# Competing against Adaptive Strategies in Online Learning via Hints

**Aditya Bhaskara**
University of Utah

**Kamesh Munagala**
Duke University

## Abstract

For many of the classic online learning settings, it is known that having a "hint" about the loss function before making a prediction yields significantly better regret guarantees. In this work we study the question, do hints allow us to go beyond the standard notion of regret (which competes against the best fixed strategy) and compete against adaptive or dynamic strategies? After all, if hints were perfect, we can clearly compete against a fully dynamic strategy. For some common online learning settings, we provide upper and lower bounds for the switching regret, i.e., the difference between the loss incurred by the algorithm and the optimal strategy in hindsight that switches state at most $L$ times, where $L$ is some parameter. We show positive results for online linear optimization and the classic experts problem. Interestingly, such results turn out to be impossible for the classic bandit setting.

## 1 Introduction

The problems of online prediction and online linear optimization are central to machine learning, online decision theory, and game theory. In the canonical online prediction or *experts problem* [31], there is a set of options (also called *arms*), each of which incurs an unknown and adversarially generated loss at each time step. A decision maker needs to choose an arm to play each time step, only with knowledge of all the losses incurred at previous time steps, and the goal is to compete with the total loss of the best arm in hindsight via the notion of *regret*. A generalization is *online linear optimization* [43], where there is an unknown linear cost function each step, and the decision maker needs to choose a point within a convex space, incurring as its loss, the cost function at that point. The goal again is to compete with the best fixed point applied to all cost functions.

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206. Copyright 2023 by the author(s).

Online learning problems typically differ in the information available to the decision maker. In the *bandit* setting [29, 3, 4], the decision maker only learns the loss of the arm played – that is, only the value of loss actually incurred – while the settings described above are *full information*, where the decision maker learns the losses of all the arms. Starting with seminal work in game theory and statistics [18, 7] and later in machine learning [31, 15, 43], sublinear regret bounds as well as matching lower bounds are known for several variants of this problem.

A line of research starting with [38] has asked the question: What if the decision maker is given a "hint" each time step about the loss vector that will arise at that step? Such a model can be natural in many settings. For instance, there can be a temporal correlation in the losses (thus previous losses could serve as a hint). Alternatively, one might be able to employ an auxiliary ML model trained on side-information to predict the loss sequence. The challenge in these settings is dealing with prediction errors (which are typically impossible to avoid). The goal then becomes to achieve a "best of both worlds" guarantee: if the hints are accurate, the regret should improve upon the standard bounds, while if the hints are inaccurate, the regret should be no worse asymptotically than that of the best policy that ignores the hints entirely.

Surprisingly, such a guarantee is indeed possible for both the experts problem [38, 40], and its bandit version [38, 41]. Further, in the context of online linear optimization when the domain is strictly convex, it was recently shown [21, 14, 5] that an exponentially better regret bound of $O(\log T)$ is achievable if the hint vector is "correlated" with the actual cost function at each step.

The above results assume that our algorithm is competing with the best fixed decision in hindsight. However, with a hint at every step, we can ask for more: after all, if hints are perfect, an algorithm can compete against the best *dynamic* (changing at every step) solution. In the classical online learning literature (without hints), sublinear regret bounds are known in many settings when competing against a solution that only changes a small number of times. This has been termed the *switching regret* and is parameterized by $L$, the number of changes/switches [24]. These works show that the online learning paradigm can adapt to changing environments without knowing the change points. For a horizon of $T$ steps, regret bounds with dependence of $O(\sqrt{(1+L)T})$ are known for the full information problems [24, 43, 22, 27, 42], and $O((1+L)\sqrt{T})$ for bandits [4]. In this work, we ask:

> Does the availability of hints allow us to obtain significantly better *switching regret* bounds for fundamental online learning problems like online linear optimization (OLO), experts, and bandits?

### 1.1 Our Results

At a high level, our results show a surprising difference between the full information and bandit problems with hints when switching is allowed. We show that "best of both worlds" guarantees are achievable for online linear optimization and for learning with experts, but are not possible in a very strong sense for the bandit problem, *even when only two switches are allowed for the comparator*!

