Algorithm Design and Stronger Guarantees for the Improving Multi-Armed Bandits Problem

Avrim Blum[1] **Marten Garicano**[2] **Kavya Ravichandran**[1] **Dravyansh Sharma**[1,3]

[1]Toyota Technological Institute at Chicago, 6045 S. Kenwood Ave, Chicago, IL 60637
[2]University of Chicago, 5801 S. Ellis Ave, Chicago, IL 60637
[3]Northwestern University, 633 Clark St, Evanston, IL 60208

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 improving bandits, 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 study two objectives: cumulative reward and best arm identification and propose new parameterized families of bandit algorithms that address these objectives. We bound the sample complexity of learning the near-optimal algorithm from each family using offline data. We also perform empirical evaluations on standard hyperparameter tuning benchmarks. The first family we define 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. Our second family contains algorithms that both guarantee best-arm identification on well-behaved instances and revert to worst-case guarantees on poorly-behaved instances.

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 its payoff. Consider the development of a new technology, where investing in research increases its efficacy, albeit with diminishing returns. Similarly, for machine learning models, more training time (e.g. more epochs in stochastic gradient descent) leads to increased training accuracy. The *improving multi-armed bandits* (IMAB) framework, originally formalized by [HKR16; Pat+23], captures this setting.

Recent work [BR25] 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 at least some arms that have minimally concave (i.e., linear) reward 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 this work, we study the problem of designing algorithms for the improving bandits problem that achieve stronger performance guarantees on more benign problem instances. In particular, we design families of algorithms parameterized by some parameter(s) α , and we wish to learn the "best" algorithms from this family.

Consider the example of tuning hyperparameters such as learning rates 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). Often the number of arms is very large (e.g. a grid of multi-dimensional hyperparame-

Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence (UAI 2026), PMLR 337:585–619.

ters), so existing approximation factors that depend on the number of arms may be too impractical here. Furthermore, in many practical settings, we have access to historical data consisting of learning curves for training similar models on related tasks or datasets (similarly, we may have access to past clinical records, or data regarding click-through-rates for online advertising and recommendation). We can use these related previously-seen tasks to design our 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.

In this work, we first develop stronger approximation guarantees that depend on the strength of concavity of the reward functions. Our guarantees are achieved by a family of algorithms, parameterized by a parameter α that corresponds to the strength of concavity. We show that by setting α appropriately, we achieve the optimal approximation guarantees for every strength of concavity. In other words, if we know how "nice" our improving bandits instance is, we can use the appropriate algorithm from our family. On the other hand, if we do not have this information, we resort to a data-driven approach to learn the best algorithm parameter from the data. We obtain bounds on the sample complexity, that is, the number of IMAB instances sampled from the distribution that suffice to learn the best algorithm.

Next, we turn our attention to the best-arm identification (BAI) task. Approaches from prior work either guarantee that the best arm is exactly recovered on a sufficiently "nice" instance (using a notion of niceness that is different from the strength of concavity [Mus+24]), or select an arm such that the reward of the selected arm is a good approximation to the reward of the best arm on any (worst-case) instance [BR25]. However, the former may suffer from sub-optimal approximation ratios on more challenging instances, while the latter may fail to recover the exact best arm on the nicer instances. We propose a hybrid algorithm which resolves this gap in the literature and obtains a best-of-both-worlds guarantee. That is, on a nice instance, our algorithm will recover the best arm, while still guaranteeing the optimal approximation factor (up to constants) on the worst-case instance. We further show how to tune the parameters of our hybrid algorithm to obtain the near-optimal parameters on typical instances for a fixed problem domain, by giving bounds on the sample complexity of data-driven algorithm design.

1.1 OUR CONTRIBUTIONS

1. We introduce a parameter β to measure the "strength" of concavity of reward functions and develop algorithms with optimal approximation ratios for each β (Sections 3.1 and 3.2). For parameter $\beta \in (0, 1]$, we show that the optimal approximation ratio is $O(k^{\beta/(1+\beta)})$, which gives an improvement over the worst-case optimal bounds of $O(\sqrt{k})$ from prior work whenever $\beta < 1$.

2. Our algorithms that achieve the optimal guarantees constitute a parameterized family. In Section 3.3, we show how to tune the algorithm parameter, given access to similar instances drawn from a fixed but unknown distribution over IMAB instances. We employ techniques from data-driven algorithm design to bound the sample complexity of configuring our algorithm given reward curves of "training" instances. As in typical data-driven algorithm design settings, our results are optimal on average over the distribution. However, we can additionally reason about instances on which we can get strong per-instance guarantees due to the relationship between our algorithm family and sufficient conditions for stronger guarantees.

   Our approach of designing a parameterized family of assumptions (that includes worst-case instances for a fixed value of the parameter) and a corresponding tunable family of algorithms that give optimal algorithms for each value of the assumption parameter is novel in the context of data-driven algorithm design, and may be useful to design algorithms with powerful expected-case as well as per-instance worst-case guarantees in the context of other algorithm design problems.

3. In Section 4, we study best-arm identification (BAI) for improving bandits. Previous literature has a gap for this problem. Namely, there are algorithms that achieve exact BAI on certain "nice" instances but with sub-optimal worst-case performance [Mus+24], and the algorithms that achieve the near-optimal worst-case approximation [BR25] fail to identify the best arm on these easier instances. In Sections 4.1 and 4.2, we propose a hybrid approach that switches between an algorithm that explores many arms and our parameterized algorithm family above and obtains best-of-both-worlds guarantees. We also bound the sample complexity of simultaneously tuning the switching time of our hybrid approach and the concavity-strength parameter.

1.2 RELATED WORK

We improve the tight worst-case bounds of $\tilde{\Theta}(\sqrt{k})$ on the competitive ratio of IMAB by exploiting the strength of concavity of instances. Prior work [Met+22; Mus+24] also gives regret bounds for more benign IMAB instances. We develop algorithms that achieve a best-of-both-worlds guarantee by simultaneously achieving low regret on benign instances and asymptotically optimal competitive ratios on worst-case instances. We also extend techniques from the data-driven algorithm design framework [Bal20; SS25] to the IMAB setting to bound the sample complexity to tuning the parameters in our algorithms.

See Appendix A for a more detailed discussion on the related prior literature.

2 PRELIMINARIES

We formally define the improving multi-armed bandits (IMAB) framework introduced in [HKR16].

Definition 2.1. *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 nondecreasing as a function of t_i , which is the number of times that arm has been pulled so far. Following [HKR16; Pat+23; BR25], we further assume the arm rewards follow* diminishing-returns *(also referred to as concavity of reward functions), i.e., $f_i(t+1) - f_i(t) \leq f_i(t) - f_i(t-1)$ for each reward function f_i .*

Unlike in stochastic MAB models, the reward functions here are deterministic: the t-th pull of arm i returns $f_i(t)$, not a random reward with expectation $f_i(t)$. Let f^* denote the best arm at horizon T (known) in terms of cumulative reward. Let $\text{OPT}_t := \sum_{n=1}^{t} f^*(n)$. Note that the optimal strategy for an instance of this problem is to play the arm with highest cumulative reward for all time.

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 simple instances that show that getting non-trivial regret is impossible. For example, consider the two-armed instance where one arm increases linearly until time T and the other follows this arm until time $T/2$ and then flattens out. Then, no algorithm can distinguish between the arms for at least time $T/2$ and could therefore suffer $\Omega(T)$ regret. Therefore, we instead study the competitive ratio of the reward.

Definition 2.2. *Suppose on a fixed instance an algorithm accrues reward ALG and the reward of the optimal policy is OPT. Then, an algorithm is said to achieve competitive ratio c if $OPT/ALG \leq c$ for every IMAB instance.*

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, [Pat+23] consider deterministic algorithms and show tight upper and lower bounds at $\Theta(k)$. In a recent work, [BR25] 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.

Data-driven Algorithm Design Perspective and Problem Statement. In Sections 3.3 and E.7, we take a *data-driven algorithm design* perspective, i.e., we show that we can use samples from a distribution over instances to *adapt* to the setting at hand and get better guarantees. First, we describe the context and motivation for this perspective.

Framework. Suppose that we are in a setting where we must train and deploy a new machine learning model for a prediction problem each month. Each month, we must select hyperparameters and then train a deployable model with those hyperparameters. How can we leverage historical data to solve the hyperparameter selection problem? We neither want to assume that the best model on a given day is the best model on another day, nor even that the best hyperparameters for one day are the best hyperparameters for another day. Instead, let us simply assume that the algorithm we use for hyperparameter selection should be similar across days, and we wish to learn the best such algorithm. Now suppose we have historical data from, January to June, and we wish to use that *offline* data to learn which hyperparameter selection algorithm is most appropriate to deploy in July. Further, assume we can inspect the available data *completely*, i.e., for each model and hyperparameter setting trained in January through June, we have access to the *full* learning curves. Then, our goal will be to learn a good algorithm from a family of algorithms. In this work, we identify relevant families of algorithms to solve the problem of *identifying the best bandit algorithm for future hyperparameter tuning* and study what happens when we choose an algorithm from this family by using empirical risk minimization (ERM) with respect to the offline instances. Finally, with this "best" algorithm identified, we deploy it on future hyperparameter tuning problems. We conclude that if future instances are drawn from the same distribution as the training instances, then in expectation over the distribution, we will do well on the future instances.

Now, we formally define the preliminaries for our study of the improving multi-armed bandits problem in the data-driven algorithm design framework following prior work on stochastic bandits [SS25]. We consider loss functions of an algorithm in an algorithm family on an instance, defining $l(I, \boldsymbol{\alpha})$ as the loss of the algorithm with the parameters fixed to $\boldsymbol{\alpha} \in \mathcal{P}$ on instance I . We also define the dual of a loss function.

Definition 2.3. *A loss function $\ell : \mathcal{I} \times \mathcal{P} \to \mathbb{R}$ is* piecewise-H-bounded *if it is piecewise-constant and has a bounded range $[0, H]$. 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 a function of the parameter vector. In particular, the* dual of the loss function *is given by: $l_T^I(\boldsymbol{\alpha}) = \ell_T(I, \boldsymbol{\alpha})$.*

Finally, we formally define the problem we study.

Definition 2.4 (Hyperparameter Transfer Setting)**.** *Suppose we have a distribution $\mathcal{D}$ over $\mathcal{I}$, the space of instances I of the improving multi-armed bandits problem as defined in*

Definition 2.1. Consider a family of algorithms parameterized by a vector of parameters $\boldsymbol{\alpha} \in \mathcal{P}$. Finally, consider a piecewise-H-bounded loss, ℓ. We achieve sample complexity $N(\epsilon, \delta)$ in the Hyperparameter Transfer Setting *if, for any $\epsilon, \delta \in (0,1)$, given $N(\epsilon, \delta)$ instances sampled iid from $\mathcal{D}$, we identify $\hat{\boldsymbol{\alpha}}$ such that with probability $1-\delta$,*

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

3 USING STRENGTH OF CONCAVITY

In this section, we introduce a family of algorithms designed to address the question of providing sharper bounds for the improving multi-armed bandits problem. The design of this family is motivated by two factors: (1) the family must contain the algorithm that is known to provide the worst-case near-optimal guarantee of [BR25]; (2) there must exist an algorithm in the family that performs better than the general worst-case if instances satisfy a stronger regularity condition. In Section 3.1, we define the family. Then, in Section 3.2, we define a strengthening of concavity and show that a variant of [BR25] achieves optimal competitive ratio guarantees on instances that satisfy the strengthened property. In Section 3.3, we show that we can learn the best algorithm from this family for instances arising from a distribution with polynomially-many samples. Finally, in Section 3.4, we present empirical evidence on real learning-curve data that different instances prefer different values of α, corroborating the intuition that adapting the choice of this parameter to data is valuable.

The data-driven perspective we take in this section differs from standard worst case guarantees in several ways. On the one hand, we do not have to make stronger assumptions to get the guarantees and can instead *adapt* to the distribution of data. On the other hand, we must be in a setting where such a distributional assumption holds and we have access to other instances from the distribution.

3.1 ALGORITHM FAMILY

Each algorithm in the family, ***PTRR***$_\alpha$ for fixed α, is a slightly modified version of Algorithm 1 in [BR25]. Where [BR25] compare an arm to a linear threshold, in our algorithm, while playing arm i, we keep pulling it as long as $f_i(t_i) \geq m\left(\frac{t_i}{\tau}\right)^\alpha$, where τ is an internal horizon parameter (set to $T-k$ when T is known), m is an approximation of the maximum final pull, and t_i denotes the number of pulls of i so far. Note that setting $\alpha = 1$ recovers Algorithm 1 of [BR25]. When the inequality first fails, we abandon i and move to a uniformly random new arm, repeating until time T. In this section, we focus on the cumulative reward R accumulated by the algorithm. We will study the estimated best arm $\hat{i}$ in Section 4.

Algorithm 1 ***PTRR***$_\alpha$

Require: estimated max final pull m, internal horizon τ
1: $t \leftarrow 0, R \leftarrow 0, S \leftarrow \{1, \ldots, k\}$
2: **while** $t < T$ **do**
3: **sample** i uniformly from S
4: $S \leftarrow S \setminus \{i\}, t_i \leftarrow 0$
5: **while** $f_i(t_i) \geq m\left(\frac{t_i}{\tau}\right)^\alpha$ **do**
6: **pull** arm i
7: $t_i \leftarrow t_i + 1, t \leftarrow t+1, R \leftarrow R + f_i(t_i)$
8: **return** $R, \hat{i} \leftarrow \text{argmax}_i f_i(t_i)$

Definition 3.1. *Define the family of algorithms PTRR as $\{$**PTRR**$_\alpha : \alpha \in (0,1]\}$, where **PTRR**$_\alpha$ (α-Power-Thresholded Round Robin) is Algorithm 1.*

3.2 SHARPER COMPETITIVE RATIO

We will now argue that it is natural to consider the simple one–parameter family ***PTRR*** from Definition 3.1. First, we observe that our family contains the algorithm from [BR25] that is known to be optimal in the unrestricted case. We then show that there are algorithms in ***PTRR*** that improve the competitive ratio on every instance that satisfies a mild growth condition. A matching lower bound shows that these algorithms are optimal on such instances.

For simplicity, we assume that both T and $f^*(T)$ are known to the algorithm. In reality, our analysis—mirroring [BR25]—only requires that m lies within a constant factor of $f^*(T-k)$, which we achieve by setting $m := \frac{T-k}{T} \cdot f^*(T)$. We can avoid this requirement altogether by spending half our time learning m, thereby incurring only an extra $O(\log k)$ factor in our competitive ratio. This method is described in detail in section 5 of [BR25]. When T is also unknown, we embed the same half-explore/half-exploit method into a standard doubling schedule, preserving this $O(\log k)$ overhead. A full proof is included in Appendix B.3.

We first define a slightly stronger version of concavity.

Definition 3.2 (Per-arm β-Lower Envelope, LE(β))**.** *For any $\beta \in (0,1]$, we say that arm i satisfies LE(β) if*

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

Definition 3.3 (Concavity Envelope Exponent, β_I)**.** *For every instance I, define its Concavity Envelope Exponent β_I as*

$$\beta_I := \inf\{\beta \in (0,1] : \textit{ every arm in } I \textit{ satisfies } \mathrm{LE}(\beta)\}\,.$$

Remark 3.4. *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 Random Round Robin algorithm from [BR25], which ensures that *PTRR* preserves the $O(\sqrt{k})$ guarantee in the unrestricted case.

