OPT2025: 17th Annual Workshop on Optimization for Machine Learning

Algorithm design and sharper bounds for improving bandits

Avrim Blum AVRIM@TTIC.EDU
Toyota Technological Institute at Chicago, 6045 S. Kenwood Ave., Chicago, IL 60637

Marten Garicano MARTENGARICANO@UCHICAGO.EDU
University of Chicago, 5801 S. Ellis Ave., Chicago, IL 60637

Kavya Ravichandran KAVYA@TTIC.EDU
Toyota Technological Institute at Chicago, 6045 S. Kenwood Ave., Chicago, IL 60637

Dravyansh Sharma DRAVY@TTIC.EDU
IDEAL Institute, Toyota Technological Institute at Chicago 6045 S. Kenwood Ave., Chicago, IL 60637

Abstract

The improving multi-armed bandits problem is a formal model for allocating effort under uncertainty, motivated by scenarios such as investing research effort into new technologies, performing clinical trials, and hyperparameter selection from learning curves. Each pull of an arm provides reward that increases monotonically with diminishing returns. A growing line of work has designed algorithms for this problem, albeit with somewhat pessimistic worst-case guarantees. Indeed, strong lower bounds of $\Omega(k)$ and $\Omega(\sqrt{k})$ multiplicative approximation factors are known for both deterministic and randomized algorithms (respectively) relative to the optimal arm, where k is the number of bandit arms. In this work, we propose a new parameterized family of bandit algorithms and bound the sample complexity of learning the near-optimal algorithm from that family using offline data. This family includes the optimal randomized algorithm from prior work. We show that an appropriately chosen algorithm from this family can achieve stronger guarantees, with optimal dependence on k, when the arm reward curves satisfy additional properties related to the strength of concavity. We further bound the sample complexity of learning a near-optimal algorithm from the family using offline data. Taking a statistical learning perspective on the bandit rewards optimization problem, we achieve stronger data-dependent guarantees without the need for actually verifying whether the assumptions are satisfied.

1. Introduction

The multi-armed bandits problem has numerous practical applications, and a large body of literature is devoted to studying its various aspects. A natural use case is for modeling situations in which investment in an option increases the payoff of that option. One example is the development of a new technology, where investing in research increases its efficacy, albeit with diminishing returns. Similarly, consider the example of tuning hyperparameters for neural networks. Here each bandit arm corresponds to a value of the hyperparameter, an arm pull corresponds to running an additional training epoch for the corresponding value of the hyperparameter, and the reward function corresponds to the learning curves that capture the training accuracy as a function of the epochs (for the different hyperparameter values). The *improving multi-armed bandits* framework, originally formalized by [8, 13], captures this setting.

Recent work by [4] has developed nearly-tight approximation guarantees for this problem. Formally, the problem consists of k bandit arms, each of which has associated with it a reward

function that increases the more the arm is pulled. Work on the problem has traditionally assumed that the reward functions are concave (i.e., satisfy diminishing returns), and lower bounds involve minimally concave (i.e., linear) functions.

However, in many practical settings, we would expect the reward functions to not only be concave but also satisfy some stronger condition on the growth rate. In addition, we often have access to historical data consisting of instances arising from related tasks. We can use these related, previously-seen tasks to design an algorithm for the current instance. Formally, we assume we have access to multiple IMAB instances drawn from an unknown distribution. We seek to perform as well as the best parameter α in the defined algorithm family, on average over the distribution. If instances drawn from the distribution are 'bad', then we wish to recover worst-case performance. However, if instances drawn from the distribution satisfy stronger assumptions, we hope to do better than in the worst case.

In this work, we propose an algorithm family that can be used to study the task at hand. This algorithm family is carefully designed to contain optimal algorithms for general instances. Furthermore, we show that for a stronger parameterized concavity assumption, an optimal algorithm with corresponding parameter α lies in this algorithm family. Our approach thus satisfies (1) robustness to the assumption made (on bad instances, we match worst-case guarantees) and (2) consistency with respect to stronger assumptions (if an instance satisfies a stronger assumption, we improve upon existing guarantees)[1]. We show that finding the best algorithm among this class needs only a finite, polynomial number of samples from the distribution. Our work considers the combinatorial optimization problem of improving multi-armed bandits from a statistical learning perspective, through data-driven algorithm design. In particular, we apply and extend results from [15].

See Appendix A for broader context on related literature.

2. Preliminaries

We formally define the improving multi-armed bandits framework introduced in [8].

Definition 1 *An instance $I \in \mathcal{I}$ of the improving multi-armed bandits problem consists of k arms, each of which has associated with it a reward function f_i that is increasing as a function of t_i , which is the number of times that arm has been pulled so far.*

Following [4, 8, 13], we assume the arms follow *diminishing returns*, i.e., $f(t+1) - f(t) \leq f(t) - f(t-1)$ for each reward function f . Let f^* denote the arm with highest cumulative reward at horizon T (known). We mainly study cumulative reward; a similar argument to that in Appendix A of [4] can be used to extend our results to best arm identification. Let $\text{OPT}_t := \sum_{n=1}^{t} f^*(n)$.

Typically, in bandits problems, we are interested in minimizing regret compared to an optimal policy, i.e., the difference between the reward due to the used policy and the optimal policy. For this problem there are some simple instances that show that getting non-trivial regret is impossible, and therefore that we should instead study the competitive ratio of the reward.

Definition 2 *Suppose an algorithm accrues reward ALG and the optimal reward possible on the instance is OPT . We say that the algorithm achieves competitive ratio c if $OPT/ALG \leq c$.*

1. We borrow the terms from the algorithms with predictions literature, using them metaphorically in our data-driven algorithm design setting.

Prior work provides upper bounds (algorithms) and lower bounds (hard instances) for this problem when optimizing for competitive ratio assuming the reward functions are concave. In particular, [13] consider deterministic algorithms and show tight upper and lower bounds at $\Theta(k)$. In a recent work, [4] show that randomization helps achieve a sharper $O(\sqrt{k}\log k)$ competitive ratio ($O(\sqrt{k})$ when $f^*(T)$ is known to the algorithm), which they complement with an $\Omega(\sqrt{k})$ lower bound.

2.1. Algorithm Family

Definition 3 *Define the family of algorithms PTRR* $:= \{PTRR_\alpha : \alpha \in (0,1]\}$, *where* $PTRR_\alpha$ *(α-Power-Thresholded Round Robin) is Algorithm 1.*

Algorithm 1 $PTRR_\alpha$

Require: maximum pull estimate m, local horizon τ

$t \leftarrow 0, R \leftarrow 0, S \leftarrow \{1, \ldots, k\}$
while $t < T$ **do**
  sample i uniformly from S
  $S \leftarrow S \setminus \{i\}, t_i \leftarrow 0$
  while $t < T$ and $f_i(t_i) \geq m\left(\frac{t_i}{\tau}\right)^\alpha$ **do**
    pull arm i
    $t_i \leftarrow t_i + 1, t \leftarrow t + 1, R \leftarrow R + f_i(t_i)$
return R

Each $PTRR_\alpha$ is a slightly modified version of Algorithm 1 in [4]. While playing arm i, we keep pulling it as long as $f_i(t_i) \geq m\left(\frac{t_i}{T}\right)^\alpha$, where t_i denotes the number of pulls of i so far. When the inequality first fails, we abandon i and move to a uniformly random new arm, repeating until time T.

2.2. Data-driven Algorithm Design Perspective and Problem Statement

We consider loss functions of an algorithm in the above family on an instance, defining $l(I, \alpha)$ as the loss of $PTRR_\alpha$ on the instance I . We also define the dual of a loss function.