Our positive results are as follows. First, we consider the setting of online linear optimization (OLO) when the domain is the unit ball.[1] Here, recent results [14, 5] showed that surprisingly, logarithmic regret bounds are possible as long as the hint vectors have *positive correlation* (measured by a possibly unknown parameter $\alpha$) with the loss at each step. Under the slightly stronger assumption that $\alpha$ is known, we prove a similar bound on the *adaptive regret* [22]. This yields an algorithm that achieves a switching regret bound of $O(((1+L)\log^2 T)/\alpha)$, where $L$ is the number of switches.

Second, we consider the classic problem of learning with expert advice. Although it is a special case of OLO where the domain is the simplex, our result above does not apply since the simplex is not strongly convex (indeed, it is easy to construct examples where the hint has a constant correlation at every time step but is still uninformative). However, a regret bound of a different form is possible: specifically, it is possible to achieve a regret that only depends on the total squared *prediction error* (defined as the difference between the true loss and the loss predicted by the hint) of the best arm in hindsight. The appeal of this result is also that it allows the hint/prediction to be arbitrarily bad for the non-optimal arms! Very recently, [12] extended this result to the case of switching regret, deriving regret that depends on the total squared prediction error along the optimal *path*. We give an alternative proof of this result by incorporating hints into the *sleeping experts* framework [16]. The regret bound for sleeping experts with hints is novel and does not follow from [12]. Our algorithm gives a way of incorporating hints into *fixed share* style algorithms [23], an idea of independent interest.

Finally, we show that an analogous result is not possible for the classic multi-armed bandit problem [4], even when $L = 1$ (i.e., only one switch is allowed for the optimal solution). We first show (Theorem 12) that a regret bound only depending on the total squared prediction error along the *optimal path* — which is what is possible for experts (Theorem 9) and for bandits with $L = 0$ (see [41]) — is impossible. Second, we show (Theorem 13) that even if we had a small squared prediction error for *every* arm (as opposed to low prediction error only along the optimal path), any online algorithm must incur $T^{\Omega(1)}$ regret, even for the case $L = 2$. In our lower bound example, we also know that the squared prediction error is 1, and thus the issue is not one of "tuning" to find the prediction error. These results may be viewed as complementing existing negative results for automatic parameter tuning and multi-scale learning in bandits [9, 12].

**Other Related Work.** For background on online learning and bandits, we refer to the excellent books of [10, 11, 19]. As discussed earlier, our work builds on the body of literature on *optimistic* regret bounds (see [38, 41] and references therein). For online linear optimization, our requirement on the strong convexity of the domain is reminiscent of the work of [26]. Indeed, the connection between the assumptions of [26] and the logarithmic regret bounds in [14] was made explicit in [6].

Apart from these, our work is related to the research on using ML predictions to improve the performance of online algorithms, an area that has received considerable recent attention. This model has been applied to a wide variety of classical online problems, including rent or buy problems [37, 17, 1], caching [33, 39, 28], scheduling [37, 30, 36], auctions [34], frequency estimation [25], Bloom filters [35], metrical task systems [2], etc.

## 2 Preliminaries

### 2.1 Learning with expert advice

In the classic experts problem [19], we have $M$ experts or *arms*; at each time step $t \in [T]$, an online algorithm is required to choose an expert (say $i_t$, possibly probabilistically), and *then* the losses $\ell_{t,i}$ (which denotes the loss of arm $i$ at time $t$) are revealed. The *regret* of the algorithm is defined as:

$$\text{Regret} = \mathbb{E}[\sum_{t=1}^{T} \ell_{t,i_t}] - \min_i \sum_{t=1}^{T} \ell_{t,i}.$$

The first term is the total expected loss incurred by the algorithm, while the second is the total loss of the best *fixed* expert in hindsight.

**Experts with hints.** In the experts with hints problem, before playing round $t$, we assume that the learner gets ac-

[1] As in previous works such as [14, 5], it is straightforward to extend our results to other strongly convex domains (where the strong convexity of a set is defined as in, say, Definition 2.1 of [14]).

cess to a prediction or hint for $\ell_{t,i}$ (for every $i$), which we denote by $m_{t,i}$. In this setting, the work of [40] shows that the classic regret bound of $\Theta(\sqrt{T})$ (which holds as long as $\ell_{t,i} \in [0,1]$)) can be improved to $\sqrt{\sum_t (\ell_{t,i^*} - m_{t,i^*})^2}$, where $i^*$ is the best expert in hindsight. In other words, the regret is bounded in terms of the $\ell_2$ "prediction error" for the best expert. This improves upon the earlier work of [38], whose bounds depend on the prediction errors of *all* experts.