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 3.5. *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*

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

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

Proof Sketch. 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 a pessimistic 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 B.7). □

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 is optimal. *PTRR* attains this when $\alpha = \beta_I$.

Theorem 3.6 (Lower Bound)**.** *Fix* $\beta \in (0, 1]$ *and* $k \geq 2$*. For every (possibly randomized) algorithm, for* T *sufficiently large, there exists an instance with Concavity Envelope Exponent* $\beta_I = \beta$ *such that* $\frac{\mathbb{E}[ALG_T]}{OPT_T} \leq C_\beta k^{-\beta/(\beta+1)}$*, with* $C_\beta = \frac{3}{2}(\beta+1)^2 \left[\beta(\beta+1)\right]^{-\beta/(\beta+1)}$*. Equivalently, any such algorithm has an* $\Omega(k^{\beta/(\beta+1)})$ *competitive ratio.*

Proof Sketch. A full proof is in Appendix B.2. We construct an instance similar to [BR25]. Define a "good" arm as the power curve $g(t) = m(t/T)^\beta$, choose this arm at random, 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 $\beta_I = \beta$. At a high-level, we carefully pick t' such that any deterministic algorithm must suffer bad competitive ratio on the distribution over instances. By Yao's principle, this results in a lower bound for randomized algorithms. □

3.3 SAMPLE COMPLEXITY OF LEARNING α

In this section, we analyze the number of samples required to learn a near-optimal algorithm from the algorithm family *PTRR* (Definition 3.1). Previously, we considered a fixed α and showed its optimality for a fixed β. However, in practice, we may not know the true β and therefore cannot pick the optimal α. Now, suppose we have access to historical examples of improving multi-armed bandits problems (e.g., learning curves for the hyperparameter tuning problem on previous time periods of data). Then, we could hope to *learn* the best possible α for this distribution, provided we have sufficiently many samples. To analyze the sample complexity of this task, we extend techniques developed by [SS25] for stochastic multi-armed bandits.

Remark 3.7. *A related approach for meta-learning bandits is the* corralling *framework of [Aga+17; AMM21; Luo+22]. However, existing corralling techniques can only handle a finite number of hyperparameters. We further show that meta-learning even with a finite band of algorithms may not be possible without further assumptions in the improving bandits setting (see Appendix E.2).*

Derandomized Dual Complexity and result of [SS25]. We start by explaining how to use results from [SS25]. In their work, they defined a relevant quantity, the *derandomized dual complexity*, $Q_{\mathcal{D}}$, 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 [SS25] refer to the distribution $\mathcal{D}$ over instances, we instead consider the distribution over instances and permutations as defined above. By derandomizing in this way, we can immediately apply their results[1].

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

From the results of [SS25] (summarized in Appendix C.1 for convenience), 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. We know 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 on the offline instances, we can guarantee that the objective in Eqn. 1 is satisfied. This is essentially the empirical risk minimization (ERM) algorithm on the training instances. It remains to (1) bound $Q_{\mathcal{D}}$ for our problem and (2) investigate various reasonable piecewise-H-bounded losses.

Bounding derandomized dual complexity in our setting. Now, we compute a bound on the value of $Q_{\mathcal{D}}$, by bounding the number of possible behaviors of algorithms from $\mathcal{A}$. 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 (details in Appendix C.3).

Lemma 3.8. *For the family $\mathcal{A}$ defined in Defn. 3.1, the improving multi-armed bandits problem, and* any *piecewise-constant loss function, $Q_{\mathcal{D}} \leq kT$.*

Sample complexity. We present sample complexity results for generic piecewise-H-bounded loss functions by extending Theorem C.2 and Lemma 3.8. In Appendix C.4, we instantiate this for a concrete loss function.

Theorem 3.9. *The sample complexity of Hyperparameter Transfer in family $\mathcal{A}$ for the improving multi-armed bandits problem with generic piecewise-H-bounded loss is $N = O\left(\left(\frac{H}{\epsilon}\right)^2 (\log kT + \log \frac{1}{\delta}\right)$.*

While the above result guarantees sample efficiency for learning the best α, it is also interesting to consider the question of computational efficiency. In Appendix C.5, we provide a framework for implementing ERM that is efficient if the number of arms is a small constant. In Appendix C.6, we propose and analyze a potentially more practical algorithm to select a value of α that provides a good approximation for most instances in the distribution.

3.4 EMPIRICAL EVALUATION

We provide empirical evidence for the main data-dependent phenomenon suggested by our theory: the value of α used in running PTRR_α can affect cumulative reward, and the best value of this parameter can vary across instances. We use the LCDB 1.1 learning-curve dataset (CC-18 benchmarks, [YMV25]). We map each LCDB dataset to an IMAB instance as follows: the k arms are the LCDB learners, the time index corresponds to the number of samples seen, and the rewards arise from inverting the error rates through $r_{i,d}(t) = 1 - \mathrm{err}_{i,d}(t)$. We average over cross-validation to obtain a single learning curve for each dataset–learner pair, and retain only datasets for which all retained learners have finite values for all $t \in \{1, \ldots, T\}$. This yields 27 datasets with $k = 22$ arms each and a time horizon of $T = 44$. For each dataset d and each $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$, we run $PTRR_\alpha$ for each $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$ and report normalized cumulative reward $\mathbb{E}[\mathrm{PTRR}_\alpha(d)]/\mathrm{OPT}(d)$ (the reciprocal of the competitive ratio in Definition 2.2) averaged over 200 random arm orderings. Figure 1 shows that the α value maximizing this quantity varies across datasets. Figure 2 plots mean reward curves for three datasets to help visualize the underlying reward dynamics. Full details of the setup and results are provided in Appendix C.7.

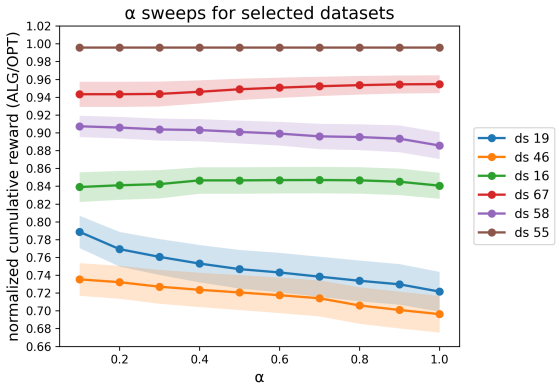

Figure 1: **Sensitivity of PTRR_α to α on selected LCDB instances ($T = 44$, $k = 22$).** Each curve corresponds to one CC-18 dataset d from LCDB 1.1 and reports the normalized cumulative reward $\mathbb{E}[\mathrm{PTRR}_\alpha(d)]/\mathrm{OPT}(d)$ as a function of $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$, where $\mathrm{OPT}(d) = \max_i \sum_{t=1}^{T} r_{i,d}(t)$ is the cumulative reward of the best fixed-arm policy in hindsight under the same horizon (which is not necessarily the optimal policy due to the absence of monotonicity, but is nonetheless a useful proxy), and $r_{i,d}(t) = 1 - \mathrm{err}_{i,d}(t)$ is the mean (over cross-validation) reward at anchor t for arm i. For each (d, α), $\mathbb{E}[\mathrm{PTRR}_\alpha(d)]$ is estimated by averaging over 200 random arm orderings. The shaded regions are pointwise 95% Student-t confidence intervals across the 200 runs (mean $\pm\, t_{0.975,199} \cdot \mathrm{sd}/\sqrt{200}$). The points at $\alpha = 1$ recover the performance of the algorithm from [BR25], and hence serve as a natural benchmark. The displayed datasets are selected to illustrate the range of sweep shapes observed across the full benchmark (near-flat curves, monotone trends, and interior maxima). For most datasets, performance differences across α are small relative to the confidence intervals, while a minority show a significant trend across α on this grid. Complete sweeps over all 27 usable datasets are included in Appendix C.7.

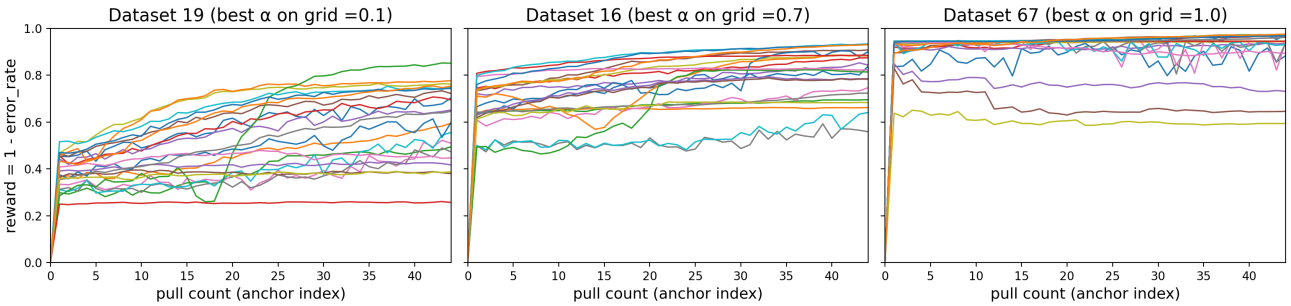

Figure 2: **Mean LCDB reward curves for three datasets with distinct best α values on the grid.** Each panel overlays the mean reward curves $r_{i,d}(t) = 1 - \mathrm{err}_{i,d}(t)$ across anchors t for all $k = 22$ arms on a single CC-18 dataset d. The title of each panel reports the value of $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$ that maximizes the estimated normalized cumulative reward $\mathbb{E}[\mathrm{PTRR}_\alpha(d)]/\mathrm{OPT}(d)$ at horizon $T = 44$ on that dataset. We selected these datasets to illustrate diverse best α values on the grid. Qualitatively, these plots suggest a mechanism consistent with the influence of α on PTRR_α: datasets preferring smaller α tend to exhibit early separation between 'good' and 'bad' arms, making aggressive abandonment beneficial, while datasets preferring larger α tend to exhibit closer early performance among many arms, making aggressive abandonment risky.

4 ACHIEVING BEST-OF-BOTH-WORLDS BEST ARM IDENTIFICATION

In this section, we bridge a gap in the literature: so far, algorithms can either identify the best arm in an instance that is sufficiently benign or identify an approximate best arm but not *both*, i.e., identify the best arm if possible and achieve the approximate goal if the exact goal cannot be achieved. To this end, we introduce a family of hybrid algorithms that address the task of identifying the arm with the highest final reward, the best arm identification (BAI) task[2]. Each algorithm in this family guarantees best arm identification whenever an instance is sufficiently benign, while simultaneously achieving optimal (up to constants) multiplicative guarantees on worst-case instances. We first provide examples that show that no existing algorithm achieves near-optimal results on both nice and worst-case instances. We define our family in Section 4.1. In Section 4.2, we formalize a 'niceness' condition under which it is possible to sample every suboptimal arm enough to safely discard it by the switch time. We then prove that $\mathrm{Hybrid}_{1,T/2}$ is guaranteed to identify the best arm on instances that satisfy this condition and, on all other instances, returns an arm whose expected maximum pull at time T competes optimally with the highest single pull in the instance. We also develop results for choosing α, B in a data-driven way.

Motivating Examples. Prior work gives two distinct types of guarantees for improving bandits—optimal approximation factors for worst-case instances [Pat+23; BR25], and better guarantees (sublinear policy regret, or small sample complexity of best arm identification) for nicer instances [HKR16; Met+22; Mus+24]. It is natural to ask whether we can achieve the best-of-both-worlds, that is, get almost optimal results on nice instances, while being within the optimal approximation factors (up to constants) in the worst-case. It turns out that the algorithms proposed in prior literature fail to achieve such a guarantee.

The following example shows that the algorithm of [BR25] has sub-optimal guarantees on nice instances.

Example 4.1. *The randomized algorithm of [BR25] may fail to identify the best-arm on instances where the UCB-style algorithms of [Met+22; Mus+24] find the best-arm. We set the reward function for the best arm as $f_{i^*}(t) = 1$ for all t, and for any other arm $i \neq i^*$ as $f_i(t) = \min\{\frac{t}{T}, \frac{1}{2}\}$, where T is the time horizon. We assume that k is large (in particular, $k \geq 4$). Now, the randomized round-robin algorithm selects the optimal arm as its first or second arm with probability at most $2/k$. Otherwise, it keeps playing a sub-optimal arm till time $T/2$, and a different sub-optimal arm for the rest of the time horizon. Thus, with probability at least $1 - \frac{2}{k}$, the best arm identified by the algorithm is sub-optimal by at least a factor of 2 (with respect to both its cumulative reward and its final pull). On the other hand, for the UCB-based algorithms, we can upper bound the number of exploratory pulls of the sub-optimal arms, and the algorithm succeeds in identifying the best arm i^*.*

The following example shows that the UCB-variant (R-UCBE) developed for the improving bandits BAI problem by [Mus+24] has sub-optimal worst-case performance.

Example 4.2. *The UCB-based algorithm of [Mus+24] for best-arm identification may output an arm with $\Omega(k)$ sub-optimal reward compared to the actual best arm on*

[2]While the reader may be more familiar with BAI in the sense of maximum mean in stochastic bandits, we note that in that case, we could equivalently study cumulative reward from a single arm.

some instances. We adapt the example used by [BR25] in their lower bound construction. Set $f_{i^*}(t) = t/T$ for all t for the optimal arm i^*, and for any other arm $i \neq i^*$ as $f_i(t) = \min\left\{\frac{t}{T}, \frac{1}{k}\right\}$. Due to the exploration term in UCB, while the arm rewards are identical, each arm gets pulled an equal number of times. By time T, each arm gets pulled T/k times, and all the arms appear identical to the algorithm. Thus, the best arm learned by the algorithm may be sub-optimal (with respect to both its cumulative reward and its final pull) by a factor of $\Omega(k)$. In contrast, the PTRR algorithm family guarantees a worst-case competitive ratio of $O(\sqrt{k})$ or smaller.

4.1 HYBRID ALGORITHM FAMILY

We now define the hybrid algorithm family. As before, we work in the standard improving-bandits setting with k arms, a known horizon T, and non-decreasing concave reward functions f_i. Each algorithm Hybrid$_{\alpha,B}$ has two stages. Stage 1 uses a UCB-style envelope: At each step, the algorithm computes a lower bound $L_i(n)$ and terminal upper bound $U_i(n)$ on the final reward $f_i(T)$ of every arm and pulls the arm with the largest optimistic estimate U_i. If the lower bound of one arm dominates the terminal upper bound of every other arm, Hybrid$_{\alpha,B}$ commits to this arm. If no commit occurs by time B, Stage 2 runs PTRR$_\alpha$ and finds an arm whose expected terminal reward is at least a substantial fraction of the best arm's.

For the terminal envelope, we define $L_i(t) := f_i(t), \triangle_i(t) := (T-t)\gamma_i(t-1)$, and $U_i(t) := L_i(t) + \triangle_i(t)$, where $\gamma_i(t-1) := f_i(t) - f_i(t-1)$. We set $U_i(0) := \infty$ to ensure first pulls. Using concavity and monotonicity, it is straightforward to prove that $L_i(t) \leq f_i(T) \leq U_i(t)$ for all i, t (see Lemma E.2 in Appendix E.3).

Definition 4.3. *Define the family of algorithms Hybrid* $:= \{Hybrid_{\alpha,B}(Algorithm\ 2) : \alpha \in (0,1]$ *and* $B \in [T]\}$.