Definition 4 *A loss function* $\ell : \mathcal{I} \times (0,1] \to \mathbb{R}$ *is* H-well-behaved *if it is bounded, i.e.,* $\ell : \mathcal{I} \times (0,1] \to [0, H]$ *and piecewise-constant.*

Definition 5 *We fix the instance and consider the loss on a fixed instance as a function of the algorithm. Since the algorithm arises from a parameterized family, the loss is now a function of the parameter. In particular, the* dual of the loss function *is given by:* $l_T^P(\alpha) = \ell(P, \alpha)$.

Finally, we formally define the problem we study.

Definition 6 (Hyperparameter Transfer Setting) *Suppose we have a distribution $\mathcal{D}$ over instances I of the improving multi-armed bandits problem as defined in Definition 1. Consider the family of algorithms parameterized by α defined in Definition 3. Finally, consider an H-well-behaved loss, l. The* Hyperparameter Transfer Setting *is the following problem. Given N instances sampled iid from $\mathcal{D}$, identify $\hat{\alpha}$ such that*

$$\left| \mathbb{E}_{P\sim\mathcal{D}}\left[l_T(P, \hat{\alpha})\right] - \min_{\alpha\in\mathcal{P}} \mathbb{E}_{P\sim\mathcal{D}}\left[l_T(P, \alpha)\right] \right| < \epsilon\,. \tag{1}$$

3. Sharper Competitive Ratio

In this section, we argue that it is natural to consider the simple one–parameter family *PTRR* from Definition 3. First, we observe that our family contains the algorithm from [4] known to be optimal in the unrestricted case. We then show that there are algorithms in *PTRR* that enjoy improved competitive ratio on every instance satisfying a mild growth condition. A matching lower bound shows that these algorithms are optimal on such instances. We start by defining a slightly stronger version of concavity.

Definition 7 (Per-arm β-Lower Envelope, LE(β)) *For any $\beta \in (0, 1]$, we say that arm i satisfies LE(β) if*

$$f_i(t) \geq f_i(T) \left(\frac{t}{T}\right)^{\beta} \quad \text{for all } t \leq T.$$

Definition 8 (Concavity Envelope Exponent, β_I) *For every instance I, we define its Concavity Envelope Exponent β_I as $\beta_I := \inf \{\beta \in (0, 1] : \text{ every arm in } I \text{ satisfies } \mathrm{LE}(\beta)\}$.*

Remark 9 *Note that smaller β_I indicate larger early rewards, making learning easier. Moreover, note that every non-decreasing concave function with $f(0) \geq 0$ satisfies LE(1), which implies that 1 is an upper bound on the CEE. When β_I approaches 1 (an instance contains near-linear arms), the problem reverts to the $\Theta(\sqrt{k})$ regime.*

We will now evaluate the performance of our family. Note that setting $\alpha = 1$ recovers the algorithm from [4], which ensures that *PTRR* can preserve the $O(\sqrt{k})$ guarantee.

Now suppose that an instance has $\beta_I < 1$, which occurs whenever it has no (arbitrarily close to) linear arms. Under this assumption, we will prove that choosing any $\alpha \in (\beta_I, 1)$ yields the strictly smaller competitive ratio $O\big(k^{\alpha/(\alpha+1)}\big) = o(\sqrt{k})$. Letting α approach β_I gives $O\big(k^{\beta_I/(\beta_I+1)}\big)$.

Theorem 10 *Given an IMAB instance I with Concavity Envelope Exponent β_I. If $T \geq 2k$, then $PTRR_\alpha$ for $\alpha \in (\beta_I, 1)$ with $\tau = T - k$, $m = \frac{\tau}{T} f^*(T)$ achieves*

$$\mathbb{E}[\textit{reward from PTRR}_\alpha] \geq \frac{1}{2^{\alpha+3}(\alpha+1)} \cdot \frac{\textit{OPT}_T}{(k+1)^{\frac{\alpha}{\alpha+1}}},$$

Equivalently, the competitive ratio is $O(k^{\alpha/(\alpha+1)})$. Setting $\alpha = 1$ recovers the $O(\sqrt{k})$ bound.

Proof [Proof Sketch] Mirroring [4], two simple facts drive the analysis of our algorithm. First, the optimal arm is never abandoned when $\alpha > \beta_I$. Second, if we ever abandon a non-optimal arm at time t, the cumulative reward collected on that arm is at least the "area" under the lower envelope up to t.

These two facts yield a recurrence that trades off (i) the chance we picked the optimal arm now, versus (ii) the value we get after spending t pulls and moving on. At any state (τ', k'), we either draw the best arm now (probability $1/k'$) and then keep it forever, earning $\mathrm{OPT}_{\tau'+k'}$, or we draw a bad arm. If it is bad and we abandon it at time t, we have already earned at least the threshold-area up to t and we continue from the smaller state $(\tau' - t, k' - 1)$. Induction on (τ', k') yields the desired result. A full proof is included in Appendix B.1 (see Theorem 20). ■

The differences with the proof in [4] stem from the change in the arm abandonment condition, which turns several quadratic expressions into polynomials whose degree is a function of α.

We next provide a matching lower bound by showing that there is a distribution on instances with the same β_I on which no algorithm beats $\Omega(k^{\beta_I/(\beta_I+1)})$. It follows that the exponent $\beta_I/(\beta_I+1)$ is optimal. *PTRR* attains this when α is tuned to β_I.

Theorem 11 (Lower Bound) *Fix $\beta \in (0,1]$, $k \geq 2$, and suppose T is sufficiently large*[2]*. For every (possibly randomized) algorithm, there exists an instance with Concavity Envelope Exponent $\beta_I = \beta$ such that*

$$\frac{\mathbb{E}[ALG_T]}{OPT_T} \leq C_\beta k^{-\beta/(\beta+1)} \quad \textit{with} \quad C_\beta = \frac{3}{2}(\beta+1)^2\left[\beta(\beta+1)\right]^{-\beta/(\beta+1)}.$$

Equivalently, any such algorithm has an $\Omega(k^{\beta/(\beta+1)})$ competitive ratio.

Proof [Proof Sketch] We construct a similar worst-case to [4]. Define a "good" arm as the power curve $g(t) = m(t/T)^\beta$, and let the other $k-1$ arms match g exactly for the first t' pulls and then flatten at $g(t')$. Every arm satisfies LE(β), and the good arm violates any stricter floor, so $\beta_I = \beta$. We study the distribution that chooses which arm is good uniformly at random.

Before t' pulls, the arms are indistinguishable, so any algorithm must spend t' pulls to "clear" an arm. With budget T it clears at most T/t' arms, and if it misses the good arm, every remaining pull is worth at most $g(t')$. We use this to upper-bound the expected reward.

Letting $t' = \lfloor x^* T \rfloor$ with $x^* = [k\,\beta(\beta+1)]^{-1/(\beta+1)}$ yields the desired $k^{-\beta/(\beta+1)}$ scaling. By Yao's principle, this also holds for randomized algorithms. At $\beta = 1$, we recover the $\Omega(\sqrt{k})$ bound from [4]. A full proof is in Appendix B.2 (see Theorem 27). ■

For the lower bound, the main change relative to the proof in [4] is replacing the linear "good" arm and the linear portion of the remaining arms by a power-law curve, which affects the generous reward and leads to a $k^{-\beta/(\beta+1)}$ dependence.

4. Sample Complexity Analysis

In this section, we analyze the number of samples required to learn a near-optional algorithm from the algorithm family *PTRR* (Definition 3). To do this, we appeal to results proven in [15], where the authors develop a theory for the general version of the Hyperparameter Transfer problem (Definition 6). We extend their sample complexity results to our setting.