**Switching regret.** Our focus in this paper is to compete against a *dynamic* optimum in hindsight, specifically, against the best sequence of experts that has at most $L$ switches. In other words, the switching regret, parameterized by $L$, is defined as

$$\text{Switching Regret} = \mathbb{E}[\sum_{t=1}^{T} \ell_{t,i_t}] - \min_{j_1, j_2, \dots j_T} \sum_{t=1}^{T} \ell_{t,j_t},$$

where the minimum is taken over all decision sequences $\vec{j}$ with $j_t \neq j_{t-1}$ for at most $L$ indices $t$. Very recently, [12], generalizing [40], gave an algorithm whose regret only depends on $L$, as well as the quantity $\mathcal{E} := \sqrt{\sum_{t=1}^{T} (\ell_{t,j_t} - m_{t,j_t})^2}$, *i.e.*, $\mathcal{E}$ is the $\ell_2$ prediction error *along the path* induced by the optimum sequence of experts. In Section 4, we present an alternative proof of this result via the *sleeping experts* problem.

**Sleeping Experts.** In the *sleeping experts* problem [16], at any time step $t$, an arbitrary subset of arms $A_t$ are awake, and the rest are asleep. The algorithm can only play one of the awake arms, and at the end of the step, learns the losses of all the awake arms. The total loss of an arm is simply the sum of the losses in steps it was awake, and the goal of the online algorithm is to achieve low regret against the total loss of any arm. For this problem, a regret bound of $O(\sqrt{T_i \ln M})$ is known when comparing against the total loss of arm $i$, where $T_i$ is the number of time steps this arm is awake [8, 13]. In Section 4, we show how to extend these results to incorporate hints.

### 2.2 Online linear optimization

Online linear optimization (OLO) is another classic online learning setting. Here, we are given a convex set $\mathcal{P}$ of possible decisions. At each of $T$ steps, the algorithm needs to choose a point $x_t \in \mathcal{P}$. After this, a linear cost function $c_t$ with $\|c_t\| \le 1$ is revealed, and the loss incurred is $\ell_t(x_t) = \langle c_t, x_t \rangle$. As before, the traditional notion of regret against the best fixed comparator is:

$$\text{Regret} = \sum_{t=1}^{T} \ell_t(x_t) - \min_{x \in \mathcal{P}} \sum_{t=1}^{T} \langle c_t, x \rangle.$$

It is well-known [43] that one can obtain $O(\sqrt{T})$ regret, and this is best possible under most natural assumptions on the domain and objective functions.

**OLO with hints.** In recent work [21, 14, 5], it was shown that if the domain $\mathcal{P}$ is strongly convex and if the learner is provided access to a *hint vector* $h_t$ that is correlated with the cost vector $c_t$ at every step $t$, the learner can achieve a regret $O(\log T)$, drastically improving upon the general case. In all our OLO results, we assume that the *domain is the unit ball* (in $\ell_2$ norm), which is strongly convex.

The works [14, 5] define $h_t$ to be $\alpha$-correlated with $c_t$ if $\langle c_t, h_t \rangle \ge \alpha \|c_t\|^2$. The regret bound achieved in [14] is $O\left(\frac{\log T}{\alpha}\right)$ if the hints are always $\alpha$-correlated. The assumption of correlation at *every* step was relaxed in [5], but both the works compare against the best *fixed* $x \in \mathcal{P}$. Here, we ask if we can compare against a sequence $u_1, u_2, \dots, u_T$ with $L$ switches. Formally,

$$\text{Switching Regret} = \sum_{t=1}^{T} \ell_t(x_t) - \min_{u_1, u_2, \dots u_T} \sum_{t=1}^{T} \langle c_t, u_t \rangle,$$

where $\ell_t(x_t)$ is the algorithm's loss at time $t$, and the minimum is taken over all sequences $\vec{u}$ with at most $L$ switches. In the absence of hints, an online gradient descent algorithm is known [43, 42] to achieve a dynamic regret bound of $O(\sqrt{(1+L)T})$. Our goal in Section 3 is to improve the dependence on $T$ to logarithmic assuming correlated hints and the domain being the unit ball.