4.2 BEST-OF-BOTH-WORLDS BAI GUARANTEES

In this section, we argue that it is meaningful to study the two-parameter family of *Hybrid* algorithms from Definition 4.3. We will do this by showing that *Hybrid* contains algorithms that simultaneously (i) guarantee best arm identification on sufficiently benign instances, and (ii) preserve tight (up to constants) multiplicative bounds for approximating the best arm on adversarial instances.[3]

We start by defining a class of 'sufficiently benign' instances and prove that our algorithm is guaranteed to return the best arm on all members of this class. If an instance is not in this class, Stage 2 pursues *approximate* BAI and runs $PTRR_\alpha$

[3]With additional information about stronger concavity, we achieve sharper bounds by using PTRR$_\alpha$ with appropriate α.

Algorithm 2 *Hybrid*$_{\alpha,B}$ BAI

Require: maximum final pull m, horizon τ
Stage 1: $t \leftarrow 0$
for each arm i **do**
  $t_i \leftarrow 0, L_i \leftarrow 0, U_i \leftarrow +\infty$
while $t < B$ **do**
  for each i with $t_i \geq 1$ **do**
    $L_i \leftarrow f_i(t_i)$
    $\gamma_i \leftarrow f_i(t_i) - f_i(t_i - 1)$
    $U_i \leftarrow L_i + (\tau - t) \cdot \gamma_i$
  $\hat{i} \leftarrow \arg\max_i L_i, \quad U_{\text{next}} \leftarrow \max_{j \neq \hat{i}} U_j$
  if $L_{\hat{i}} > U_{\text{next}}$ **then**
    return $\hat{i}$.
  $i' \leftarrow \arg\max_i (U_i - L_i)$
  pull i'; $t_{i'} \leftarrow t_{i'} + 1, t \leftarrow t + 1$
Stage 2: $\tau' \leftarrow (\tau - B) - k, m' \leftarrow \left(\frac{\tau'}{T}\right) \cdot m$
for each i **do**
  $g_i(s) \leftarrow f_i(t_i + s)$
return $\hat{i} \leftarrow$ arm returned by *PTRR*$_\alpha$ with parameters (m', τ') on $\{g_i\}$ for $T - B$ steps.

with α dependent on the strength of concavity ($\alpha = 1$ works for *all* instances). Having reverted to an approximate goal, we identify an arm $\hat{i}$ whose final reward satisfies $\mathbb{E}[f_{\hat{i}}(T)] \geq \Omega\left(k^{\frac{-\alpha}{1+\alpha}}\right) f^\star(T)$. We assume that both T and $f^*(T)$ are known to the algorithm, which we input as τ and m.

We will now define a condition (Definition 4.4) under which we can guarantee best arm identification. For any arm i and $\epsilon > 0$, the *terminal budget* h_i of arm i is defined as $h_i(\epsilon) := \min\{n \in \{2, \ldots, T\} : \triangle_i(n) \leq \epsilon\}$, where $\triangle_i(t) := (T-t)\,\gamma_i(t-1)$. For any instance I, its *Best Arm Gap* Δ_I is defined as $\Delta_I := f^\star(T) - \max_{j \neq i^*} f_j(T)$.

Definition 4.4 (Gap Clearance Condition, $GCC(B)$)**.** *For any $B \leq T$, we say that an instance satisfies the Gap Clearance Condition $GCC(B)$ if $\Delta_I > 0$ and $\sum_{i=1}^K h_i(\Delta_I/3) \leq B$.*

Under concavity, $\triangle_i(n)$ is an upper bound on the remaining possible increase in terminal value of arm i after n pulls: it denotes the most additional reward an adversary can "hide in the tail" by continuing with the last observed slope. The per-arm budget $h_i(\epsilon)$ is therefore the minimal number of pulls needed to ensure its optimistic terminal value is close to current lower envelope. Taking $\epsilon = \Delta_I/3$, once a suboptimal arm has been pulled this many times, its best possible continuation still loses to the best arm's lower envelope. Thus GCC(B) ensures that the total work needed to certify the best arm fits within budget B. If the sum exceeds B, concavity allows at least one suboptimal arm to remain plausibly optimal by time B, so no sound mid-horizon certificate can be guaranteed. For a comparison of this requirement to other 'niceness' conditions from the literature and a proof

of Theorem 4.5, see Appendix E.

Theorem 4.5 (best-of-both-worlds guarantees)**.** *Suppose an instance $I \in \mathcal{I}$ has Concavity Envelope Exponent $\beta_I \in (0, 1]$. Algorithm 2 with $\alpha \in (\beta_I, 1]$ satisfies the following:*

(1) If the instance further satisfies $\Delta_I > 0$, and $GCC(\theta)$ holds for $\theta \le T/2$, then the algorithm with $B \in (\theta, T/2]$ identifies and commits to the best arm $i^\star$ in Stage 1.

(2) If Stage 1 does not certify a best arm by time B, then Stage 2 finds an approximate best arm $\hat{i}$ such that $\mathbb{E}\big[f_{\hat{i}}(T)\big] \ge \Omega\Big(k^{\frac{-\alpha}{1+\alpha}}\Big)\, f^\star(T)$, where the expectation is over the randomness of the algorithm.

Proof Overview. (1) $\mathrm{GCC}(\theta)$ implies that the per–arm budgets $h_i(\Delta_I/3)$ sum to at most θ. Since the algorithm pulls the arm with the largest slack, we know that these budgets are met within $B \ge \theta$ pulls. Then every suboptimal arm has $U_j \le f^*(T) - 2\Delta_I/3$ and the best arm has $L_{i^\star} \ge f^*(T) - \Delta_I/3$, which implies that $L_{i^\star} \ge \max_{j \ne i^\star} U_j$. We commit to $i^\star$. (2) If no certificate fires by $B \le T/2$, running *PTRR*$_\alpha$ for T_{rem} steps yields an average reward within a $k^{\alpha/(1+\alpha)}$ factor of the residual optimum (up to constants). Using the fact that the best single pull is at least the average, and $\mathrm{OPT}^{\mathrm{res}}$ is at least a constant times $g^\star(T_{\mathrm{rem}})\, T_{\mathrm{rem}}$ by concavity, we get $\mathbb{E}[f_{\hat{i}}(T)] \ge \Omega\big(k^{-\alpha/(1+\alpha)}\big)\, f^\star(T)$. □

Finally, in Appendix E.7, we establish bounds on the sample complexity of tuning α, B in the *Hybrid* family.

5 CONCLUSION

We provide beyond worst-case guarantees for the improving multi-armed bandits problem. Employing the lens of data-driven algorithm design, we show that using sampled instances from a distribution, we can find algorithms that are near-optimal for instances arising from *that* distribution. We introduce two novel and interesting families of algorithms. Our first family achieves stronger competitive ratios that depend on the strength of concavity of the reward curves, and the second achieves a best-of-both-worlds guarantee on benign as well as worst-case instances which prior techniques in the literature fail to achieve.

Our data-driven results assume access to completed historical instances, such as full learning curves. A natural future direction is to study a more realistic online-transfer setting, where for each instance the learner only has a limited pull budget and must jointly decide how to explore the improving bandits instances and configure the algorithm hyperparameters for future instances. Furthermore, the concavity-envelope condition used in our analysis is only one sufficient condition for improved guarantees. Understanding its applicability and developing other sufficient conditions based on real learning-curve data is an important empirical and theoretical direction.

ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation under grants CCF-2212968, ECCS-2216970, 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

[GR16] 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*. 2016, pp. 123–134.

[HKR16] 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. New York, New York, USA: AAAI Press, 2016, pp. 1562–1570. ISBN: 9781577357704.

[Aga+17] Alekh Agarwal, Haipeng Luo, Behnam Neyshabur, and Robert E Schapire. "Corralling a band of bandit algorithms." In: *Conference on Learning Theory*. PMLR. 2017, pp. 12–38.

[Bal+17] Maria-Florina Balcan, Vaishnavh Nagarajan, Ellen Vitercik, and Colin White. "Learning-theoretic foundations of algorithm configuration for combinatorial partitioning problems." In: *Conference on Learning Theory*. PMLR. 2017, pp. 213–274.

[BDV18] Maria-Florina Balcan, Travis Dick, and Ellen Vitercik. "Dispersion for data-driven algorithm design, online learning, and private optimization." In: *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2018, pp. 603–614.

[Bal20] Maria-Florina Balcan. "Data-Driven Algorithm Design (Book Chapter)." In: *Beyond Worst-Case Analysis of Algorithms, Tim Roughgarden (Ed)*. Cambridge University Press, 2020.

[AMM21] Raman Arora, Teodor Vanislavov Marinov, and Mehryar Mohri. "Corralling stochastic bandit algorithms." In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 2116–2124.

[BDS21] Avrim Blum, Chen Dan, and Saeed Seddighin. "Learning complexity of simulated annealing." In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR. 2021, pp. 1540–1548.

[BIW22] Peter Bartlett, Piotr Indyk, and Tal Wagner. "Generalization bounds for data-driven numerical linear algebra." In: *Conference on Learning Theory (COLT)*. PMLR. 2022, pp. 2013–2040.

[Luo+22] Haipeng Luo, Mengxiao Zhang, Peng Zhao, and Zhi-Hua Zhou. "Corralling a larger band of bandits: A case study on switching regret for linear bandits." In: *Conference on Learning Theory*. PMLR. 2022, pp. 3635–3684.

[Met+22] Alberto Maria Metelli, Francesco Trovo, Matteo Pirola, and Marcello Restelli. "Stochastic rising bandits." In: *International Conference on Machine Learning*. PMLR. 2022, pp. 15421–15457.

[Kho+23] Mikhail Khodak, Ilya Osadchiy, Keegan Harris, Maria-Florina Balcan, Kfir Y Levy, Ron Meir, and Steven Z Wu. "Meta-learning adversarial bandit algorithms." In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 35441–35471.

[Pat+23] 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*. Thirty-Second International Joint Conference on Artificial Intelligence {IJCAI-23}. Macau, SAR China: International Joint Conferences on Artificial Intelligence Organization, Aug. 2023, pp. 4100–4108. ISBN: 978-1-956792-03-4. DOI: 10.24963/ijcai.2023/456. URL: https://www.ijcai.org/proceedings/2023/456 (visited on 01/22/2024).

[BSS24] Maria-Florina Balcan, Christopher Seiler, and Dravyansh Sharma. "Accelerating ERM for data-driven algorithm design using output-sensitive techniques." In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 72648–72687.

[BS24] Maria-Florina Balcan and Dravyansh Sharma. "Learning Accurate and Interpretable Decision Trees." In: *Uncertainty in Artificial Intelligence*. PMLR. 2024, pp. 288–307.

[CB24] Hongyu Cheng and Amitabh Basu. "Learning cut generating functions for integer programming." In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 61455–61480.

[Jin+24] 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.

[Kho+24] Mikhail Khodak, Edmond Chow, Maria-Florina Balcan, and Ameet Talwalkar. "Learning to Relax: Setting Solver Parameters Across a Sequence of Linear System Instances." In: *The Twelfth International Conference on Learning Representations (ICLR)* (2024).

[Mus+24] Marco Mussi, Alessandro Montenegro, Francesco Trovò, Marcello Restelli, and Alberto Maria Metelli. "Best arm identification for stochastic rising bandits." In: *Proceedings of the 41st International Conference on Machine Learning*. 2024, pp. 36953–36989.

[SO24] Shinsaku Sakaue and Taihei Oki. "Generalization bound and learning methods for data-driven projections in linear programming." In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 12825–12846.

[Sha24] Dravyansh Sharma. "No internal regret with non-convex loss functions." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 13. 2024, pp. 14919–14927.

[BNS25] Maria Florina Balcan, Anh Tuan Nguyen, and Dravyansh Sharma. "Algorithm Configuration for Structured Pfaffian Settings." In: *Transactions on Machine Learning Research* (2025).

[BR25] Avrim Blum and Kavya Ravichandran. "Nearly-tight Approximation Guarantees for the Improving Multi-Armed Bandits Problem." In: *Proceedings of The 36th International Conference on Algorithmic Learning Theory*. Ed. by Gautam Kamath and Po-Ling Loh. Vol. 272. Proceedings of Machine Learning Research. PMLR, Feb. 2025, pp. 228–245. URL: https://proceedings.mlr.press/v272/blum25a.html.

[Cha+25] Vaggos Chatziafratis, Ishani Karmarkar, Yingxi Li, and Ellen Vitercik. "Accelerating data-driven algorithm selection for combinatorial partitioning problems." In: *The Thirty-ninth Annual Conference on Neural Information Processing Systems*. 2025.

[SS25] Dravyansh Sharma and Arun Suggala. "Offline-to-online hyperparameter transfer for stochastic bandits." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 39. 19. 2025, pp. 20362–20370. URL: https://arxiv.org/abs/2501.02926.

[YMV25] Cheng Yan, Felix Mohr, and Tom Julian Viering. "LCDB 1.1: A Database Illustrating Learning Curves Are More Ill-Behaved Than Previously Thought." In: *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2025.

[Sha26] Dravyansh Sharma. "Gradient Descent with Provably Tuned Learning-rate Schedules." In: *The 29th International Conference on Artificial Intelligence and Statistics*. 2026.

A ADDITIONAL RELATED WORK

Improving (Rising) Bandits. Initially, [HKR16] formulated the improving (and decaying) bandits problem where payoffs increase the more the arm is played, and obtain sub-linear policy regret. [Pat+23] show that a linear regret is unavoidable in the worst-case and obtain a tight $\Theta(k)$ on the competitive ratio of deterministic algorithms. Then, [BR25] extend the development on this problem to randomized algorithms, achieving an $O(\sqrt{k}\log k)$ upper bound, nearly matching their $\Omega(\sqrt{k})$ lower bound. A related line of work is that on (deterministic, rested) *rising* bandits [Met+22; Mus+24]. This line of work bounds the instance-dependent policy regret on benign instances but otherwise considers similar increasing reward functions. We note that policy regret is a stronger notion of regret than *external regret*, where an algorithm is compared to the best action on *the same* set of rewards. In policy regret (and in the competitive ratios we consider), an algorithm is compared to the best policy for the sequence of rewards generated by *playing that policy*.

Data-Driven Algorithm Design. Data-driven algorithm design is emerging as a powerful approach for using machine learning to design algorithms that have strong performance on "typical" input instances, as opposed to worst-case or average-case analysis [GR16; Bal+17; BDV18]. The approach is known to be effective in both statistical and online learning settings [Bal20], and, under this paradigm, a growing line of research has successfully developed techniques for designing several fundamental algorithms in machine learning and beyond (e.g. [BDS21; BIW22; Jin+24; Kho+24; BS24; CB24; SO24; Sha24; BNS25; Sha26]). Recent work [SS25] 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 [Kho+23] shows how to tune hyperparameters for adversarial bandit algorithms in an online-with-online meta-learning setting. While we focus on statistical complexity of tuning our bandit algorithms, an interesting future direction is to give computationally efficient algorithms [BSS24; Cha+25].

B FULL PROOFS FOR SHARPER COMPETITIVE RATIO

In this Appendix, we give full proofs of our optimal sharper competitive ratio for each value of the Concavity Envelope Exponent β_I satisfied by the instance I, along with a formal proof of the "doubling trick" applied in our context for completeness. Our analysis extends the techniques due to [BR25].

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 B.1). Second, abandoning any non-optimal arm after t pulls yields at least the "area under the envelope" up to t (Lemma B.2). These two facts give the value recurrence in Lemma B.3: 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 B.4). A clean lower bound on this minimum (Lemma B.5) 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 B.6). Finally, substituting $(\tau', k') = (T-k, k)$ and OPT_T's comparison to m gives Theorem B.7. A more detailed overview of the proof is included after the theorem.