Derandomized Dual Complexity and result of [15]. We start by explaining how to use results from [15]. In their work, they defined a relevant quantity, the *derandomized dual complexity*, which accounted for randomness in data sampling and any additional randomness after fixing the instance. Since our family of algorithms includes randomized algorithms, we must also derandomize this additional source of randomness. We do this by defining $\mathcal{D}'$ as an extension of the original distribution. For each element in the support of the original distribution, we define $k!$ copies, each with a different permutation π_k of $[k]$ associated with it. Each new augmented instance (I, π_k) has probability $\mathcal{D}'_I := \mathcal{D}_I/k!$ associated with it, where $\mathcal{D}_I$ is the probability associated with that instance originally. Wherever the results of [15] refer to the distribution $\mathcal{D}$ over instances, we instead consider

2. see Theorem 27.

the distribution over instances and permutations as defined above. By derandomizing in this way, we can immediately apply their results[3].

From the results of [15] (reproduced in Appendix C.1), we know that if we have sufficiently many samples, as a function of $Q_{\mathcal{D}}$, H and the accuracy and success probability parameters, then with high probability the empirical loss will be close to the population loss. It is then straightforward to show that if we have enough samples to achieve uniform convergence, we will find a hyperparameter value that achieves near-optimal performance on future instances from the distribution (see Appendix C.2). Thus, if we pick the value of α with the smallest empirical loss, we can guarantee that the objective in Eqn. 1 is satisfied. It remains to (1) bound $Q_{\mathcal{D}}$ for our problem and (2) investigate various reasonable H-well-behaved losses.

Bounding derandomized dual complexity in our setting. Now, let us compute a bound on the value of $Q_{\mathcal{D}}$, i.e., compute the number of possible behaviors of algorithms from $\mathcal{A}$. In order to do so, we identify a sufficient statistic for the loss of a fixed algorithm on an instance then count the number of possible values of the statistic.

Lemma 12 *For the family $\mathcal{A}$ defined in Defn. 3, the improving multi-armed bandits problem, and* any *piecewise-constant loss function, $Q_{\mathcal{D}} \leq kT$.*

Proof Sketch Our proof hinges on the fact that for a fixed value of α , for each of the k times, there is a unique time $t \in \{1, \ldots, T\}$ at which the algorithm discards the arm. The total of T values t could be for each of the k arms multiply to yield the kT upper bound. Full proof is in Appendix C.3.

Sample complexity results. Now we present sample complexity results for generic H-well-behaved loss functions. In Appendix C.4, we instantiate this for a concrete loss function.

Theorem 13 *For the Hyperparameter Transfer setting for the improving multi-armed bandits problem with generic H-well-behaved loss, $N = O\left(\left(\frac{H}{\epsilon}\right)^2 (\log kT + \log \frac{1}{\delta})\right)$ instances drawn from $\mathcal{D}$ suffice to get the uniform convergence guarantee in Theorem 29.*

This is a simple application of Theorem 29 by incorporating the bound on $Q_{\mathcal{D}}$ from Lemma 12. We can see that the sample complexity only has logarithmic dependence on k , the number of arms, implying we could hope to use this procedure even in cases where k is very large.

5. Future Directions

Here, we discuss the limitations of our work and related extensions.

Anytime guarantees. We may also want to solve this problem without knowledge of the time horizon T. Since $Q_{\mathcal{D}}$ depends on T , i.e., the number of pieces in the loss function, removing this assumption might not be as simple as the doubling trick employed in [4].

Knowing $f^\star(T)$. In the data-driven perspective, the boundedness of the loss is a function of $f^\star(T)$, and so we have assumed that we know the maximum value of the best arm. It would be interesting to relax this assumption by learning it from data.

3. For our purposes, it suffices to handle randomness in the algorithm in this way. However, it would be interesting and valuable to extend their results to general randomized algorithms in future work.

Acknowledgments

This work was supported in part by the National Science Foundation under grants CCF-2212968 and ECCS-2216899, by the Simons Foundation under the Simons Collaboration on the Theory of Algorithmic Fairness, and by the Office of Naval Research MURI Grant N000142412742. The views expressed in this work do not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred.

References

[1] Maria-Florina Balcan. Data-Driven Algorithm Design (book chapter). In *Beyond Worst-Case Analysis of Algorithms, Tim Roughgarden (Ed)*. Cambridge University Press, 2020.

[2] Maria-Florina Balcan and Dravyansh Sharma. Learning accurate and interpretable decision trees. In *Uncertainty in Artificial Intelligence*, pages 288–307. PMLR, 2024.

[3] Peter Bartlett, Piotr Indyk, and Tal Wagner. Generalization bounds for data-driven numerical linear algebra. In *Conference on Learning Theory (COLT)*, pages 2013–2040. PMLR, 2022.

[4] Avrim Blum and Kavya Ravichandran. Nearly-tight approximation guarantees for the improving multi-armed bandits problem. In Gautam Kamath and Po-Ling Loh, editors, *Proceedings of The 36th International Conference on Algorithmic Learning Theory*, volume 272 of *Proceedings of Machine Learning Research*, pages 228–245. PMLR, 2 2025. URL https://proceedings.mlr.press/v272/blum25a.html.

[5] Avrim Blum, Chen Dan, and Saeed Seddighin. Learning complexity of simulated annealing. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1540–1548. PMLR, 2021.

[6] Hongyu Cheng and Amitabh Basu. Learning cut generating functions for integer programming. *Advances in Neural Information Processing Systems*, 37:61455–61480, 2024.

[7] Rishi Gupta and Tim Roughgarden. A PAC approach to application-specific algorithm selection. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 123–134, 2016.

[8] Hoda Heidari, Michael Kearns, and Aaron Roth. Tight policy regret bounds for improving and decaying bandits. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 1562–1570. AAAI Press, 2016. ISBN 9781577357704.

[9] Billy Jin, Thomas Kesselheim, Will Ma, and Sahil Singla. Sample complexity of posted pricing for a single item. In *Advances in Neural Information Processing Systems*, 2024.

[10] Mikhail Khodak, Edmond Chow, Maria-Florina Balcan, and Ameet Talwalkar. Learning to relax: Setting solver parameters across a sequence of linear system instances. *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[11] Misha Khodak, Ilya Osadchiy, Keegan Harris, Maria-Florina Balcan, Kfir Y Levy, Ron Meir, and Steven Z Wu. Meta-learning adversarial bandit algorithms. *Advances in Neural Information Processing Systems*, 36:35441–35471, 2023.

[12] Alberto Maria Metelli, Francesco Trovò, Matteo Pirola, and Marcello Restelli. Stochastic rising bandits, 2022. URL http://arxiv.org/abs/2212.03798.

[13] Vishakha Patil, Vineet Nair, Ganesh Ghalme, and Arindam Khan. Mitigating disparity while maximizing reward: Tight anytime guarantee for improving bandits. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 4100–4108. International Joint Conferences on Artificial Intelligence Organization, 2023. ISBN 978-1-956792-03-4. doi: 10.24963/ijcai.2023/456. URL https://www.ijcai.org/proceedings/2023/456.

[14] Shinsaku Sakaue and Taihei Oki. Generalization bound and learning methods for data-driven projections in linear programming. *Advances in Neural Information Processing Systems*, 37: 12825–12846, 2024.

[15] Dravyansh Sharma and Arun Suggala. Offline-to-online hyperparameter transfer for stochastic bandits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 20362–20370, 2025.

Appendix A. Additional Related Work

Our work most concretely extends [4] and [15]. In this section, we survey these works and other work that contextualizes each of these.