### 2.3 The multi-armed bandit problem

The adversarial bandit problem [4] follows the same setting as learning with experts, but the difference is in the feedback received by the online algorithm. In the experts problem, the algorithm is provided with the losses of all the arms (i.e., experts) at the end of each step, whereas in bandits, the only feedback provided to the algorithm is the loss of the arm it actually played. The notion of regret, switching, and hints are now exactly the same as for the experts problem described above. For bandits, a switching regret bound of $O((1+L)\sqrt{T})$ in the absence of hints is presented in [4]. In Section 5, we wish to study if hints help with achieving dynamic regret that depends on a quantity analogous to $\mathcal{E}$ defined above. This was shown for $L = 0$ (no switches) in [41].

## 3 Switching Regret for OLO with Hints

We first consider the online linear optimization (OLO) problem where the domain is the unit ball. In the presence of *$\alpha$-correlated* hints (defined in Section 2), we prove that the well-known switching regret bound of $O(\sqrt{(1+L)T})$ (due to [43, 42]) can be improved to $O(\frac{(1+L)\log^2 T}{\alpha})$. As in the work of [5], we do not require the hint to be correlated with the cost vector at *every* step: the number of *bad* hints enters our regret bounds as a natural additional term. Throughout the paper, we assume that $\alpha$ is a *given* parameter. It is an interesting open question to obtain similar guarantees without knowing $\alpha$.

**Theorem 1.** *There exists an algorithm (Algorithm 1 below) for the OLO with hints problem on the unit ball (denoted $\mathcal{B}(0,1)$) with the following guarantee. Suppose $\{c_t, h_t\}_{t=1}^T$ denotes the sequence of cost and hint vectors, and let $B$ be the number of rounds in which the hint is* bad, *i.e., $\langle c_t, h_t\rangle < \alpha \|c_t\|^2$. The algorithm outputs a sequence of vectors $x_t \in \mathcal{B}(0,1)$ such that for every sequence $\{u_t\}_{t=1}^T$ of vectors in $\mathcal{B}(0,1)$ with $L$ switches, we have:*

$$\sum_{t=1}^T \langle c_t, x_t - u_t\rangle \le O\left((1+L)\cdot \frac{\log^2 T}{\alpha}\sqrt{B}\right).$$

Indeed, the $\sqrt{B}$ term can be replaced with $r_T$ as defined in Algorithm 1.

**Tightness of the bound.** Modulo logarithmic factors, for constant $\alpha$, our regret bound is $O(L\sqrt{B})$. We remark that the dependence on $\sqrt{B}$ is unavoidable, even in the case $L = 1$ (see, e.g., the lower bounds in [5]). It is also easy to see a regret lower bound of $\Omega(L)$ even when $\alpha = 1/2$ and all hints are good. Consider the following instance in 2D (with an orthogonal basis $e_0, e_1$, say), and $L > 1$: the hint $h_t$ is always $e_0$. For the first $(L-1)$ time steps, $c_t$ are of the form $\frac{1}{2}e_0 \pm \frac{\sqrt{3}}{2}e_1$, where the signs are chosen uniformly at random. Time $L$ and beyond, suppose that $c_t = e_0$. An $L$ switching optimum has loss $-T$, while any online algorithm incurs regret $\Omega(L)$ in the first $(L-1)$ steps. This instance shows that an $L$ dependence is needed even when all the hints are good. However, an overall bound of $O(L + \sqrt{LB})$ is not ruled out to the best of our knowledge.

The theorem above follows from the adaptive regret guarantee (low regret in every sub-interval [22]) presented below, which is our main technical result.