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

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

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

Lemma B.1 (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 $f_\star(t) \geq f_\star(T)\left(\frac{t}{T}\right)^{\beta_I}$ for all $0 \leq t \leq T$. Moreover, recall that $m \leq f^*(T)\left(\frac{\tau}{T}\right)^{\alpha}$. Since $\frac{\tau}{T} \in (0,1]$ and $\alpha > \beta_I$, it follows that

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

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

Lemma B.2 (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 $1 \leq s < t_i$, we know that $f_i(s) \geq m(s/\tau)^\alpha$. Summing and using the monotonicity of x^α on $[0,\infty)$ gives

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

□

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 B.2, and recurse on $(\tau' - t, k' - 1)$. To ensure a worst-case guarantee, we take the minimum over all feasible abandonment times t.

Lemma B.3 (Value recurrence)**.** *For all $\tau' \geq 0$, $k' \geq 1$,*

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

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

Proof. With probability $1/k'$ the next arm is f^* and, by Lemma B.1, 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 B.2 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 B.4 (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 B.5 (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

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

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

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

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 B.6 (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

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

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

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

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

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

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

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

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

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

By Lemma B.4, the minimum equals $\big(1+(2k'^{\gamma})^{1/\alpha}\big)^{-\alpha}$. Lemma B.5 then yields (4). □

Finally, we apply Lemma B.6 with $(\tau',k') = (T-k,k)$ to obtain the desired competitive ratio.

Theorem B.7 (Upper bound)**.** *Assume $T \ge 2k$. If we run $PTRR_\alpha$ with $\tau = T-k$ and $m := \frac{T-k}{T} f^\star(T)$, we obtain*

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

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

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

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 $\mathrm{OPT}_{\tau'+k'}$. 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

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

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

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

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

(**Full proof of final step.**) Applying Lemma B.6 with $(\tau',k') = (T-k,k)$ (where the k extra time accounts for the "switching" steps when the keep-test first fails) and $T \ge 2k$ gives

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

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 [BR25]: 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 B.8 shows every instance drawn from $\mathcal{D}_s$ has $\beta_I = \beta$. For any deterministic schedule, Lemma B.9 upper-bounds the expected reward by balancing OPT_T with the flat value $Tg(s)$. Lemma B.10 gives $\mathrm{OPT}_T \geq mT/(\beta+1)$. Combining yields Lemma B.11: with $x := s/T$,

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

Lemma B.12 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 $T \geq 2/x^\star$, Lemma B.13 controls discretization within a constant factor. This proves Theorem B.14 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 $k \geq 2$, $T \geq 1$, and $m > 0$. For $s \in \{1, \ldots, T\}$ define a good arm as

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

Define for this s a bad arm by

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

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 B.8 (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

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

we likewise know that

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

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

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 $\mathrm{LE}(\beta')$. 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 $A \leq \lfloor \frac{T}{s} \rfloor$. 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 B.9 (Generous evaluation)**.** *For any deterministic algorithm* ALG*, let S be the sequence played by* ALG *when all arms are f_{bad}, and let A be the number of arms that receive at least s pulls under S. Then*

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

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 B.10 (Lower bound on the benchmark)**.**

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

Proof. Note that

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

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 B.11 (Ratio at a given s)**.** *For any deterministic algorithm* ALG *and* $x := s/T \in \{1/T, \ldots, 1\}$,

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

Proof. Divide (5) by Lemma B.10. □

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

Lemma B.12 (Continuous minimizer)**.** *h has a unique minimizer on $(0, 1]$ at*

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

and

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

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

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

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

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

□

Recall that s must be an integer. Following [BR25], 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 B.13 (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 $s/T = a x^\star$ with $a \in [1/2, 1]$. Using $1/(k x^\star) = (\beta+1)\beta (x^\star)^\beta$, it follows that

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

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 B.14 (Lower Bound). *Fix $\beta \in (0, 1]$ and $k \geq 2$. Suppose $T \geq \frac{2}{x^\star} = 2\,[\beta(\beta+1)]^{\frac{1}{\beta+1}}\, k^{\frac{1}{\beta+1}}$. Then there exists a distribution on instances with $\beta_I = \beta$ such that for every (possibly randomized) algorithm,*

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

Proof. (**Overview.**) We construct a similar hard family to [BR25]. 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

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

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

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

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 B.11, B.12, and B.13 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 B.8 guarantees $\beta_I = \beta$ for the constructed family. □

B.3 DOUBLING TRICK FOR UNKNOWN T

We remove the need to know T by using the doubling schedule and exploration procedure of [BR25]. At a high level, we start with a guess $T_0 = 4k$, pretend this is the horizon, and spend $T'/2$ steps using the same adaptation of **Algorithm 2** of [BR25] that they use to estimate $\widehat{m}$, the reward of the best arm, for the relevant time scale. After this, we spend $T'/2$ steps exploiting (running $PTRR_\alpha$) with $\tau' := \frac{T'}{2} - k$ and $m = \frac{\widehat{m}}{2 \cdot 16^\alpha}$ (we shrink $\hat{m}$ to ensure we don't discard the best arm). If time remains after T' steps, we double T' and repeat. This yields the below result:

Theorem B.15 (Unknown-T guarantee)**.** *Assume $\beta_I \in (0,1)$ and fix $\alpha \in (\beta_I, 1)$, $\gamma = \alpha/(\alpha+1)$. If $T > 4k$, then the doubling trick above achieves*

$$\mathbb{E}[\mathrm{ALG}_T] \geq \frac{1}{2048 \cdot 16^{\alpha}(\alpha+1)\log(128k)} \cdot \frac{\mathrm{OPT}_T}{(k+1)^{\gamma}}.$$

Proof. Consider i such that $T' = 2^i T_0$ is the last iteration for which an explore / exploit cycle was completed. Define $T'' := \sum_{j=0}^{i} 2^j T_0 = (2^{i+1}-1)T_0$. Then $T'' < T \leq 2T''$. We will argue two things: first, we will show that spending $T'/2$ time on the procedure for estimating m provides a sufficiently good estimate for m for our purposes; second, we will quantify the gap between spending $T'/2$ collecting reward from the instance based on our estimated m and T time spent on the optimal arm.

Let $\tau := T'/2 - k$. Let $\hat{m}$ denote the estimated maximum value at horizon $\tau + k$ from running the estimation procedure for time $T'/2$ (from time $T'' - T'$ to $T'' - T'/2$). From the analysis in [BR25], we know that $\frac{1}{2}\widehat{m} \leq f^{\star}(\tau) \leq 2\widehat{m}$ with probability at least $\frac{1}{\log(128k)}$, and that

$$\frac{T}{\tau} = \frac{T}{\frac{T'}{2} - k} \leq \frac{T}{\frac{T'}{4}} = 4\,\frac{T}{T''}\frac{T''}{T'} \leq 4 \cdot 2 \cdot 2 = 16\,.$$

Let $m := \frac{\hat{m}}{2 \cdot 16^{\alpha}}$, and run *PTRR*$_\alpha$ with m, τ for $\frac{T'}{2}$ steps.

Since $\tau/T \geq 1/16$ and $\widehat{m} \leq 2f^{\star}(\tau) \leq 2f^{\star}(T)$, we know that

$$m \leq \frac{2f^{\star}(T)}{2 \cdot 16^{\alpha}} \leq f^{\star}(T)\left(\frac{\tau}{T}\right)^{\alpha},$$

and therefore that

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

for all $t \leq T$. It follows that *PTRR*$_\alpha$ again does not switch away from the best arm f^*.

Now, we proceed to the second part of our argument. Applying Lemma B.6 with $(\tau', k') = (\tau, k)$ gives

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

Since $m = \widehat{m}/(2 \cdot 16^{\alpha})$ and $\tau/T \geq 1/16$, it follows that

$$\mathbb{E}[\text{exploit}] \geq \frac{\widehat{m}\tau}{4(\alpha+1)16^{\alpha}(k+1)^{\gamma}} \geq \frac{\widehat{m}T}{64(\alpha+1)16^{\alpha}(k+1)^{\gamma}}.$$

Since $f^{\star}(T) \leq 16f^{\star}(\tau) \leq 32\widehat{m}$, it further follows that $\mathrm{OPT}_T \leq f^{\star}(T)T \leq 32\widehat{m}T$, and therefore that

$$\frac{\mathbb{E}[\text{exploit}]}{\mathrm{OPT}_T} \geq \frac{1}{2048(\alpha+1)16^{\alpha}} \cdot \frac{1}{(k+1)^{\gamma}}.$$

Multiplying by the $1/\log(128k)$ selection probability gives

$$\mathbb{E}[\mathrm{ALG}_T] \geq \frac{1}{2048(\alpha+1)16^{\alpha}\log(128k)} \cdot \frac{\mathrm{OPT}_T}{(k+1)^{\gamma}}.$$

□

C DETAILS FOR SAMPLE COMPLEXITY ANALYSIS

We now provide some background from prior work and full proofs for the sample complexity of tuning the α parameter in our algorithm family $PTRR_\alpha$.

C.1 RESULTS FROM PRIOR WORK

Definition C.1 (Definition 1 in Arxiv version of [SS25]). *Suppose the derandomized dual function $l_T^{I,\boldsymbol{z}}(\rho)$ is a piecewise constant function. The derandomized dual complexity of $\mathcal{D}$ w.r.t. instance I is given by $\mathcal{Q}_{\mathcal{D}} \coloneqq \mathbb{E}_{I\sim\mathcal{D}}\left[\mathbb{E}_z\left[q\left(l_T^{I,\boldsymbol{z}}(\cdot)\right)\right]\right]$, 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 [SS25], a uniform convergence guarantee on learning the best value of the parameter α:

Theorem C.2 (Theorem 6.1 in arXiv Version of [SS25]). *Consider the Hyperparameter Transfer setup for any arbitrary $\mathcal{D}$[4] and suppose the derandomized dual function $l_T^{I,\boldsymbol{z}}(\alpha)$ is a piecewise constant function. For any $\epsilon, \delta > 0$, N problems $\{I_i\}_{i=1}^N$ 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 $\boldsymbol{\alpha}\in\mathcal{P}$, we have that:*

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

Note that by the argument below, this guarantee solves the Hyperparameter Transfer Setting described in Definition 2.4.

C.2 UNIFORM CONVERGENCE IMPLIES POPULATION LOSS NEAR-OPTIMALITY

We recall the standard argument below for completeness.

Lemma C.3 (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:

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

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. □

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

C.3 PROOF OF LEMMA 3.8

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 (augmented) instance. That is, since one value of the tuple gives rise at most one value of loss, we can bound 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-bounded loss function.

Definition C.4. *Suppose at time t, the reward collected by the algorithm is r_t. Define the* average regret *as:*

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

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

Fact C.1. *The average regret is H-bounded for $H = m$.*

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

Corollary C.5. *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 C.2.*

Note that the guarantee of this corollary is that the parameters found will produce a sequence of pulls that has regret that is close to the regret of the sequence of pulls induced by the *best set of parameters*.

C.5 COMPUTATIONAL COMPLEXITY OF ERM

In Section 3.3 we analyze the sample complexity of selecting α by ERM over the family $\mathcal{A} = \{\text{PTRR}_\alpha : \alpha \in (0, 1]\}$ in the Hyperparameter Transfer Setting (Definition 2.6), using the uniform convergence results from Appendix B.1 and Appendix B.2. Here we add to that analysis by showing that, given offline access to the full learning curves, the ERM minimizer over the continuous domain $\mathcal{P} = (0, 1]$ can be computed exactly. Our implementation is efficient for small k, and we leave open the question of efficient exact or approximate implementation for large k.

As in Section 3.3, we derandomize Algorithm 1 by augmenting each instance with a permutation $z \in \Pi_k$ that fixes the order in which arms are sampled from S. We therefore work with i.i.d. samples $(I_1, z_1), \dots, (I_N, z_N) \sim (D')^N$. For fixed (I, z) and α, the execution of PTRR_α is deterministic. Let $l_T^{I,z}(\alpha)$ denote the corresponding (derandomized) dual loss value at horizon T (Appendix B.1).

If $S = \emptyset$ while $t < T$, we halt (this does not occur for appropriate alpha, but ensures that the algorithm is well-defined for all α and does not affect the arguments below).

[5] again, augmented instance

Finite critical set for fixed (I, z) .

Fix an augmented instance (I, z) with reward curves $\{f_i(t)\}_{i\in[k],\, t\in\{0,1,\ldots,T\}}$, and recall that Algorithm 1 makes α-dependent decisions through the keep-test

$$f_i(t_i) \;\geq\; m\Big(\frac{t_i}{\tau}\Big)^{\alpha}, \qquad t_i \in \{0, 1, \ldots, T\}. \tag{12}$$

For each pair (i, t) with $i \in [k]$ and $t \in \{1, \ldots, T\} \setminus \{\tau\}$, define $c_{i,t} \in (0, 1]$ to be the unique solution (if it exists) to the equality

$$f_i(t) \;=\; m\Big(\frac{t}{\tau}\Big)^{\alpha}, \qquad \alpha \in (0, 1]. \tag{13}$$

Uniqueness holds because for fixed $t \neq \tau$, the map $\alpha \mapsto m(t/\tau)^{\alpha}$ is strictly monotone. Let the critical set be

$$\mathcal{C}(I) \;:=\; \big\{c_{i,t} \in (0, 1] : i \in [k],\; t \in \{1, \ldots, T\} \setminus \{\tau\},\; \text{and (13) has a solution in } (0, 1]\big\}.$$

Then clearly $|\mathcal{C}(I)| \leq kT$.

Lemma C.6 (Constancy between critical values)**.** *Fix (I, z). The function $\alpha \mapsto l_T^{I,z}(\alpha)$ is constant on each connected component of $(0, 1] \setminus \mathcal{C}(I)$. Moreover, any change in $l_T^{I,z}(\alpha)$ can occur only at $\alpha \in \mathcal{C}(I)$.*

Proof. For each $i \in [k]$ and $t \in \{0, 1, \ldots, T\}$, define the predicate

$$\mathsf{Test}_{i,t}(\alpha) \;:=\; \mathbf{1}\Big[\, f_i(t) \;\geq\; m\Big(\frac{t}{\tau}\Big)^{\alpha}\,\Big].$$

When $t = \tau$, the right-hand side equals m and is independent of α. When $t \neq \tau$, strict monotonicity of $\alpha \mapsto m(t/\tau)^{\alpha}$ implies that $\mathsf{Test}_{i,t}(\alpha)$ can change value only at the unique equality solution $c_{i,t}$ (if it exists). Therefore, on any connected component $U \subset (0, 1] \setminus \mathcal{C}(I)$, every predicate $\mathsf{Test}_{i,t}(\alpha)$ is constant for all $\alpha \in U$.

Now fix $\alpha, \alpha' \in U$ and couple the executions of PTRR$_\alpha$ and PTRR$_{\alpha'}$ on the same augmented instance (I, z). Because z fixes the order of arms sampled from S, the only remaining branching in Algorithm 1 is the outcome of the keep-test (12) at the current arm and current pull-count. Each such branch is determined by some $\mathsf{Test}_{i,t}(\cdot)$, and all of these predicates agree on U. Therefore both executions take identical branches at every step and therefore generate the same pull sequence and the same accumulated reward, which implies that $l_T^{I,z}(\alpha) = l_T^{I,z}(\alpha')$. This proves constancy on U and that trace changes can occur only at $\alpha \in \mathcal{C}(I)$. □

Because Algorithm 1 uses a non-strict inequality in (12), the value of $l_T^{I,z}(\alpha)$ at $\alpha \in \mathcal{C}(I)$ can differ from the constant values on the adjacent open intervals. Therefore, any exact ERM procedure must treat critical points as candidates.