Improving / Rising Bandits Initially, [8] formalized the problem of payoffs that increase the more the arm is played and study a notion of regret in hindsight. [13] extend the setting by studying competitive ratio. They show deterministic algorithms cannot achieve better than $O(k)$ upper and lower bounds. Then, [4] extend the development on this problem to randomized algorithms, showing that randomized algorithms can achieve an $O(\sqrt{k}\log k)$ upper bound, nearly matching the $\Omega(\sqrt{k})$ lower bound. A related line of work is that on *rising* bandits [12]. This line of work bounds regret in terms of quantities computed about the instance but otherwise considers similar increasing reward functions.

Data-Driven Algorithm Design. Introduced by [7], data-driven algorithm design is emerging as a powerful approach for using machine learning to design algorithms that have optimal performance on "typical" input instances (as opposed to worst-case or average-case analysis). The approach is effective in both statistical and online learning settings [1], and has been successfully used for designing several fundamental algorithms in machine learning and beyond (e.g. [2, 3, 5, 6, 9, 10, 14]). Recent work [15] has shown how to tune hyperparameters in standard multi-armed bandit algorithms like UCB, LinUCB and GP-UCB under this paradigm. While they focus on stochastic bandits, we extend their techniques to tune hyperparameters in the non-stochastic improving bandits setting. Another related work [11] shows how to tune hyperparameters for adversarial bandit algorithms in an online-with-online meta-learning setting.

Appendix B. Full Proofs for Sharper Competitive Ratio

B.1. Upper Bound

Suppose an instance has Concavity Envelope Exponent $\beta_I \in (0,1)$, and fix $\alpha \in (\beta_I, 1)$. Let $\gamma := \frac{\alpha}{\alpha+1}$. For $\tau' \geq 0$ and $k' \geq 1$, let $V(\tau', k')$ denote the expected reward earned in the next $\tau' + k'$ pulls when k' arms are still untouched. Let $V(\tau'', \cdots) = 0$ for all $\tau'' \leq 0$.

We analyze *PTRR*$_\alpha$ through a recurrence that trades off two properties. First, the optimal arm is never abandoned when $\alpha > \beta_I$ (Lemma 14). Second, abandoning any non-optimal arm after t pulls yields at least the "area under the envelope" up to t (Lemma 15). These two facts give the value recurrence in Lemma 16: with probability $1/k'$ we hit f^* and earn $\mathrm{OPT}_{\tau'+k'}$; otherwise we bank the area from time t and recurse from $(\tau' - t, k' - 1)$ with the minimizer attained on $[0, \tau']$. We then solve the recurrence by normalizing t to $y = t/\tau'$ and minimizing a one-variable convex form $u\,y^{\alpha+1} + v(1-y)^{\alpha+1}$ exactly (Lemma 17). A clean lower bound on this minimum (Lemma 18) closes the induction and yields

$$V(\tau', k') \geq \frac{m}{(\alpha+1)\tau^\alpha} \cdot \frac{(\tau')^{\alpha+1}}{2\,(k'+1)^\gamma}$$

(Lemma 19). Finally, substituting $(\tau', k') = (T - k, k)$ and OPT_T's comparison to m gives Theorem 20. A more detailed overview of the proof is included after the theorem.

Our analysis of $PTRR_\alpha$ holds for any parameter m that satisfies

$$\frac{1}{c_2} f^*(T-k) \le m \le f^*(T) \left(\frac{T-k}{T}\right)^\alpha$$

for some constant $c_2 \ge 1$. In particular, setting $m := \frac{T-k}{T} f^\star(T)$ gives $c_2 = \frac{T}{T-k} \in [1,2]$ when $T \ge 2k$. Since $\frac{T-k}{T} \in (0,1]$ and $\alpha \in (0,1)$, we also know that $f^\star(T)\frac{T-k}{T} \le f^*(T)\left(\frac{T-k}{T}\right)^\alpha$, and therefore that $\frac{1}{2} f^\star(T-k) \le m \le f^*(T)\left(\frac{T-k}{T}\right)^\alpha$ in this case.

Lemma 14 (Optimal arm is never dropped) *$PTRR_\alpha$ will never switch away from the optimal arm f^*.*

Proof By definition of the CEE, we know that $f_\star(t) \ge f_\star(T)\left(\frac{t}{T}\right)^{\beta_I}$ for all $0 \le t \le T$. Moreover, recall that $m \le f^*(T)\left(\frac{\tau}{T}\right)^\alpha$. Since $\frac{\tau}{T} \in (0,1]$ and $\alpha > \beta_I$, it follows that

$$f^\star(t) \ge f^\star(T)\left(\frac{t}{T}\right)^{\beta_I} \ge f^\star(T)\left(\frac{t}{T}\right)^{\alpha} = f^\star(T)\left(\frac{\tau}{T}\right)^\alpha \left(\frac{t}{\tau}\right)^\alpha \ge m\left(\frac{t}{\tau}\right)^\alpha.$$

for all $t \le T$, and therefore that the test never fails once the optimal arm is reached. ■

Lemma 15 (Area before abandonment) *If the keep-test holds for $t_i - 1$ pulls on arm i and fails at t_i, we obtain a reward of at least*

$$\frac{m}{(\alpha+1)\tau^\alpha}(t_i-1)^{\alpha+1}.$$

Proof For $1 \le s < t_i$, we know that $f_i(s) \ge m(s/\tau)^\alpha$. Summing and using the monotonicity of x^α on $[0,\infty)$ gives

$$\sum_{s=1}^{t_i-1} f_i(s) \;\ge\; m\tau^{-\alpha}\sum_{s=1}^{t_i-1} s^\alpha \;\ge\; m\tau^{-\alpha}\int_0^{t_i-1} x^\alpha\,dx \;\ge\; \frac{m}{(\alpha+1)\tau^\alpha}(t_i-1)^{\alpha+1}.$$

■

With k' untouched arms, we either hit the best arm now (probability $1/k'$) and then take the optimum path, or we spend t pulls on a non-best arm, bank the area from Lemma 15, and recurse on $(\tau'-t, k'-1)$. To ensure a worst-case guarantee, we take the minimum over all feasible abandonment times t.

Lemma 16 (Value recurrence) *For all $\tau' \ge 0$, $k' \ge 1$,*

$$V(\tau',k') \;\ge\; \frac{1}{k'}\,\mathrm{OPT}_{\tau'+k'} + \left(1-\frac{1}{k'}\right)\min_{0\le t\le \tau'+k'-1}\left[\frac{m}{(\alpha+1)\tau^\alpha}\,t^{\alpha+1} + V(\tau'-t,k'-1)\right], \quad (2)$$

and the minimum is attained on $[0,\tau']$.

Proof With probability $1/k'$ the next arm is f^* and, by Lemma 14, the algorithm stays on it and earns $\mathrm{OPT}_{\tau'+k'}$. Else it plays a non-optimal arm. If it abandons this arm at time t, Lemma 15 gives the earned area and the process recurses on $(\tau' - t, k' - 1)$. If $t \geq \tau'$, then $V(\tau' - t, k' - 1) = 0$ while the area term increases in t, which implies that the minimizer lies in $[0, \tau']$. ■

Below, we prove two technical results that aid our analysis of the recurrence. First, the recurrence reduces to minimizing $u\,y^p + v(1-y)^p$ over $y \in [0,1]$. This convex function has a unique minimizer balancing the two terms. We write that minimum in closed form to keep constants explicit.

