumulative=True))
    y = np.array(hvs)
    analysis.plot_median_convergence(
        ax, y, xs=np.array(range(y.shape[1])), label=key, percentiles=((30, 70),))
ax.legend()
ax.set_title(f'Cumulative HV from {origin}', fontsize=15)
ax.set_ylabel('hypervolume (HV)')
ax.set_xlabel('# of Trials')

dim = 5
num_metrics = 16
thetas = np.random.normal(size = (num_metrics,dim))

# Anti-correlate thetas.
thetas[0,:] = - thetas[1,:] + np.random.normal(size=thetas[1,:].shape, scale = 0.01)
for i in range(int(num_metrics/2)):
    index = int(2*i)
    thetas[index,:] = - thetas[index+1,:] + np.random.normal(size=thetas[index+1,:].shape, scale = 0.01)
# Renormalize to make sure |theta| = 1

```

```
thetas = thetas/np.linalg.norm(thetas,axis=1,keepdims=True)
reference = -2
```