Exact ERM over $\alpha \in (0, 1]$.

Given i.i.d. samples $(I_1, z_1), \ldots, (I_N, z_N) \sim (D')^N$, define the empirical objective

$$\widehat{L}_N(\alpha) \;:=\; \frac{1}{N}\sum_{j=1}^{N} l_T^{I_j, z_j}(\alpha), \qquad \alpha \in (0, 1].$$

We show how to compute an exact minimizer $\widehat{\alpha} \in \arg\min_{\alpha\in(0,1]} \widehat{L}_N(\alpha)$. For each sample j, form $\mathcal{C}(I_j)$ and sort its distinct elements as

$$0 < c_{j,1} < \cdots < c_{j,q_j} \leq 1, \qquad q_j \leq kT.$$

Define the open intervals induced by these breakpoints:

$$U_{j,0} := (0, c_{j,1}), \quad U_{j,r} := (c_{j,r}, c_{j,r+1})\ (r = 1, \ldots, q_j - 1), \quad U_{j,q_j} := (c_{j,q_j}, 1),$$

(with the convention that if $q_j = 0$ then $U_{j,0} = (0, 1)$). By Lemma C.6, we know that $l_T^{I_j,z_j}(\alpha)$ is constant on each $U_{j,r}$, but may take a different value at each $\alpha = c_{j,r}$. Define the global critical set

$$\mathcal{C}_{\text{all}} \;:=\; \bigcup_{j=1}^{N} \mathcal{C}(I_j), \qquad |\mathcal{C}_{\text{all}}| \leq NkT,$$

and let its distinct elements be $0 < b_1 < \cdots < b_M < 1$ (possibly $M = 0$).

Theorem C.7 (Running time for implementing exact ERM for PTRR$_\alpha$ over the derandomized loss function)**.** *Given offline access to full reward curves for each training instance, an exact empirical minimizer* $\widehat{\alpha} \in \arg\min_{\alpha\in(0,1]} \widehat{L}_N(\alpha)$ *can be computed in time*

$$O\big(NkT^2 \;+\; NkT\log(NkT)\big)\,.$$

Proof. We first show that it suffices to minimize $\widehat{L}_N(\alpha)$ over a finite candidate set. By Lemma C.6, for each j, the function $l_T^{I_j,z_j}(\alpha)$ is constant on each open interval of $(0,1]\setminus\mathcal{C}(I_j)$. Therefore the empirical average $\widehat{L}_N(\alpha)$ is constant on each open interval of $(0,1]\setminus\mathcal{C}_{\text{all}}$. The only points at which $\widehat{L}_N(\alpha)$ can differ from these open-interval constants are the breakpoints in $\mathcal{C}_{\text{all}}$ and the endpoint $\alpha=1$, which implies that an exact minimizer is attained by some α in the finite set

$$\mathcal{A}_{\text{cand}} \;:=\; \Big(\{b_1,\ldots,b_M,1\}\Big) \;\cup\; \Big\{\text{one representative } \tilde{\alpha}_r\in(b_r,b_{r+1}):\; r=0,\ldots,M\Big\}$$

(where we set $b_0:=0$ and $b_{M+1}:=1$).

We now describe an evaluation procedure for $\mathcal{A}_{\text{cand}}$ and bound its runtime. For each sample j, we compute (i) the constant value of $l_T^{I_j,z_j}(\alpha)$ on each local interval $U_{j,r}$ by simulating PTRR$_\alpha$ at one representative point in $U_{j,r}$, and (ii) the value at each local breakpoint $c_{j,r}$ by simulating at $\alpha=c_{j,r}$, and also simulate once at $\alpha=1$. Each simulation performs at most T pulls and therefore runs in $O(T)$ time. Since each sample has at most $q_j\le kT$ breakpoints and $q_j+1\le kT+1$ open intervals, this requires $O((2q_j+2)\cdot T)=O(kT^2)$ time per sample and $O(NkT^2)$ time total.

Next, we sort the global breakpoint set $\mathcal{C}_{\text{all}}$ in $O(NkT\log(NkT))$ time. Sweep α from left to right across the sorted list. On each global open interval (b_r,b_{r+1}), compute $\widehat{L}_N(\tilde{\alpha}_r)$ using the current local-interval value for each sample. At each breakpoint b_r, compute $\widehat{L}_N(b_r)$ by using the precomputed breakpoint value for those samples that have a local breakpoint at b_r, and otherwise using the current local-interval value. Track the best value encountered across all candidates in $\mathcal{A}_{\text{cand}}$, including $\alpha=1$. This yields an exact minimizer over $(0,1]$.

The sweep itself is linear in the number of breakpoint events, $O(NkT)$, and is dominated by the $O(NkT^2)$ simulation cost and the $O(NkT\log(NkT))$ sorting cost. □

This is a purely computational result, which shows that the ERM minimizer assumed in Appendix B.2 can be computed exactly over the continuous parameter space $\alpha\in(0,1]$ from offline learning curves in time

$$O\big(NkT^2 \;+\; NkT\log(NkT)\big)\,.$$

We can use this result for implementing exact ERM over the derandomized dual loss function to implement exact ERM over the true dual loss function (which involves expectation over the randomization of PTRR$_\alpha$) by taking an average over $k!$ permutations of the arms.

C.6 AN ALGORITHM FOR FINDING A SUITABLE VALUE OF THE PARAMETER

In this section, we provide a natural algorithm for finding a suitable value of the parameter α for the PTRR algorithm based on access to instances. We then analyze the sample complexity, runtime complexity, and performance of the algorithm.

At a high level, the algorithm starts by (approximately) identifying the underlying value of the CEE, β_i for each instance i in the training set. Then, we aggregate estimated values $\hat{\beta}_i$ by taking the maximum and that aggregated value (or 1, if it is bigger than 1) is returned as a value of α to use in future deployments. We now present a very general version of the result.

Algorithm 3 Approximate Learner

Require: n samples from distribution $\mathcal{D}$
1: **for** each instance I_i **do**
2: $\quad\hat{\beta}_i \leftarrow$ `CEE_Oracle`(I_i)
3: $\beta \leftarrow \max_i \hat{\beta}_i$
4: **return** β

For both steps, we can adjust the sophistication of the protocols. For the first step, we start by assuming we have an oracle that returns $CEE(I)$ for any instance I. We can implement an approximate oracle by dividing the interval $(0,1]$ into

subintervals of size ϵ and binary searching over that discrete set for the smallest satisfying value. Similarly, for the second step, it is easiest to analyze the aggregation protocol that simply takes the maximum. However, this has the awkward property that with more samples, the predicted parameter increases toward the supremum of the support of the induced distribution over βs. Instead, we ought to use a $1-\delta$ order statistic that concentrates around its mean. This implies discarding some probability mass as part of the failure probability.

At a high level, in order to argue that this algorithm provides interesting, non-vacuous guarantees on the performance of PTRR with the learned parameter on a new instance drawn from the same distribution, we argue that the learned value of the parameter is good for a good fraction of the support of the distribution. To argue about this, we first define the induced distribution over β:

Definition C.8. *For each instance I_i drawn from the distribution $\mathcal{D}$, suppose we compute $\beta_i = CEE(I)$. We call the distribution over the β_i the* induced distribution over βs.

Next, we define the optimal parameter $\alpha^\star$ for the distribution.

Definition C.9. *Define $\alpha^\star$ as the maximizer of the expected competitive ratio over the distribution of instances. In particular, $\alpha^\star := \arg\max_{\alpha\in(0,1]} \mathbb{E}_{I_{test}\sim\mathcal{D}}\left[\frac{Rew(PTRR_\alpha, I_{test})}{OPT(I_{test})}\right]$.*

Now, assuming the induced distribution over β for a distribution $\mathcal{D}$ has some density and cumulative density, we can show that, for a fixed number of samples, with quantifiable probability (approaching 1 as the number of samples goes to infinity), we can bound the reward ratio in terms of the learned parameter. We formalize this in the theorem below.

Theorem C.10. *Suppose the induced distribution over βs has density $f(\beta)$ and cumulative density $F(\beta)$. Suppose we run Algorithm 3 with a fixed number of samples $n > 10$ and receive $\hat{\alpha}$ as the learned value of the parameter. If $\hat{\alpha} \geq 1$, we run $PTRR_1$ and achieve $1/\sqrt{k}$ fraction of the reward. Otherwise, with probability $\left(\frac{1}{n+1}\right)^{1/n}\left(\frac{n}{n+1}\right)$ (which approaches 1 as $n \to \infty$) over the test sample and the training samples, the reward of running $PTRR_{\hat{\alpha}}$ on the test sample is at least $1/k^{\hat{\alpha}/(\hat{\alpha}+1)}$ fraction of running $PTRR_{\alpha^\star}$ on that instance. That is:*

$$\frac{Rew(PTRR_{\hat{\alpha}}, I_{test})}{Rew(PTRR_{\alpha^\star}, I_{test})} \geq \Omega\left(\frac{1}{k^{\hat{\alpha}/(\hat{\alpha}+1)}}\right). \tag{14}$$

Remark C.11. *Note that this is different from the PAC guarantee given by ERM in two important ways. First, we cannot choose arbitrarily small failure probability and error values and draw a number of samples that is a function of those. Secondly, we are providing a guarantee for a single fixed test instance drawn from the distribution, not for the "average" test instance. In particular, we are using the fact that PTRR with the best value of the parameter for the distribution cannot get more reward on the instance than the optimal reward achievable on that instance.*

Proof. In order to prove this result, at a high-level, we will show that even if we resign ourselves to failure on a small fraction of the distribution (i.e., bad approximation ratio), we can still achieve a good approximation with high probability on the rest. To show this, we proceed in three steps:

1. First, we define two events that are crucial to our guarantee: first, that the learned $\hat{\alpha}$ is sufficiently large; second, that the test example is sufficiently nice.
2. We argue that for any test example for which the CEE is at most τ, running $PTRR_{\hat{\alpha}}$ obeys the reward ratio in Eqn. 14 as long as $\hat{\alpha} \geq \tau$.
3. We argue that with good probability over the training sample, $\hat{\alpha} \geq \tau$, and with good probability over the test instance, $CEE(I_{\text{test}}) \leq \tau$. Thus, we can combine the probabilities of the relevant events to show that with the stated probability, the reward ratio guarantee holds.

Relevant Events . At a high level, our strategy will be to resign ourselves to failure on a small fraction of the support of the induced distribution over βs in the interest of guaranteeing an approximation on the rest of the support. To that end, we define the following two events:

$$\mathcal{E}_1 := \{\hat{\alpha} \geq \tau\} \tag{15}$$

$$\mathcal{E}_2 := \{CEE(I_{\text{test}}) \leq \tau\} \tag{16}$$

We know from the analysis in Section 3 that for an instance with CEE β, running $PTRR_\alpha$ for any $\alpha \geq \beta$ will achieve $1/k^{\alpha/(\alpha+1)}$ fraction of the optimal reward. Thus, on a new test example, provided the value of the PTRR parameter we use is larger than the CEE of the instance, we will accrue sufficient reward. We formalize this subsequently.

Reward Ratio Holds . Now, recall from Theorem 3.5 that for an instance with CEE β_I, $PTRR_\alpha$ for any $\alpha > \beta_I$ accrues $O(OPT/k^{\alpha/(\alpha+1)})$ reward. Thus, if $\hat{\alpha} > \tau > \beta_{\text{test}} \coloneqq CEE(I_{\text{test}})$, the $Rew(PTRR_{\hat{\alpha}}, I_{\text{test}}) \geq O(OPT/k^{\alpha/(\alpha+1)})$. Finally, we observe that by the definition of OPT, we have that $OPT \geq Rew(PTRR_\alpha, I_{\text{test}}) \, \forall \alpha$, so in particular this holds for $\alpha = \alpha^\star$, the value of α for which the expected competitive ratio is maximized. Thus, we have that if $\hat{\alpha} \geq \tau$ and $CEE(I_{\text{test}}) \leq \tau$, then:

$$\frac{Rew(PTRR_{\hat{\alpha}}, I_{\text{test}})}{Rew(PTRR_{\alpha^\star}, I_{\text{test}})} \geq O\left(\frac{1}{k^{\hat{\alpha}/(\hat{\alpha}+1)}}\right).$$

Thus, it remains to argue that the conditions in the previous statement occur with good probability.

Probability . First, let us consider event $\mathbb{P}[\mathcal{E}_2]$. This is simply the cumulative density at $\beta = \tau$, $F(\tau)$. Next, we analyze $\mathbb{P}[\mathcal{E}_1]$:

$$\begin{aligned}
\mathbb{P}[\mathcal{E}_1] &= \mathbb{P}[\hat{\alpha} \geq \tau] && (17)\\
&= \mathbb{P}\left[\max_i \hat{\beta}_i \geq \tau\right] && (18)\\
\text{under oracle } &= 1 - \mathbb{P}[\forall\, i\, \beta_i \leq \tau] && (19)\\
&= 1 - F(\tau)^n. && (20)
\end{aligned}$$

The probability of success is the probability of the intersection of these two events, so the probability of success is $F(\tau)(1-F(\tau)^n)$. Now, it suffices to show that we can pick a τ that gives us a good probability of success. We can therefore pick the τ that maximizes the probability of success, namely $\arg\max_\tau F(\tau)(1-F(\tau)^n)$.

$$\begin{aligned}
\max_\tau F(\tau)(1-F(\tau)^n) \Leftrightarrow f(\tau) - (n+1)F(\tau)^n f(\tau) &= 0 && (21)\\
f(\tau)\left(1-(n+1)F(\tau)^n\right) &= 0 && (22)\\
\frac{1}{n+1} &= F(\tau)^n && (23)\\
\tau &= F^{-1}\left(\left(\frac{1}{n+1}\right)^{1/n}\right). && (24)
\end{aligned}$$

Plugging this back into the probability of success, we get the stated result.

□

C.7 FULL EMPIRICAL EVALUATION

The main phenomenon highlighted by the theory is that the parameter α controls the exploration–abandonment tradeoff, and therefore that the best α can depend on the underlying instance. The purpose of this empirical section is to provide evidence on real learning-curve data that different instances prefer different α. We use sample-wise learning curves from the Learning Curve Database (LCDB 1.1), specifically the CC-18 benchmarks. LCDB stores error-rate curves at multiple training-set sizes ("anchors") for a fixed set of learners, together with repeated evaluations under a nested cross-validation protocol.

Each dataset d defines one improving-bandit instance. Arms correspond to learners, and the time index corresponds to LCDB anchor points. We convert error rates to rewards through $r_{i,d}(t) = 1 - \text{err}_{i,d}(t)$, so that larger rewards correspond to better performance. LCDB stores multiple repetitions per (dataset, learner, anchor) due to nested cross-validation. We average across these repetitions (ignoring missing values) to obtain a single mean learning curve per (dataset, learner). This yields deterministic reward functions $r_{i,d}(\cdot)$, matching our setting.

LCDB curves can terminate early for some dataset–learner pairs (in the stored tensors this appears as NaNs). For a fixed horizon T, we restrict to datasets for which all included learners have finite values for anchors $t = 1, \ldots, T$. This ensures that each dataset corresponds to a well-defined bandit instance with a common horizon.