Lemma 17 (One-variable minimum) *For $p > 1$ and $u, v > 0$,*

$$\min_{y\in[0,1]} \{\, u\,y^p + v(1-y)^p \,\} \;=\; \left(u^{-1/(p-1)} + v^{-1/(p-1)}\right)^{-(p-1)}.$$

Proof Let $f(y) := u\,y^p + v(1-y)^p$ on $[0,1]$. For $p > 1$, we have

$$f''(y) = u\,p(p-1)y^{p-2} + v\,p(p-1)(1-y)^{p-2} > 0 \quad (y \in (0,1)),$$

so f is strictly convex. Moreover, note that

$$f'(y) = up\,y^{p-1} - vp\,(1-y)^{p-1}, \quad f'(0^+) = -vp < 0, \;\; f'(1^-) = up > 0,$$

which implies that f' has a unique global minimizer $y^\star \in (0,1)$. Solving $u\,y^{\star p-1} = v\,(1-y^\star)^{p-1}$ gives

$$y^\star = \frac{v^{1/(p-1)}}{u^{1/(p-1)} + v^{1/(p-1)}}.$$

Letting $a := u^{1/(p-1)}$ and $b := v^{1/(p-1)}$, it follows that

$$\min_{y\in[0,1]} f(y) = f(y^\star) = \frac{a^{p-1}b^p + b^{p-1}a^p}{(a+b)^p} = \frac{(ab)^{p-1}}{(a+b)^{p-1}} = \left(u^{-1/(p-1)} + v^{-1/(p-1)}\right)^{-(p-1)}.$$

■

We now lower-bound the exact minimum with a simple balancing inequality that cleanly shows the dependence on k'.

Lemma 18 (Balancing inequality) *Let $\alpha \in (0,1]$ and $\gamma := \frac{\alpha}{\alpha+1}$. For all integers $k' \geq 1$,*

$$\frac{1}{k'} + \left(1 - \frac{1}{k'}\right)\left(1 + \left(2\,k'^{\gamma}\right)^{1/\alpha}\right)^{-\alpha} \;\geq\; \frac{1}{2\,(k'+1)^{\gamma}}. \tag{3}$$

Proof Let $A := (2\,k'^{\gamma})^{1/\alpha} \geq 0$. Note that the function $u \mapsto (1+u)^{-\alpha}$ is convex and decreasing on $[0,\infty)$ for every $\alpha > 0$, which implies that

$$(1+A)^{-\alpha} = A^{-\alpha}\left(1 + \frac{1}{A}\right)^{-\alpha} \;\geq\; A^{-\alpha}\left(1 - \frac{\alpha}{A}\right) = \frac{1}{2\,k'^{\gamma}} - \frac{\alpha}{2^{1+1/\alpha}\,k'}$$

for $A > 0$. Substituting this lower bound into the left-hand side gives

$$\begin{aligned}\frac{1}{k'} + \Big(1 - \frac{1}{k'}\Big)(1+A)^{-\alpha} &\geq \frac{1}{k'} + \Big(1 - \frac{1}{k'}\Big)\left(\frac{1}{2\,k'^{\gamma}} - \frac{\alpha}{2^{1+1/\alpha}\,k'}\right) \\ &= \frac{1}{2\,k'^{\gamma}} + \frac{1}{k'} - \frac{1}{2\,k'^{\gamma+1}} - \frac{\alpha}{2^{1+1/\alpha}\,k'} + \frac{\alpha}{2^{1+1/\alpha}\,k'^{2}}.\end{aligned}$$

Since $k'^{\gamma} \geq 1$, we know that $-\frac{1}{2\,k'^{\gamma+1}} \geq -\frac{1}{2k'}$ and $\frac{\alpha}{2^{1+1/\alpha}\,k'^{2}} \geq 0$. Moreover, $2^{-1/\alpha} \leq \frac{1}{2}$ for $\alpha \in (0,1]$, which implies that $\frac{\alpha}{2^{1+1/\alpha}} \leq \frac{\alpha}{2}$, and therefore that

$$\frac{1}{k'} - \frac{1}{2\,k'^{\gamma+1}} - \frac{\alpha}{2^{1+1/\alpha}\,k'} \;\geq\; \frac{1}{k'} - \frac{1}{2k'} - \frac{\alpha}{2k'} \;=\; \frac{1-\alpha}{2k'} \;\geq\; 0$$

(with equality only when $\alpha = 1$). Combining these estimates gives

$$\frac{1}{k'} + \Big(1 - \frac{1}{k'}\Big)(1+A)^{-\alpha} \;\geq\; \frac{1}{2\,k'^{\gamma}} + \frac{1-\alpha}{2k'} \;\geq\; \frac{1}{2\,k'^{\gamma}}.$$

Finally, $(k'+1)^{\gamma} \geq k'^{\gamma}$ implies that $\frac{1}{2\,k'^{\gamma}} \geq \frac{1}{2\,(k'+1)^{\gamma}}$, proving (3). ■

We can now prove the master lower bound on $V(\tau', k')$ by induction on (k', τ'). The base cases are immediate. The inductive step plugs the one-variable minimum into the recurrence and then uses the balancing inequality.

Lemma 19 (Solving the recurrence) *For all $\tau' \geq 0$ and $k' \geq 1$,*

$$V(\tau', k') \;\geq\; \frac{m}{(\alpha+1)\tau^{\alpha}}\,\frac{(\tau')^{\alpha+1}}{2\,(k'+1)^{\gamma}}. \tag{4}$$

Proof We prove this by induction on (k', τ'). *Base cases.* If $\tau' = 0$ there is nothing to show. If $k' = 1$, then we have

$$V(\tau', 1) = \mathrm{OPT}_{\tau'} \;\geq\; \sum_{s=1}^{\tau'} m\Big(\frac{s}{\tau}\Big)^{\alpha} \;\geq\; \frac{m}{(\alpha+1)\tau^{\alpha}}(\tau')^{\alpha+1} \;\geq\; \frac{m}{(\alpha+1)\tau^{\alpha}}\,\frac{(\tau')^{\alpha+1}}{2\cdot 2^{\gamma}},$$

as $2 \cdot 2^{\gamma} \geq 1$. *Inductive step.* Now assume (4) holds for all (k'', τ'') with $k'' < k'$ and $0 \leq \tau'' \leq \tau'$. From (2), we know that

$$V(\tau', k') \;\geq\; \frac{1}{k'}\,\mathrm{OPT}_{\tau'+k'} + \Big(1 - \frac{1}{k'}\Big) \min_{0 \leq t \leq \tau'} \left[\frac{m}{(\alpha+1)\tau^{\alpha}}\,t^{\alpha+1} + V(\tau' - t, k' - 1)\right].$$

Using CEE and $m \leq f^{\star}(T)(\tau/T)^{\alpha}$, recall that we have

$$f^{\star}(s) \geq f^{\star}(T)\left(\frac{s}{T}\right)^{\beta_I} \geq f^{\star}(T)\left(\frac{s}{T}\right)^{\alpha} = f^{\star}(T)\left(\frac{\tau}{T}\right)^{\alpha}\left(\frac{s}{\tau}\right)^{\alpha} \geq m\left(\frac{s}{\tau}\right)^{\alpha}$$

for all $s \leq T$, which implies that

$$\frac{1}{k'}\,\mathrm{OPT}_{\tau'+k'} \;\geq\; \frac{1}{k'}\sum_{s=1}^{\tau'} m\Big(\frac{s}{\tau}\Big)^{\alpha} \;\geq\; \frac{m}{(\alpha+1)\tau^{\alpha}}\,\frac{(\tau')^{\alpha+1}}{k'}.$$

Applying the inductive hypothesis to $V(\tau' - t, k' - 1)$ and letting $y := t/\tau' \in [0, 1]$ gives

$$V(\tau', k') \geq \frac{m}{(\alpha+1)\tau^\alpha} (\tau')^{\alpha+1} \left[\frac{1}{k'} + \left(1 - \frac{1}{k'}\right) \min_{y \in [0,1]} \left\{ y^{\alpha+1} + \frac{1}{2\, k'^\gamma} (1-y)^{\alpha+1} \right\} \right].$$

By Lemma 17, the minimum equals $\left(1 + (2k'^\gamma)^{1/\alpha}\right)^{-\alpha}$. Lemma 18 then yields (4). ■

Finally, we apply Lemma 19 with $(\tau', k') = (T - k, k)$ to obtain the desired competitive ratio.

Theorem 20 (Upper bound) *Assume $T \geq 2k$. If we run $PTRR_\alpha$ with $\tau = T - k$ and $m := \frac{T-k}{T} f^\star(T)$, we obtain*

$$\mathbb{E}[\text{reward}] \geq \frac{1}{2^{\alpha+3}(\alpha+1)} \cdot \frac{\text{OPT}_T}{(k+1)^\gamma},$$

where $\gamma = \frac{\alpha}{\alpha+1}$. The competitive ratio is $O(k^{\alpha/(\alpha+1)})$.

Proof **(Overview.)** Two simple properties drive our analysis. First, the optimal arm is never abandoned when $\alpha > \beta_I$. Once we encounter f^*, we stay on it forever. Second, if we ever abandon a non-optimal arm at time t, the cumulative reward collected on that arm is at least the "area" under the threshold up to t

$$\sum_{s=1}^{t-1} f_i(s) \geq \frac{m}{(\alpha+1)\tau^\alpha} (t-1)^{\alpha+1}.$$

These two facts yield a value recurrence. Let $V(\tau', k')$ be the expected reward from a state with k' untouched arms and τ' 'free" pulls left. With probability $1/k'$ the next random pick is f^*, after which the algorithm earns $\text{OPT}_{\tau'+k'}$. Otherwise it spends t pulls on a bad arm, banks the area above, and recurses on $(\tau' - t, k' - 1)$. Minimizing over feasible t gives

$$V(\tau', k') \geq \frac{1}{k'} \text{OPT}_{\tau'+k'} + \left(1 - \frac{1}{k'}\right) \min_{t \in [0, \tau']} \left\{ \frac{m}{(\alpha+1)\tau^\alpha} t^{\alpha+1} + V(\tau' - t, k' - 1) \right\}.$$

Induct on (τ', k'). The recurrence reduces to a one-variable balance

$$\min_{y \in [0,1]} \left\{ u\, y^{\alpha+1} + v(1-y)^{\alpha+1} \right\},$$

which yields

$$V(\tau', k') \geq \frac{m}{(\alpha+1)\tau^\alpha} \cdot \frac{(\tau')^{\alpha+1}}{2(k'+1)^{\alpha/(\alpha+1)}}.$$

Finally, plug in (τ, k) with $\tau = T - k$, use $T \geq 2k$ so $\tau \geq T/2$, and upper-bound $\text{OPT}_T \leq c_2\, m\, (T/\tau)^\alpha T$ to obtain the desired result.

(Full proof of final step.) Applying Lemma 19 with $(\tau', k') = (T - k, k)$ (where the k extra time accounts for the "switching" steps when the keep-test first fails) and $T \geq 2k$ gives

$$\mathbb{E}[\text{reward}] \;\geq\; \frac{m}{(\alpha+1)\tau^\alpha} \cdot \frac{\tau^{\alpha+1}}{2\,(k+1)^\gamma} \;=\; \frac{m\,\tau}{2(\alpha+1)} \cdot \frac{1}{(k+1)^\gamma}.$$

Since $\mathrm{OPT}_T \le f^*(T) \cdot T \le c_2\, m\,(T/\tau)^{\beta_I} T \le c_2\, m\,(T/\tau)^{\alpha} T$ (by the CEE), we know that $m\tau \ge \frac{(\tau/T)^{1+\alpha}}{c_2}\,\mathrm{OPT}_T$, and therefore that

$$\mathbb{E}[\text{reward}] \;\ge\; \frac{1}{2(\alpha+1)c_2}\left(\frac{\tau}{T}\right)^{1+\alpha} \cdot \frac{\mathrm{OPT}_T}{(k+1)^{\gamma}} \ge \frac{1}{2^{\alpha+3}(\alpha+1)} \cdot \frac{\mathrm{OPT}_T}{(k+1)^{\gamma}},$$

as desired. ■

B.2. Lower Bound

To prove a matching lower bound, we construct hard a family of instances with $\beta_I = \beta$ on which every algorithm has expected approximation factor at least on the order of $k^{\beta/(\beta+1)}$. The idea is straightforward and mirrors the construction in [4]: one arm g is defined as $m(t/T)^{\beta}$, while the others all match it up to a time s and then flatten so that no deterministic sequence of pulls can safely discard one before investing s pulls. We let $\mathcal{D}_s$ denote the distribution that picks one arm uniformly at random to be g. Lemma 21 shows every instance drawn from $\mathcal{D}_s$ has $\beta_I = \beta$. For any deterministic schedule, Lemma 22 upper-bounds the expected reward by balancing OPT_T with the flat value $Tg(s)$. Lemma 23 gives $\mathrm{OPT}_T \ge mT/(\beta+1)$. Combining yields Lemma 24: with $x := s/T$,

$$\frac{\mathbb{E}[\mathrm{ALG}_T]}{\mathrm{OPT}_T} \;\le\; h(x) := \frac{1}{kx} + (\beta+1)x^{\beta}.$$

Lemma 25 minimizes h at $x^\star = [k\beta(\beta+1)]^{-1/(\beta+1)}$ and gives $h(x^\star) = \Theta\big(k^{-\beta/(\beta+1)}\big)$. Setting $s = \lfloor x^\star T \rfloor$ and assuming $T \ge 2/x^\star$, Lemma 26 controls discretization within a constant factor. This proves Theorem 27 for deterministic schedules. Yao's principle extends it to randomized algorithms with the same exponent. As before, a more detailed overview of the proof is included after the theorem.

Hard family. Fix $k \ge 2$, $T \ge 1$, and $m > 0$. For $s \in \{1, \dots, T\}$ define a good arm as

$$g(t) := m(t/T)^{\beta}, \quad t = 1, \dots, T.$$

Define for this s a bad arm by

$$f_{\mathrm{bad}}(t) = \begin{cases} g(t), & t \le s, \\ g(s), & t > s. \end{cases}$$

Let $\mathcal{D}_s$ be the distribution that chooses one arm uniformly at random to be g and sets all other arms to f_{bad}.

Lemma 21 (Membership and Concavity Envelope Exponent) *Every instance drawn from $\mathcal{D}_s$ has $\beta_I = \beta$.*

Proof For g, we know that $\mathrm{LE}(\beta)$ holds with equality. Since

$$f_{\mathrm{bad}}(T) = g(s) \le g(T) = m,$$

we likewise know that

$$f_{\mathrm{bad}}(t) \ge g(s)(t/T)^{\beta} = f_{\mathrm{bad}}(T)(t/T)^{\beta}$$

for all $t \le T$ and every f_{bad}. It follows that LE(β) holds for each arm, and therefore that $\beta_I \le \beta$.

Now suppose for the sake of contradiction that $\beta' < \beta$. Note that

$$m(t/T)^{\beta} < m(t/T)^{\beta'} = g(T)(t/T)^{\beta'},$$

which implies that g violates LE(β'). It follows that $\beta_I = \beta$. ■

To show that no algorithm can outperform ***PTRR***$_\beta$ on this distribution, we will consider the following 'generous game.' Fix a deterministic sequence S of T arm pulls. Let A denote the number of distinct arms that receive at least s pulls under S, and note that $A \le \lfloor \frac{T}{s} \rfloor$. Draw an arbitrary instance from $\mathcal{D}_s$, play S on this instance, and only afterwards reveal which arm was good. If S gave arm g at least s pulls, pay out a reward of OPT_T. Else provide the actual reward that S obtained. Note that paying OPT_T in certain cases can only increase the payoff of the sequence, which implies that the payoff of this game upper-bounds the expected reward $\mathbb{E}_{\mathcal{D}_s}[\mathrm{ALG}_T]$.