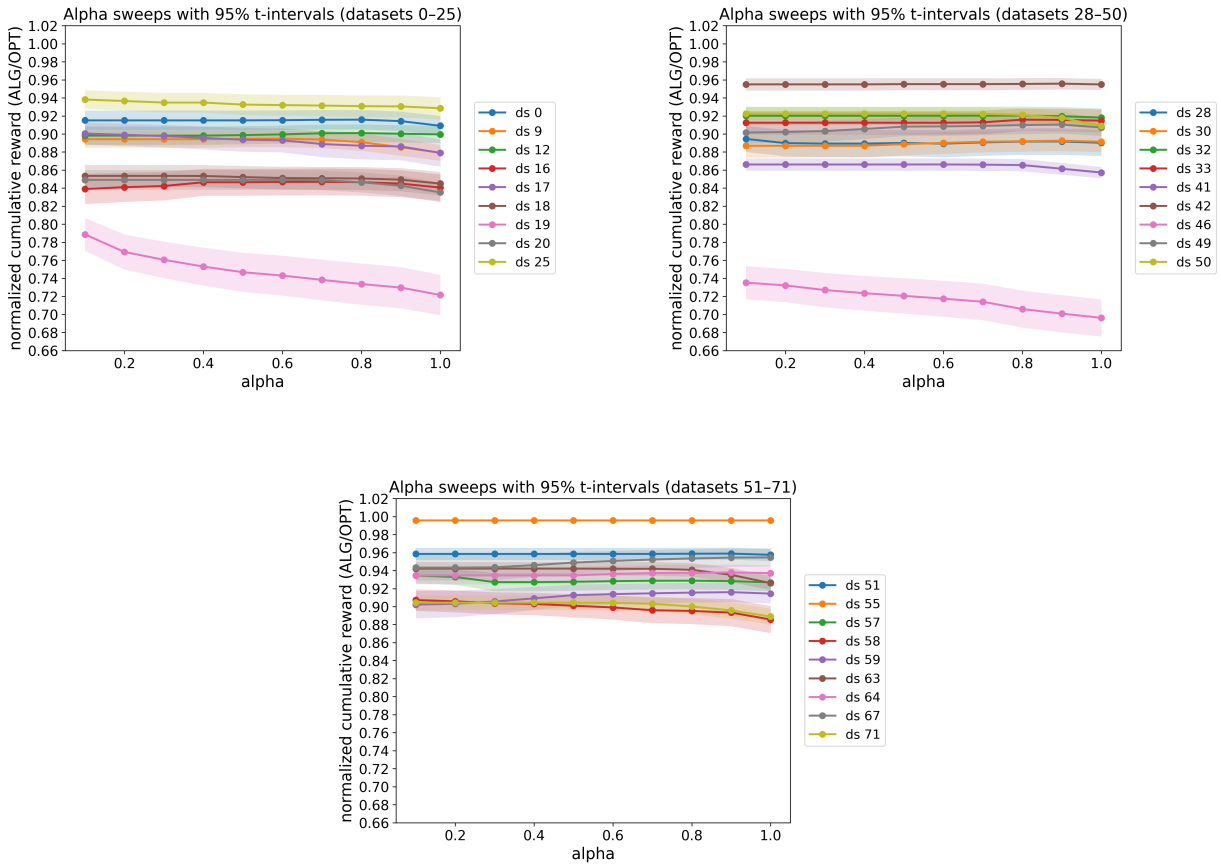

Figure 3: **Sensitivity of PTRR_α to α on all LCDB instances ($T = 44$, $k = 22$).** Each curve corresponds to one CC-18 dataset d from LCDB 1.1 and reports the normalized cumulative reward $\mathbb{E}[\mathrm{PTRR}_\alpha(d)]/\mathrm{OPT}(d)$ as a function of $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$, where $\mathrm{OPT}(d) = \max_i \sum_{t=1}^{T} r_{i,d}(t)$ is the cumulative reward of the best fixed-arm policy in hindsight under the same horizon (which does not necessarily correspond to the global optimal policy due to the absence of monotonicity) and $r_{i,d}(t) = 1 - \mathrm{err}_{i,d}(t)$ is the mean (over cross-validation) reward at anchor t for arm i. For each (d, α), $\mathbb{E}[\mathrm{PTRR}_\alpha(d)]$ is estimated by averaging over 200 random arm orderings. The shaded regions are pointwise 95% Student-t confidence intervals across the 200 runs (mean $\pm\, t_{0.975,199} \cdot \mathrm{sd}/\sqrt{200}$). For most datasets, performance differences across α are small relative to the confidence intervals, while a minority show a significant trend across α on this grid.

For each usable dataset d and each α on a fixed grid, we run PTRR_α for a horizon of T pulls. The only randomness in our implementation is the random ordering in which arms are first considered. We average results over 200 random seeds. We evaluate performance using the normalized cumulative reward

$$\frac{\mathbb{E}[\mathrm{ALG}(d;\alpha)]}{\mathrm{OPT}(d)} \qquad \text{where} \qquad \mathrm{OPT}(d) = \max_i \sum_{t=1}^{T} r_{i,d}(t),$$

which is the reciprocal of the competitive ratio $\mathrm{OPT}/\mathbb{E}[\mathrm{ALG}]$ from Definition 2.2. Note that $\mathrm{OPT}(d)$, the best fixed-arm policy on d in hindsight, is not necessarily the global best policy here due to the absence of monotonicity.

Main experiment: $T = 44$, $k = 22$. Our primary experiment uses horizon $T = 44$ and a learner set of size $k = 22$ (we exclude two learners with $\geq 50\%$ ill-behavior in the CC-18 file). Under this choice, we obtain 27 datasets with complete prefixes of length T across all k learners.

We sweep $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$. Figure 3 shows all of the per-dataset performance curves $\alpha \mapsto \mathbb{E}[\mathrm{ALG}]/\mathrm{OPT}$, demonstrating that the maximizer varies across certain instances. Across the 27 datasets, the best α is widely distributed:

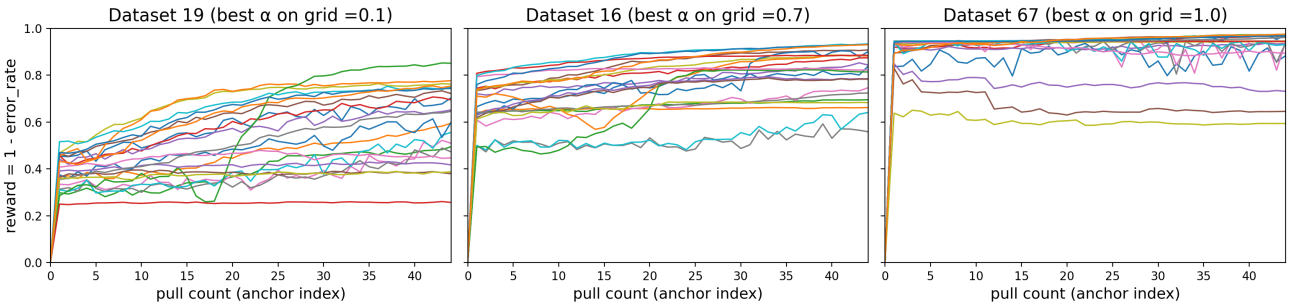

Figure 4: **Mean LCDB reward curves for three datasets with distinct best α values on the grid.** Each panel overlays the mean reward curves $r_{i,d}(t) = 1 - \mathrm{err}_{i,d}(t)$ across anchors t for all $k = 22$ arms on a single CC-18 dataset d. The title of each panel reports the value of $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$ that maximizes the estimated normalized cumulative reward $\mathbb{E}[\mathrm{PTRR}_\alpha(d)]/\mathrm{OPT}(d)$ at horizon $T = 44$ on that dataset. We selected these datasets to illustrate the reward dynamics underlying distinct best α values on the grid.

11 datasets select $\alpha = 0.1$, while many others select α in the range $[0.8, 1.0]$ (it is worth noting that 4 of the 11 cases are actually flat across several small α values). Mechanistically, smaller α corresponds to more aggressive early abandonment in PTRR_α, so these datasets are those for which aggressive abandonment is (on this grid) empirically most favorable.

Naturally, real learning curves in LCDB are not guaranteed to satisfy the monotonicity/concavity assumptions used in this work (and largely don't). To connect α to concrete learning dynamics, we plot mean learning curves with fixed axes for three datasets with distinct best α's. Figure 4 displays all learners for dataset 19 (which selects $\alpha = 0.1$), dataset 16 (which selects $\alpha = 0.7$) and dataset 67 (which selects $\alpha = 1.0$). Qualitatively, these plots suggest a mechanism consistent with the influence of α on PTRR_α, where datasets preferring smaller α tend to exhibit early separation between 'good' and 'bad' arms (making aggressive abandonment beneficial), while datasets preferring larger α exhibit closer early performance among many learners (making aggressive abandonment riskier). We emphasize that this interpretation is qualitative and based directly on the plotted mean curves; the purpose of these plots is merely to contextualize the observed instance dependence.

D BEST-OF-BOTH-WORLDS FOR MAXIMIZING CUMULATIVE REWARD

In this section, we consider the problem of getting the best policy regret possible if it is sublinear and reverting to optimal competitive ratio if it is not. In the main body, we considered a similar formulation for best-arm identification. Here, we instead hybridize between an algorithm known to get good policy regret on good instances from [Met+22] and an algorithm known to get optimal competitive ratio on worst-case instances from [BR25]. We take a data-driven approach to identifying exactly how to hybridize, i.e., exactly when to switch from the policy-regret-optimizing algorithm to the competitive-ratio-optimizing algorithm. We note that as a result of this approach, we do *not* get per-instance worst-case guarantees. This is an interesting direction for future work.

D.1 MOTIVATING EXAMPLES

[BR25] give a randomized algorithm for improving bandits (also known as deterministic rested rising multi-armed bandits) that achieves the optimal $\sqrt{k}$ competitive ratio on worst-case instances. Although it is well known that a sublinear regret is not attainable for worst-case instances of improving bandits, under some niceness assumptions, UCB-style algorithms [Met+22] can be shown to obtain sublinear policy regret (regret w.r.t. the best single arm pulled for the entire time horizon). Here we present examples that show that both the above algorithms from the literature may be sub-optimal and there are instances where each is dominated by the other.

Example D.1. *The randomized robin algorithm of [BR25] may suffer linear policy regret on instances where the UCB-based algorithm of [Met+22] achieves sublinear regret. We set the reward function for the best arm as $f_{i^*}(t) = 1$ for all t, and for any other arm $i \neq i^*$ as $f_i(t) = t/T$, where T is the time horizon. Now the randomized robin algorithm selects the optimal arm first with probability $1/k$, and otherwise, it keeps playing a sub-optimal arm. The expected total reward is $T \cdot \frac{1}{k} + \frac{T}{2} \cdot \frac{k-1}{k} = \frac{T}{2}\left(1 + \frac{1}{k}\right)$, which corresponds to $\Omega(T)$ regret. On the other hand, the UCB-based algorithm has a logarithmic upper bound on its regret. Indeed, after $O(\log T)$ exploratory pulls of any sub-optimal arm, it will always prefer*

the optimal arm, implying an upper bound of $O(k \log T)$ on the policy regret.

Example D.2. *The UCB-based algorithm of [Met+22] may have $\Omega(k)$ competitive ratio on some instances. We use the example used in the lower bound construction of [BR25]. Set $f_{i^*}(t) = t/T$ for all t for the optimal arm i^*, and for any other arm $i \neq i^*$ as $f_i(t) = \min\left\{t/T, \frac{1}{\sqrt{k}}\right\}$. Due to the exploration term, while the arm rewards are identical, each arm gets pulled an equal number of times. By time T, each arm gets pulled T/k times for a total reward of $T/2k$, sub-optimal by a factor of k.*

D.2 DATA-DRIVEN HYBRID APPROACH

Our goal, therefore, is to take a step toward devising algorithms that can achieve sublinear regret for nice instances but fall back to optimal competitive ratio for general instances. To this end, we describe an algorithm that interpolates between a UCB-based algorithm by [Met+22] that gets sublinear regret on nice instances and the algorithm of [BR25] which achieves the optimal worst-case competitive ratio.

Algorithm 4 Regret-optimizing Hybrid Algorithm, i.e. $\texttt{Regret-Hybrid}_B$

Run Algorithm 1 (R-ed-UCB) of [Met+22] for B time steps
Run PTRR for T-B time steps

We define a family of algorithms parameterized by B:

Definition D.3. *Define the family of algorithms $\texttt{Regret-Hybrid} \coloneqq \{\texttt{Regret-Hybrid}_B : B \in [T]\}$, where $\texttt{Regret-Hybrid}_B$ is Algorithm 4.*

Now, as before, we analyze Q_D and then instantiate the result for a loss function of interest.

Lemma D.4. *For the family $\texttt{Regret-Hybrid}$ defined in Defn. D.3, the improving multi-armed bandits problem, and any piecewise constant loss function, $Q_D \leq k\,T^2$.*

Proof. B takes on T discrete values, and for each fixed B , by Lemma 3.8, there are at most $k\,T$ behaviors for the PTRR family on an instance. □

In order to study cumulative reward via regret, we can use the average regret loss function defined in Defn. C.4. Then, extending Theorem E.8, we get the following corollary:

Corollary D.5. *For the Hyperparameter Transfer setting for the improving multi-armed bandits problem optimizing for averaged regret over the algorithm family described in Defn. 4.3, $N = O\left(\left(\frac{m}{\epsilon}\right)^2 (\log kT + \log \frac{1}{\delta})\right)$ instances drawn from $\mathcal{D}$ suffice to get the uniform convergence guarantee in Theorem C.2.*

Note that this result provides an algorithm that is competitive with the best possible algorithm in the family on instances from distribution $\mathcal{D}$. Thus, that algorithm may not provide the worst-case fallback guarantee on a *fixed* instance. As mentioned earlier, this is an interesting consideration for future work.

E BAI COMPARISON WITH PRIOR WORK AND PROOF DETAILS

We provide below a comparison with prior work on best arm identification along with complete proof details for Section 4.

E.1 COMPARISON WITH PRIOR WORK

First, let us compare the BAI task in [Mus+24] to ours. In their work, they study the task of identifying the arm with the greatest single pull in the instance (this equals the pull of that arm at time T due to reward monotonicity). In our work, we study the task of identifying the arm with the greatest cumulative reward in the instance. While these tasks are slightly different, we note that both are important and relevant tasks, and the best arm for one task will be a 2-approximation (that is,

not be more than a factor of two suboptimal) for the other task. On the one hand, consider a setting in which each arm is a technology, and investing in developing the technology increases its utility. If we are interested in identifying one out of many technologies that will have the highest utility after the investment period, we are interested in identifying the arm with the highest single pull, corresponding to [Mus+24]'s setting. On the other hand, consider a setting in which each arm is an advertisement, and the reward associated with a pull is the amount of money a user spends as a result of that advertisement. A corporation would care to identify the arm that has the highest cumulative reward and then use that arm for future users. Thus, both are interesting and valid goals albeit slightly different from each other.