Lemma 22 (Generous evaluation) *For any deterministic algorithm* ALG*, let S be the induced sequence of T pulls, and let A be the number of arms that receive at least s pulls under S. Then*

$$\mathbb{E}_{\mathcal{D}_s}[\mathrm{ALG}_T(S)] \;\le\; \frac{A}{k}\,\mathrm{OPT}_T \;+\; \Big(1 - \frac{A}{k}\Big)\,T\,g(s) \;\le\; \frac{T}{ks}\,\mathrm{OPT}_T \;+\; T\,m\Big(\frac{s}{T}\Big)^{\beta}. \tag{5}$$

Proof Draw an instance from $\mathcal{D}_s$ and run S. With probability A/k, the good arm lies among those A arms, and the reward is at most OPT_T. Otherwise the good arm is pulled $< s$ times, which implies that any pull is worth at most $g(s)$, and therefore that the realized reward is at most $T \cdot g(s) = T \cdot m(s/T)^{\beta}$. Taking expectations gives the desired reward. ■

We now bound OPT_T.

Lemma 23 (Lower bound on the benchmark)

$$\mathrm{OPT}_T \;\ge\; \sum_{t=1}^{T} g(t) \;\ge\; \frac{mT}{\beta+1}.$$

Proof Note that

$$\mathrm{OPT}_T = \sum_{t=1}^{T} g(t) = m \sum_{t=1}^{T} (t/T)^{\beta} \ge mT^{-\beta} \int_0^T x^{\beta}\,dx = mT/(\beta+1),$$

as x^{β} is non-decreasing on $[0, T]$ for $\beta > 0$. ■

Combining the earlier generous bound with this lower bound gives a clean formula for the approximation ratio as a function of $x = s/T$.

Lemma 24 (Ratio at a given s) *For any deterministic algorithm* ALG *and $x := s/T \in \{1/T, \ldots, 1\}$,*

$$\frac{\mathbb{E}_{\mathcal{D}_s}[\mathrm{ALG}_T]}{\mathrm{OPT}_T} \;\le\; h(x) := \frac{1}{kx} + (\beta+1)\,x^{\beta}.$$

Proof Divide (5) by Lemma 23. ■

Simple calculus gives a unique minimizer x^* and the exact value $h(x^*)$.

Lemma 25 (Continuous minimizer) *h has a unique minimizer on $(0, 1]$ at*

$$x^\star \;:=\; \big[k\,\beta(\beta+1)\big]^{-1/(\beta+1)},$$

and

$$h(x^\star) \;=\; (\beta+1)^2\,[\beta(\beta+1)]^{-\beta/(\beta+1)}\,k^{-\beta/(\beta+1)}.$$

Proof First, note that

$$h'(x) = -1/(kx^2) + (\beta+1)\beta x^{\beta-1} = \frac{(\beta+1)\beta\,x^{\beta+1} - \frac{1}{k}}{x^2}$$

is strictly increasing in $x > 0$, which implies that it has a unique minimum

$$(\beta+1)\beta\,(x^\star)^{\beta+1} = \frac{1}{k} \quad\Longrightarrow\quad x^\star = \big[k\,\beta(\beta+1)\big]^{-1/(\beta+1)}.$$

At $x^\star$, we use the first-order condition to get

$$h(x^\star) = \frac{1}{kx^\star} + (\beta+1)(x^\star)^\beta = (\beta+1)\beta(x^\star)^\beta + (\beta+1)(x^\star)^\beta = (\beta+1)^2(x^\star)^\beta.$$

Since $(x^\star)^\beta = [k\,\beta(\beta+1)]^{-\beta/(\beta+1)}$, it follows that

$$h(x^\star) = (\beta+1)^2\,[\beta(\beta+1)]^{-\beta/(\beta+1)}\,k^{-\beta/(\beta+1)}.$$

■

Recall that s must be an integer. Following [4], we take $s = \lfloor x^*T \rfloor$ and require $x^*T \geq 2$. This simple condition ensures s/T lies between $x^*/2$ and x^*, inflating h by at most a constant factor.

Lemma 26 (Rounding) *Let $s := \lfloor x^\star T \rfloor$. If $x^\star T \geq 2$, then $s/T \in [x^\star/2,\, x^\star]$ and*

$$h\Big(\frac{s}{T}\Big) \;\leq\; \frac{3}{2}\,h(x^\star).$$

Proof Since $x^\star T \geq 2$, we know that $s/T \geq (x^\star T - 1)/T \geq x^\star/2$, and trivially $s/T \leq x^\star$. Let $s/T = ax^\star$ with $a \in [1/2, 1]$. Using $1/(kx^\star) = (\beta+1)\beta(x^\star)^\beta$, it follows that

$$\frac{h(ax^\star)}{h(x^\star)} = \frac{\beta/a + a^\beta}{\beta+1} \;\leq\; \frac{2\beta+1}{\beta+1} \;\leq\; \frac{3}{2},$$

since $1/a \leq 2$ and $a^\beta \leq 1$ for $a \in [1/2, 1]$, $\beta \in (0, 1]$. ■

We now plug our choice of s into the ratio bound. Because the bound holds for every deterministic schedule against D_s, it also holds for any randomized algorithm by linearity of expectation.

Theorem 27 (Lower Bound) *Fix $\beta \in (0,1]$ and $k \geq 2$. Suppose $T \geq \frac{2}{x^\star} = 2\left[\beta(\beta+1)\right]^{\frac{1}{\beta+1}} k^{\frac{1}{\beta+1}}$. Then, for every (possibly randomized) algorithm, there exists an instance with $\beta_I = \beta$ such that*

$$\frac{\mathbb{E}[\mathrm{ALG}_T]}{\mathrm{OPT}_T} \leq \frac{3}{2}(\beta+1)^2\left[\beta(\beta+1)\right]^{-\beta/(\beta+1)} k^{-\beta/(\beta+1)}.$$

Proof (Overview.) We construct a similar hard family to [4]. Define a "good" arm as the power curve $g(t) = m(t/T)^\beta$. Let the other $k-1$ arms copy g up to a breakpoint s and then flatten. Every arm satisfies LE(β), and the good arm violates any stricter floor, so $\beta_I = \beta$. Let $\mathcal{D}_s$ denote the distribution that picks one arm uniformly at random to be g.

Against any fixed schedule S of length T, let A denote the number of distinct arms that receive at least s pulls under S. Note that a "generous" evaluation that pays OPT_T if the good arm lies among those A and otherwise pays at most $Tg(s)$ provides an upper-bound for the expected reward of S under a random draw from D_s. Since $A \leq \lfloor T/s \rfloor$ and $\mathrm{OPT}_T \geq mT/(\beta+1)$, it follows that

$$\frac{\mathbb{E}[\mathrm{ALG}_T]}{\mathrm{OPT}_T} \leq \frac{1}{kx} + (\beta+1)x^\beta \quad \text{where } x := s/T \in (0,1].$$

Call the right-hand side $h(x)$. Simple calculus gives a unique minimizer

$$x^* = \left[k\beta(\beta+1)\right]^{-1/(\beta+1)}, \quad \text{which evaluates to } h(x^*) = (\beta+1)^2\left[\beta(\beta+1)\right]^{-\beta/(\beta+1)} k^{-\beta/(\beta+1)}.$$

Let $s = \lfloor x^\star T \rfloor$ and $T \geq 2/x^\star$. Using $\frac{1}{kx^*} = \beta(\beta+1)(x^*)^\beta$, we get $h(s/T) \leq \frac{3}{2}h(x^*)$, which implies that

$$\frac{\mathbb{E}[\mathrm{ALG}_T]}{\mathrm{OPT}_T} \leq \tfrac{3}{2}h(x^\star) = \frac{3}{2}(\beta+1)^2\left[\beta(\beta+1)\right]^{-\beta/(\beta+1)} k^{-\beta/(\beta+1)}.$$

By Yao's principle, the same bound holds for randomized algorithms.

(Full proof of final step.) Let $s = \lfloor x^\star T \rfloor$, and note that the condition on T ensures $x^\star T \geq 2$. Combine Lemmas 24, 25, and 26 to bound the ratio for any deterministic schedule against $\mathcal{D}_s$. Since this bound holds for every deterministic schedule, it also holds for any randomized algorithm by linearity of expectation (by Yao's principle). Lemma 21 guarantees $\beta_I = \beta$ for the constructed family. ■

Appendix C. Details for Sample Complexity Analysis

C.1. Results We Use from [15]

Definition 28 (Definition 1 in Arxiv verson of [15]) *Suppose the derandomized dual function $l_T^{P,\boldsymbol{z}}(\rho)$ is a piecewise constant function. The derandomized dual complexity of $\mathcal{D}$ w.r.t. P is given by $\mathcal{Q}_\mathcal{D} := \mathbb{E}_P\left[\mathbb{E}_z\left[q\left(l_T^{P,\boldsymbol{z}}(\cdot)\right)\right]\right]$, where $q(\cdot)$ is the number of pieces over which the function is piecewise constant.*

Having defined this complexity measure that is capable of characterizing the complexity of an algorithm family of interest, we now present the main theorem we apply from [15], a uniform convergence guarantee on learning the best value of the parameter α:

Theorem 29 (Theorem 6.1 in Arxiv Version of [15]) *Consider the Hyperparameter Transfer setup for any arbitrary $\mathcal{D}$[4] and suppose the derandomized dual function $l_T^{P,\boldsymbol{z}}(\alpha)$ is a piecewise constant function. For any $\epsilon, \delta > 0, N$ problems $\{P_i\}_{i=1}^N$ sampled from $\mathcal{D}$ with corresponding random coins $\{\boldsymbol{z}_i\}_{i=1}^N$ such that $N = O\left(\left(\frac{H}{\epsilon}\right)^2 (\log \mathcal{Q}_{\mathcal{D}} + \log \frac{1}{\delta})\right)$ are sufficient to ensure that with probability at least $1-\delta$, for all $\alpha \in \mathcal{P}$, we have that:*

$$\left| \frac{1}{N} \sum_{i=1}^{N} l_T^{P_i, \boldsymbol{z}_i}(\alpha) - \mathbb{E}_{P\sim\mathcal{D}} \left[l_T^P(\alpha) \right] \right| < \epsilon$$

C.2. Uniform Convergence Implies Population Loss Near-Optimality

We briefly argue why below.

Lemma 30 (Uniform convergence implies near-optimality in population loss) *Suppose $\hat{\alpha}$ is the minimizer of $\frac{1}{N}\sum_{i=1}^{N} l(P_i, \alpha)$ for N large enough to satisfy uniform convergence with error ϵ and $\alpha^\star \coloneqq \arg\min_\alpha \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \alpha) \right]$. Then,*

$$\left| \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \hat{\alpha}) \right] - \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \alpha^\star) \right] \right| < 2\epsilon \,.$$

Proof We can see this by adding and subtracting empirical losses:

$$\begin{aligned}
\left| \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \hat{\alpha}) \right] - \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \alpha^\star) \right] \right| = \Bigg| & \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \hat{\alpha}) \right] - \frac{1}{N} \sum_{i=1}^{N} l_T(P_i, \hat{\alpha}) && (6) \\
& + \frac{1}{N} \sum_{i=1}^{N} l_T(P_i, \hat{\alpha}) - \frac{1}{N} \sum_{i=1}^{N} l_T(P_i, \alpha^\star) && (7) \\
& + \frac{1}{N} \sum_{i=1}^{N} l_T(P_i, \alpha^\star) - \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \alpha^\star) \right] \Bigg| && (8) \\
\leq & \left| \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \hat{\alpha}) \right] - \frac{1}{N} \sum_{i=1}^{N} l_T(P_i, \hat{\alpha}) \right| + && (9) \\
& + \left| \frac{1}{N} \sum_{i=1}^{N} l_T(P_i, \alpha^\star) - \mathbb{E}_{P\sim\mathcal{D}} \left[l_T(P, \alpha^\star) \right] \right| && (10) \\
\leq & \, 2\epsilon && (11)
\end{aligned}$$

Note that the term in Eqn. 7 is negative, since $\hat{\alpha}$ is the minimizer of the empirical loss, and the last inequality follows from the uniform convergence guarantee. ■

C.3. Proof of Lemma 12

Proof We show this lemma by defining a instance[5]- and algorithm- specific tuple R_α . We define R_α such that it contains sufficient information to evaluate the loss of the algorithm on the fixed

4. To reiterate, we now consider $\mathcal{D}$ supported over $\mathcal{I} \times \Pi_k$, where $\Pi_k = \{\pi_k\}$ the set of permutations of $[k]$.
5. again, augmented instance

(augmented) instance. That is, since one value of the tuple gives rise to exactly one value of loss, we can count the number of possible values of loss by counting the number of such tuples.

First, fix an algorithm $\mathcal{A}_\alpha$. This algorithm proceeds by first generating a curve to which any chosen arm is compared, namely $c_\alpha(t) \coloneqq \left(\frac{t}{T}\right)^\alpha f^\star(T)$. Next, call the first arm chosen by the algorithm $f_1(t)$. The algorithm plays f_1 until the first time $t^{(1)}_{\text{stop}} \in \{1, 2, \dots T\}$ such that $f_1(t^{(1)}_{\text{stop}}) < \left(\frac{t^{(1)}_{\text{stop}}}{T}\right)^\alpha f^*(T)$. Similarly, we can calculate a $t^{(2)}_{\text{stop}}$ for the second arm played by the algorithm and so on. This provides us a tuple $R_\alpha \coloneqq (t^{(1)}_{\text{stop}}, t^{(2)}_{\text{stop}}, \dots, t^{(k)}_{\text{stop}})$ that provides enough information to compute the loss of the algorithm on the instance.

Now, we observe that as we vary α, the number of possible tuples that can be generated is upper bounded by the total number of possible tuples. Thus, we simply need to count the number of possible such tuples R_α. Since each $t^{(i)}_{\text{stop}}$ takes on a discrete value in $[T]$, there are at most T values an element in the tuple could take on, and since there are k elements in the tuple, the total number of such tuples is *at most* kT. ■

C.4. Average Regret

Now, let us consider specific instantiations for an H-well-behaved loss function.

Definition 31 *Suppose at time t, the reward collected by the algorithm is r_t. Define the* average regret *as:*

$$R(T) \coloneqq \frac{1}{T}\left(\sum_{t=1}^{T} r_t - \min_i \sum_{t=1}^{T} f_i(t)\right).$$

That is, the regret in hindsight is in reference to the best fixed arm.

Fact 32 *The average regret is H-well-behaved for $H = m^\star \coloneqq \max_{I\in\mathcal{I}} \max_{f\in I} f(T)$.*

Thus, if we are interested in solving the problem with respect to average regret, we get the sample complexity below:

Corollary 33 *For the Hyperparameter Transfer setting for the improving multi-armed bandits problem optimizing for averaged regret, $N = O\left(\left(\frac{m}{\epsilon}\right)^2 (\log kT + \log \frac{1}{\delta})\right)$ instances drawn from $\mathcal{D}$ suffice to get the uniform convergence guarantee in Theorem 29.*