Having defined the tasks, we will now see more clearly that the difference in the "niceness" conditions relates to this, as well. We consider the deterministic variant of the setting which is the focus of this work (that is, set the stochasticity in the instances to zero). Now, we study the conditions and terms in Theorem 4.1 of [Mus+24]. Note that setting the stochasticity to zero implies $\epsilon, \sigma = 0$. Then, y is no longer a function of a , and thus Eqn. 4 is satisfied for *all* a , so we can achieve BAI with probability 1 if the conditions are met. This is the same guarantee we achieve, and so we are ready to compare the conditions more carefully.

In [Mus+24], they study y_i , the number of pulls of arm i until the first difference is smaller than the average gap between the best single pull over all arms and the best single pull of this arm. This quantity corresponds to our quantity $h_i(\epsilon)$, which is the number of pulls of arm i after which the amount of cumulative reward we could achieve if we continued growing with the same slope is bounded by ϵ. Thus, both y_i and h_i study how quickly arms "converge" to their final value, y_i directly in the value (of individual pull) sense and h_i in the cumulative sense.

Finally, we note that [Mus+24] compares the sum of these "convergence times" to $(1-\iota)T$, whereas we compare to a parameter B . In both cases, the smaller the comparison value, the more value we can extract from the best arm in the remaining ιT or $T - B$ time. Thus, the overall flavor of the conditions in both works is the same, namely that if arms "converge quickly," then BAI is possible. The exact notions of the words in quotes are what differ.

E.2 DIFFICULTY IN CORRALLING IMPROVING BANDIT ALGORITHMS

A natural approach towards obtaining best-of-both-worlds guarantees in the above examples would be to design a meta-algorithm that can potentially switch between [BR25] and [Mus+24]. We examine whether it is possible in the improving bandits setting to perform corralling [Aga+17; AMM21; Luo+22], a recently developed paradigm for meta-learning bandit algorithms in a fully online setting. Our main result here is an impossibility result which rules out the ability to use corralling to achieve best-of-both-worlds in a fully online improving bandits setting. We show that in the improving bandits setting, no matter what the meta-algorithm is, it is possible to suffer linear regret relative to the best base algorithm.

In the meta-learning setting, we have a collection of M *base* algorithms $\mathcal{B} = \{B_1, \ldots, B_M\}$ for the IMAB problem and the goal is to (nearly) recover the guarantees of the best algorithm on any given instance. Each base algorithm can give its own prediction for which arm to play next, given the history of arm pulls and rewards so far and the meta-learner can decide which arm to actually pull based on these predictions. For cumulative reward maximization, we seek to bound the meta-regret with respect to the best algorithm among the base algorithms

$$\tilde{R}_T(\mathcal{M}, \mu, \mathcal{B}) := \max_j R_T^{B_j}(\mu) - R_T^{\mathcal{M}}(\mu),$$

where μ denotes the IMAB instance, $R_T^{B_j}$ denotes the cumulative reward if algorithm B_j is used exclusively to decide the arm pulls for T rounds, and $R_T^{\mathcal{M}}$ denotes the cumulative reward collected by the corralling meta-algorithm $\mathcal{M}$.

Theorem E.1 (Minimax Impossibility of Corralling Improving Bandits)**.** *Consider deterministic two-arm rested bandits with rising concave reward sequences and horizon T. Let $\mathcal{B} = \{B_1, \ldots, B_M\}$ be any finite class of (possibly randomized) base algorithms. A meta-algorithm $\mathcal{M}$ selects at each round one base algorithm whose recommended arm is executed. Then there exists a universal constant $c > 0$ such that*

$$\inf_{\mathcal{M}} \sup_{\mu, \mathcal{B}} \tilde{R}_T(\mathcal{M}, \mu, \mathcal{B}) \;\geq\; cT.$$

Proof Sketch. We construct two environments that are identical for the first $T/2$ pulls of each arm but diverge thereafter, with opposite optimal arms. Any meta-algorithm attempting to compete with the best run-alone base must effectively determine

which environment it faces before the divergence point. However, since the two instances are indistinguishable during the prefix, identifying the correct environment requires linear exploration. Consequently, linear meta-regret is unavoidable on at least one of the two instances. A full proof is located in Appendix E.2. □

This theorem establishes a minimax barrier for corralling in improving bandits. The obstruction is information-theoretic: the two environments are indistinguishable for a linear prefix yet demand opposite commitments thereafter. This motivates the need to consider alternative approaches towards achieving best-of-both-worlds guarantees.

We provide below a complete proof for Theorem E.1.

Proof. Fix horizon T and let $L = T/2$. Choose $\Delta > 0$ sufficiently small. Construct two deterministic concave rising environments E^+ and E^- as follows.

For $s \leq L$, define

$$\mu_1(s) = \mu_2(s) = s\Delta.$$

That is, the first L pulls of either arm yield identical rewards in both environments.

For $s > L$, define:

Environment E^+:

$$\mu_1(s) = L\Delta, \qquad \mu_2(s) = L\Delta + (s - L)\Delta.$$

Environment E^-:

$$\mu_2(s) = L\Delta, \qquad \mu_1(s) = L\Delta + (s - L)\Delta.$$

Both reward sequences are nondecreasing and concave. Observe that the two environments are identical until some arm is pulled more than L times. In E^+, arm 2 is uniquely optimal after the divergence; in E^-, arm 1 is uniquely optimal. Set the base algorithm class $\mathcal{B} = \{B_1, B_2\}$, where B_j always pulls arm j.

Consider any meta-algorithm $\mathcal{M}$. Since rewards are identical for the first L pulls of each arm, the distribution over the first L actions of $\mathcal{M}$ is identical under E^+ and E^-. Let N_1 be the expected number of pulls of arm 1 by round L. Without loss of generality, suppose $N_1 \leq L/2$ (otherwise swap the roles of the arms).

Consider environment E^-, where arm 1 becomes uniquely optimal after the divergence. Let j^* be an index maximizing $R_T^{B_j}(E^-)$. Base algorithm B_1 must obtain $\Theta(T\Delta)$ additional reward by exploiting arm 1 after round L.

Because $\mathcal{M}$ allocates at most $L/2$ pulls to arm 1 during the prefix, it under-invests in the arm that later becomes uniquely optimal. Due to concavity, the cumulative post-divergence reward is linear in the number of additional pulls. Thus there exists a constant $c > 0$ such that

$$R_T^{B_1}(E^-) - R_T^{\mathcal{M}}(E^-) \geq cT.$$

If instead $N_1 > L/2$, the same argument applied to E^+ yields an identical bound. Therefore,

$$\sup_{\mu \in \{E^+, E^-\}} \tilde{R}_T(\mathcal{M}, \mu, \mathcal{B}) \geq cT.$$

□

E.3 PROOFS FOR RESULTS IN SECTION 4.2

We prove the lemma used in Section 4.2 to justify the quantities L_i, U_i used in Algorithm 2.

Lemma E.2. *For every arm i and every $t \leq T$, we have $L_i(t) \leq f_i(T) \leq U_i(t)$.*

Proof. By concavity, we know that $\gamma_i(s)$ is non-increasing, and therefore that for any $x \geq t$, we have

$$f_i(x) - f_i(t) = \sum_{s=t}^{x-1} (f_i(s+1) - f_i(s)) \leq \sum_{s=t}^{x-1} \gamma_i(t) = (x-t)\gamma_i(t).$$

Setting $x = T$ gives $f_i(T) \leq f_i(t) + (T-t)\gamma_i(t) = U_i(t)$. By monotonicity, we likewise know that $L_i(t) = f_i(t) \leq F_i(T)$. □

We include below a complete proof for Theorem 4.5.

Proof (of Theorem 4.5.)

Proof of 1. (certificate correctness). Suppose an instance I satisfies $\mathrm{GCC}(\theta)$. Since $\triangle_i(t)$ is non-increasing for all i and $\sum_{i=1}^{k} h_i(\Delta_I/3) \leq \theta$, we know that there are at most θ total pulls (across arms) such that $\triangle_i(t_i) > \Delta_I/3$. Note that we can never have $L_i(t_i) > \max_{j \neq i} U_j(t_j)$ for some $i \neq i^*$, as we know by concavity and monotonicity that $L_i(t) \leq f_i(T) \leq U_i(t)$, so this would imply $f_{i^*}(T) \leq U_{i^*}(t_{i^*}) < L_i(t_i) \leq f_i(T)$ (a contradiction). Since $B > \theta$ and Stage 1 always pulls the arm i that maximizes $(U_i - L_i)$, it follows that the algorithm will reach some point (namely $t = \theta$) where $\triangle_i(t_i) \leq \Delta_I/3$ for all i. At this point, for each $j \neq i^\star$, we have

$$U_j(t_j) \leq f_j(T) + \frac{\Delta_I}{3} \leq f^*(T) - \Delta_I + \frac{\Delta_I}{3} = f^*(T) - \frac{2\Delta_I}{3}.$$

Moreover, i^* satisfies

$$L_{i^\star}(t_{i^*}) \geq f^*(T) - \frac{\Delta_I}{3}.$$

It follows that $L_{i^\star}(t_{i^*}) > \max_{j \neq i^\star} U_j(t_j)$, and therefore that the algorithm returns i^* in Stage 1. □

Proof of 2. (approximation fallback). Write $g^\star(h) := \max_i g_i(h) = \max_i f_i(t_i + h)$. Let $T_{\text{rem}} := T - B$, and let $\tau' = T_{\text{rem}} - k$. Suppose $m' := (\tau'/T) f^\star(T)$. Since $g^\star(T) = \max_i f_i(T) \geq f^\star(T)$, we know that

$$m' = \frac{\tau'}{T} f^\star(T) \leq f^\star(T) \left(\frac{\tau'}{T}\right)^\alpha \leq g^\star(T) \left(\frac{\tau'}{T}\right)^\alpha.$$

Using monotonicity and the fact that $g^\star(T) \leq 2f^\star(T)$, we also know that $g^\star(\tau') \leq 2f^\star(T) = 2(T/\tau')m'$, and therefore that $m' \geq (\tau'/(2T))g^\star(\tau')$. Since $B \leq T/2$ and $T \geq 4k$, it follows that

$$\frac{1}{8} g^\star(\tau') \leq m' \leq g^\star(T) \left(\frac{\tau'}{T}\right)^\alpha.$$

Run *PTRR*$_\alpha$ for T_{rem} steps with parameters m' and τ'. By Theorem 3.5, we know that

$$\mathbb{E}\big[\mathrm{ALG}_{T_{\text{rem}}}\big] \;\geq\; \frac{\mathrm{OPT}^{\text{res}}_{T_{\text{rem}}}}{C_\alpha c_2 (k+1)^{\alpha/(1+\alpha)}},$$

where $\mathrm{OPT}^{\text{res}}_{T_{\text{rem}}}$ denotes the optimal cumulative reward for $\{g_i\}$ over T_{rem} rounds and $C_\alpha = 2^{\alpha+2}(\alpha+1)$.

Now note that $\mathrm{OPT}^{\text{res}}_{T_{\text{rem}}} \geq \frac{1}{2} g^\star(T_{\text{rem}}) T_{\text{rem}}$, and that the algorithm's maximum single-pull reward dominates its average reward (Facts B.1 and B.3 in the appendix of [BR25]). Let $\hat{i}$ denote the arm that achieves this maximum single-pull reward, and note that

$$\mathbb{E}\Big[\max_{t \leq T_{\text{rem}}} \text{reward}_t\Big] \;\geq\; \frac{1}{T_{\text{rem}}} \mathbb{E}\big[\mathrm{ALG}_{T_{\text{rem}}}\big] \;\geq\; \frac{g^\star(T_{\text{rem}})}{2C_\alpha c_2 (k+1)^{\alpha/(1+\alpha)}}.$$

Since $f_{\hat{i}}(T) \geq \max_{t \leq T_{\text{rem}}} \text{reward}_t$ by monotonicity, we know that taking expectations gives

$$\mathbb{E}\big[f_{\hat{i}}(T)\big] \;\geq\; \frac{g^\star(T_{\text{rem}})}{2C_\alpha c_2 (k+1)^{\alpha/(1+\alpha)}}.$$

Monotonicity and concavity give $g^\star(T_{\rm rem}) \geq (T_{\rm rem}/T) f^\star(T) \geq \frac{1}{2} f^\star(T)$, as $T_{\rm rem} \geq T/2$. Combining with $c_2 \leq 8$ from Step 1, it follows that

$$\mathbb{E}\big[f_{\hat{i}}(T)\big] \;\geq\; \frac{1}{2C_\alpha c_2 (k+1)^{\alpha/(1+\alpha)}} \cdot \frac{1}{2} f^\star(T) \;\geq\; \frac{1}{2^{\alpha+7}(\alpha+1)} (k+1)^{-\alpha/(1+\alpha)} f^\star(T),$$

as desired. □

E.4 PROOF OF LEMMA E.7

Proof. We argue this by applying the proof of Lemma 3.8. For a fixed (augmented) instance (i.e., fixed instance and fixed random permutation), we are interested in computing the number of different behaviors as we vary B, α . To understand this, let us start by fixing B. Then, by Lemma 3.8, we know that there are at most kT possible behaviors as we vary α . Now, for a fixed value of B, for either algorithm, the sequence of pulls is determined, which exactly determines the loss in Stage 1. Thus, each value of B corresponds to at most 1 new value of the loss. This implies that there are at most kT possible behaviors in both stages for a fixed value of B. Since there are at most T values of B, we have that there are at most kT^2 possible behaviors. □

E.5 CUMULATIVE REWARD BEST ARM IDENTIFICATION

In the below sections, we provide comparable BAI guarantees to Section 4.2 for cumulative reward instead of maximum reward. As before, we work in the standard improving-bandits setting with k arms, a known horizon T, and non-decreasing concave reward functions f_i. Each algorithm Hybrid$_{\alpha,B}$ has two stages. Stage 1 uses a UCB-style envelope: At each step, the algorithm computes a lower bound $L_i(n)$ and terminal upper bound $U_i(n)$ on the final accumulated reward of every arm and pulls the arm with the largest optimistic estimate U_i. If the lower bound of one arm dominates the terminal upper bound of every other arm, Hybrid$_{\alpha,B}$ commits to this arm. If no commit occurs by time B, Stage 2 runs PTRR$_\alpha$ and finds an arm whose expected terminal reward is at least a substantial fraction of the best arm's.

For the terminal envelope, we define $L_i(t) := F_i(t) + (T-t) f_i(t), \triangle_i(t) := \frac{(T-t)(T-t+1)}{2} \gamma_i(t-1)$, and $U_i(t) := L_i(t) + \triangle_i(t)$, where $F_i(T) := \sum_{t=1}^{T} f_i(t)$ and $\gamma_i(t-1) := f_i(t) - f_i(t-1)$. We set $U_i(0) := \infty$ to ensure first pulls. Using concavity and monotonicity, it is again straightforward to prove that $L_i(t) \leq F_i(T) \leq U_i(t)$ for all i, t.

Algorithm 5 *Cumulative Hybrid*$_{\alpha,B}$

Require: m
Stage 1: $t \leftarrow 0$
for each arm i **do**
  $t_i \leftarrow 0, F_i \leftarrow 0, L_i \leftarrow 0, U_i \leftarrow +\infty$
while $t < B$ **do**
  for each i with $t_i \geq 1$ **do**
    $L_i \leftarrow F_i + (T - t_i)\, f_i(t_i)$
    $\gamma_i \leftarrow f_i(t_i) - f_i(t_i - 1)$
    $U_i \leftarrow L_i + \frac{(T-t_i)(T-t_i+1)}{2} \cdot \gamma_i$
  $\hat{i} \leftarrow \arg\max_i L_i, \quad U_{\rm next} \leftarrow \max_{j \neq \hat{i}} U_j$
  if $L_{\hat{i}} > U_{\rm next}$ **then**
    return $\hat{i}$.
  $i' \leftarrow \arg\max_i (U_i - L_i)$
  pull i'; $t_{i'} \leftarrow t_{i'} + 1, t \leftarrow t + 1, F_{i'} \leftarrow F_{i'} + f_{i'}(t_{i'})$
Stage 2: $\tau' \leftarrow (T - B) - k, m' \leftarrow \left(\frac{\tau'}{T}\right) \cdot m$
for each i **do**
  $g_i(s) \leftarrow f_i(t_i + s)$
return $\hat{i} \leftarrow$ arm returned by *PTRR*$_\alpha$ with parameters (m', τ') on $\{g_i\}$ for $T - B$ steps.

E.6 BEST-OF-BOTH-WORLDS BEST ARM IDENTIFICATION GUARANTEES

In this section, we show that *Cumulative Hybrid* contains algorithms that simultaneously (i) guarantee best arm identification on sufficiently benign instances, and (ii) preserve tight (up to constants) multiplicative bounds for approximating the best arm on adversarial instances.[6]

We start by defining a class of 'sufficiently benign' instances and prove that our algorithm is guaranteed to return the best arm on all members of this class. If an instance is not in this class, Stage 2 pursues *approximate* BAI and runs $PTRR_\alpha$ with α dependent on the strength of concavity ($\alpha = 1$ works for *all* instances). Having reverted to an approximate goal, we identify an arm $\hat{i}$ whose final reward satisfies $\mathbb{E}[f_{\hat{i}}(T)] \geq \Omega\left(k^{\frac{-\alpha}{1+\alpha}}\right) f^\star(T)$.

As before, we assume for simplicity that both T and $f^*(T)$ are known to the algorithm, which we use to set $\tau' = (T-B)-k$ and $m' = \frac{\tau'}{T} \cdot f^*(T)$. As in Section 3.2, these assumptions can be removed with only $O(\log k)$ overhead (see also Appendix B.3).

We will now define a condition under which we can guarantee best arm identification.

Definition E.3 (Per-arm terminal budget, h_i)**.** *For any arm i and $\epsilon > 0$, the terminal budget h_i of arm i is defined as $h_i(\epsilon) := \min\{\, n \in \{2, \ldots, T\} : \triangle_i(n) \leq \epsilon \,\}$, where $\triangle_i(t) := \frac{(T-t)(T-t+1)}{2}\, \gamma_i(t-1)$.*

Definition E.4 (Best Arm Gap, Δ_I)**.** *For any instance I, its Best Arm Gap Δ_I is defined as $\Delta_I := OPT_T - \max_{j \neq i^*} F_j(T)$.*

Definition E.5 (Gap Clearance Condition, $GCC(B)$)**.** *For any $B \leq T$, we say that an instance satisfies the Gap Clearance Condition $GCC(B)$ if $\Delta_I > 0$ and $\sum_{i=1}^{K} h_i(\Delta_I/3) \;\leq\; B$.*

Under concavity, $\triangle_i(n)$ denotes the worst-case remaining terminal mass for arm i after n pulls: it is the most reward an adversary can "hide in the tail" by continuing with the last observed slope. The per-arm budget $h_i(\epsilon)$ is therefore the minimal number of pulls needed to ensure its optimistic terminal value is close to current lower envelope. Taking $\epsilon = \Delta_I/3$, once a suboptimal arm has been pulled this many times, its best possible continuation still loses to the best arm's lower envelope. Thus $\mathrm{GCC}(B)$ states that the total work needed to certify the best arm fits within the mid-horizon budget B. If the sum exceeds B, concavity allows at least one suboptimal arm to remain plausibly optimal by time B, so no sound mid-horizon certificate can be guaranteed.

Theorem E.6 (best-of-both-worlds guarantees)**.** *Suppose an instance $I \in \mathcal{I}$ has Concavity Envelope Exponent $\beta_I \in (0, 1]$. Algorithm 5 with $\alpha \in (\beta_I, 1]$ satisfies the following properties:*

1. *If the instance further satisfies $\Delta_I > 0$, and $GCC(\theta)$ holds for $\theta \leq T/2$, then the algorithm with $B \in (\theta, T/2]$ identifies and commits to the best arm $i^\star$ in Stage 1.*
2. *If Stage 1 does not certify a best arm by time B, then Stage 2 finds an approximate best arm $\hat{i}$ such that $\mathbb{E}\big[F_{\hat{i}}(T)\big] \;\geq\; \Omega\Big(k^{\frac{-\alpha}{1+\alpha}}\Big)\; F^\star(T)$, where the expectation is over the randomness of the algorithm.*

Proof Sketch. We argue (1) and (2) separately. (1) $\mathrm{GCC}(\theta)$ implies that the per–arm budgets $h_i(\epsilon)$ sum to at most θ. As long as the certificate fails, Stage 1 pulls an arm with slack $> \Delta_I/3$, so these budgets are met within $B \geq \theta$ pulls. Then every suboptimal arm has $U_j \leq \mathrm{OPT}_T - \frac{2\Delta_I}{3}$ and the best arm has $L_{i^\star} \geq \mathrm{OPT}_T - \frac{\Delta_I}{3}$, which implies that $L_{i^\star} \geq \max_{j \neq i^\star} U_j$. We commit to $i^\star$. (2) If no certificate fires by $B \leq T/2$, running *PTRR*$_\alpha$ for T_{rem} steps yields an average reward within a $k^{\alpha/(1+\alpha)}$ factor of the residual optimum (up to constants). Using the fact that the best single pull is at least the average, and $\mathrm{OPT}^{\mathrm{res}}$ is at least a constant times $g^\star(T_{\mathrm{rem}})\, T_{\mathrm{rem}}$ by concavity, we get $\mathbb{E}[f_{\hat{i}}(T)] \geq \Omega\big(k^{-\alpha/(1+\alpha)}\big)\, f^\star(T)$. □

Proof of 1. (certificate correctness). Suppose an instance I satisfies $\mathrm{GCC}(\theta)$. Since $\triangle_i(t)$ is non-increasing for all i and $\sum_{i=1}^{k} h_i(\Delta_I/3) \leq \theta$, we know that there are at most θ total pulls (across arms) such that $\triangle_i(t_i) > \Delta_I/3$. Note that we can never have $L_i(t_i) > \max_{j \neq i} U_j(t_j)$ for some $i \neq i^*$, as we know that $L_i(t) \leq F_i(T) \leq U_i(t)$, so this would imply $F_{i^*}(T) \leq U_{i^*}(t_{i^*}) < L_i(t_i) \leq F_i(T)$ (a contradiction). Since $B > \theta$ and Stage 1 always pulls the arm i that maximizes $(U_i - L_i)$, it follows that the algorithm will reach some point (namely $t = \theta$) where $\triangle_i(t_i) \leq \Delta_I/3$ for all i. At this point, for each $j \neq i^\star$, we have

$$U_j(t_j) \leq F_j(T) + \frac{\Delta_I}{3} \leq \mathrm{OPT}_T - \Delta_I + \frac{\Delta_I}{3} = \mathrm{OPT}_T - \frac{2\Delta_I}{3}.$$

[6] With additional information about stronger concavity, we achieve sharper bounds by using the corresponding version of PTRR.

Moreover, i^* satisfies

$$L_{i^\star}(t_{i^\star}) \geq F_{i^\star}(T) - \Delta_I/3 = \mathrm{OPT}_T - \Delta_I/3.$$

It follows that $L_{i^\star}(t_{i^\star}) > \max_{j \neq i^\star} U_j(t_j)$, and therefore that the algorithm returns i^* in Stage 1. □

Proof of 2. (approximation fallback). Write $g^\star(h) := \max_i g_i(h) = \max_i f_i(t_i + h)$. Let $T_{\text{rem}} := T - B$, and let $\tau' = T_{\text{rem}} - k$. Suppose $m' := (\tau'/T) f^\star(T)$. Since $g^\star(T) = \max_i f_i(T) \geq f^\star(T)$, we know that

$$m' = \frac{\tau'}{T} f^\star(T) \leq f^\star(T) \left(\frac{\tau'}{T}\right)^\alpha \leq g^\star(T) \left(\frac{\tau'}{T}\right)^\alpha.$$

Using monotonicity and the fact that $g^\star(T) \leq 2f^\star(T)$, we also know that $g^\star(\tau') \leq 2f^\star(T) = 2(T/\tau')m'$, and therefore that $m' \geq (\tau'/(2T))g^\star(\tau')$. Since $B \leq T/2$ and $T \geq 4k$, it follows that

$$\frac{1}{8} g^\star(\tau') \leq m' \leq g^\star(T) \left(\frac{\tau'}{T}\right)^\alpha.$$

Run ***PTRR***$_\alpha$ for T_{rem} steps with parameters m' and τ'. From the analysis of ***PTRR***$_\alpha$ (Theorem 3.5), we know that

$$\mathbb{E}\big[\mathrm{ALG}_{T_{\text{rem}}}\big] \;\geq\; \frac{\mathrm{OPT}^{\text{res}}_{T_{\text{rem}}}}{C_\alpha c_2 (k+1)^{\alpha/(1+\alpha)}},$$

where $\mathrm{OPT}^{\text{res}}_{T_{\text{rem}}}$ denotes the optimal cumulative reward for $\{g_i\}$ over T_{rem} rounds and $C_\alpha = 2^{\alpha+2}(\alpha+1)$.

Now note that $\mathrm{OPT}^{\text{res}}_{T_{\text{rem}}} \geq \frac{1}{2} g^\star(T_{\text{rem}}) T_{\text{rem}}$, and that the algorithm's maximum single-pull reward dominates its average reward (Facts B.1 and B.3 in the appendix of [BR25]). Let $\hat{i}$ denote the arm that achieves this maximum single-pull reward, and note that

$$\mathbb{E}\Big[\max_{t \leq T_{\text{rem}}} \text{reward}_t\Big] \;\geq\; \frac{1}{T_{\text{rem}}} \mathbb{E}\big[\mathrm{ALG}_{T_{\text{rem}}}\big] \;\geq\; \frac{g^\star(T_{\text{rem}})}{2C_\alpha c_2 (k+1)^{\alpha/(1+\alpha)}}.$$

Since $f_{\hat{i}}(T) \geq \max_{t \leq T_{\text{rem}}} \text{reward}_t$ by monotonicity, we know that taking expectations gives

$$\mathbb{E}\big[f_{\hat{i}}(T)\big] \;\geq\; \frac{g^\star(T_{\text{rem}})}{2C_\alpha c_2 (k+1)^{\alpha/(1+\alpha)}}.$$

Monotonicity and concavity give $g^\star(T_{\text{rem}}) \geq (T_{\text{rem}}/T) f^\star(T) \geq \frac{1}{2} f^\star(T)$, as $T_{\text{rem}} \geq T/2$. Combining with $c_2 \leq 8$ from Step 1, it follows that

$$\mathbb{E}\big[f_{\hat{i}}(T)\big] \;\geq\; \frac{1}{2C_\alpha c_2 (k+1)^{\alpha/(1+\alpha)}} \cdot \frac{1}{2} f^\star(T) \;\geq\; \frac{1}{2^{\alpha+7}(\alpha+1)} (k+1)^{-\alpha/(1+\alpha)} f^\star(T),$$

as desired.

Finally, in terms of cumulative reward of the chosen vs. best arms:

$$\mathbb{E}\left[F_{\hat{i}}(T)\right] \geq \frac{T}{2} \mathbb{E}\left[f_{\hat{i}}(T)\right] \geq \frac{T}{2} \frac{1}{2^{\alpha+7}(\alpha+1)} (k+1)^{-\alpha/(1+\alpha)} f^\star(T) \geq \mathrm{OPT}_T \frac{1}{2^{\alpha+8}(\alpha+1)} (k+1)^{-\alpha/(1+\alpha)}$$

□

E.7 SAMPLE COMPLEXITY FOR TUNING α, B

In the previous sections, we described two hybrid algorithms which each proceed as follows: first, they spend a portion of the time attempting to solve exact best-arm identification (BAI) and then they revert to approximate best-arm identification if it is not successful. These algorithms differ only in the implementation of lines 6,7, and 8, i.e., in how the tracked variables are defined. In general, instances could lie anywhere along the spectrum of easy-to-identify to worst-case. Is there a better way to decide how *much* time to allocate to attempting exact BAI and when to switch to worst-case approximate BAI? Suppose the instances we see have some stationarity. In particular, if we see historical examples of instances arising from a

certain task, and we are expected to complete that task with a good algorithm in the future, we might model the instances as having been drawn from a distribution. In that case, we could hope to *learn* to what degree it is worth attempting exact BAI. In particular, we could hope to learn the time after which we should switch from the exact goal to the relaxed goal. We show that we can pick a switch time and a PTRR parameter with polynomially many samples from the distribution over instances that is almost as good as the optimal such time and parameter *on average* over the instances. Here, unlike in the previous sections, we require a distributional assumption to hold and access to samples from the distribution, and we achieve an *on average* rather than per instance guarantee. However, we adapt to the data distribution.

We study the fully hybrid algorithm family defined in Definition 4.3. Each algorithm in this family has two parameters: the first is the time B at which it switches from the exact BAI goal to the approximate BAI goal, and the second specifies which $PTRR_\alpha$ algorithm it runs *after* switching to the approximate BAI goal. Our goal is to learn from a set of offline instances which value of B and α gets the best performance on a new instance from the same distribution. In order to analyze this, we can apply the same set of data-driven algorithm design tools as before. Once again, we need to understand for a fixed instance, how many possible behaviors an algorithm from the family could have in terms of loss. To this end, it remains to (1) bound $Q_{\mathcal{D}}$ for this problem and (2) investigate various reasonable H-bounded losses. We do this next.

Bounding derandomized dual complexity in our setting. We bound $Q_{\mathcal{D}}$ by computing the number of possible behaviors of algorithms from $\mathcal{H}$.

Lemma E.7. *For the family $\mathcal{B}$ defined in Defn. 3.1, the improving multi-armed bandits problem, and* any *piecewise-constant loss function, $Q_{\mathcal{D}} \leq k\,T^2$.*

Sample complexity results. As in Section 3.3, we combine Thm. C.2 and Lemma E.7 to derive sample complexity results for generic H-bounded loss functions. We then instantiate it for both BAI and regret loss functions.

Theorem E.8. *For the Hyperparameter Transfer setting for tuning α, B in Algorithm 2 with generic H-bounded loss, $O\left(\left(\frac{H}{\epsilon}\right)^2(\log kT + \log\frac{1}{\delta})\right)$ instances drawn from $\mathcal{D}$ suffice to get the uniform convergence guarantee in Theorem C.2.*

Finally, we instantiate the loss function for BAI. We can study "maximum pull regret," and this loss is H-bounded for $H = m$. Note that since m will affect the sample complexity, we need an upper bound on it to know how many samples to draw.

Definition E.9. *Define "maximum pull regret" as $R_{mp}(T) \coloneqq \max_{i\in[k]} f_i(T) - \max_{i\in[k]} f_i(t_i)$, where t_i is the number of rounds for which the algorithm played arm i.*

Thus, with knowledge of an upper bound for the best pull of the best arm, we have that:

Corollary E.10. *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 C.2 for the $Hybrid_{\alpha,B}$ algorithm family.*

Thus, with polynomially many samples from a distribution over instances, we can achieve near-optimal loss on a new instance drawn from the same distribution.