**Theorem 2.** *There exists an algorithm (Algorithm 1) for the OLO with hints problem on $\mathcal{B}(0,1)$ with the following guarantee. Suppose $\{c_t, h_t\}_{t=1}^T$ denote the sequence of cost and hint vectors, and let $r_t$ be defined as in Algorithm 1. Then for any interval $[a,b]$ and for any $u \in \mathcal{B}(0,1)$, the output $x_t$ of the algorithm satisfies*

$$\sum_{t=a}^b \langle c_t, x_t - u\rangle \le O\left(\frac{\log^2 T}{\alpha}\cdot r_{b+1}\right).$$

Note that the term $r_{b+1}$ is always upper bounded by the square root of the number of bad hints in time steps $[1,b]$. The algorithm and its analysis involve adapting the ideas from [5] and [22], and then using a meta algorithm that can help avoid knowledge of a certain crucial hyperparameter.

## 3.1 Adaptive algorithm for OLO with hints

The algorithm used in both the theorems is described in Algorithm 1). It assumes access to an "inner" online learning algorithm that is denoted by INNER.

**Algorithm 1** Adaptive OLO with Hints

```
input  Hints $h_t$ followed by costs $c_t$, parameter $\alpha$
  Initialize $r_1 = 1$.
  for $t = 1 \dots T$ do
    Receive hint $h_t$
    Get $\overline{x}_t$ from procedure INNER, and set
        $x_t = \overline{x}_t + \frac{(\|\overline{x}_t\|^2 - 1)}{2r_t} h_t$
    Play $x_t$ and receive cost vector $c_t$
    Define $z_t = \max(0,\ \alpha\|c_t\|^2 - \langle c_t, h_t\rangle)$, and set
        $r_{t+1} = \sqrt{r_t^2 + z_t}$
    Define the function:
    $\ell_t(x) = \langle c_t, x\rangle + \frac{\max(\alpha\|c_t\|^2, \langle c_t, h_t\rangle)}{2r_t}\left(\|x\|^2 - 1\right)$
    Send $\ell_t$ as the loss at time $t$ to procedure INNER
  end for
```

**Algorithm 2** Procedure INNER

```
input  Loss functions $\ell_t$, knowledge of time horizon $T$
  Initialize $c^* = 1$, $\mathcal{C} = \{2^{-j}; 0 \le j \le 2\lceil \log T\rceil\}$
  Start procedure INNER[$\delta$] for all $\delta \in \mathcal{C}$, in parallel (as
  described in the proof of Lemma 5)
  for $t = 1 \dots T$ do
    Let $c^*$ be the largest $c \in \mathcal{C}$ for which $\ell_s$ are $c$-exp
    concave, for $1 \le s < t$
    Return the output of INNER[$c^*$]
    Receive $\ell_t$; send it to all the INNER[$\delta$]
  end for
```

**Outline.** Algorithm 1 follows the same structure as the algorithm from [5], but in order to show an adaptive guarantee (i.e., regret bound for every interval), we need different parameters and a different procedure INNER. As in [5], Algorithm 1 uses the inner learner to obtain a point $\overline{x}_t$, and then moves along the (negative) direction of the hint by an amount proportional to $r_t$, which is (the inverse of) the "trust" in the hints so far. The analysis relates the regret of the algorithm to the regret of the inner learner (Lemmas 3 and 4). While the inner learner in [5] is a simple FTRL procedure, here we need to design a new algorithm and show that it has a low adaptive regret. The choice of parameters $r_t$ and $z_t$ in Algorithm 1 are also important: they ensure that the functions $\ell_t$ passed to INNER have a *time-independent* exp-concavity parameter, which is needed for our analysis.

We now begin the analysis of Algorithm 1 with an elementary lemma:

**Lemma 3.** *(Proved in Appendix A.) Let $\overline{x}_t$ be any point in $\mathcal{B}(0,1)$. Then we have:*

1. *As long as $\|h_t\| \le 1$, we have $x_t \in B(0,1)$. (I.e., the algorithm always plays a feasible point.)*

2. *Loss function $\ell_t(x)$ is $2\,\|c_t\|$-Lipschitz and is strongly convex with a parameter $\geq \alpha\,\|c_t\|^2/r_t$ for all t.*

3. *If the hint $h_t$ is good, then $\ell_t(\overline{x}_t) = \langle c_t, x_t\rangle$.*

4. *Whether the hint $h_t$ is good or bad, we have*

$$\langle c_t, x_t\rangle \leq \ell_t(\overline{x}_t) + \frac{z_t}{2r_t}.$$

Part (1) of the lemma is important as it shows that the algorithm always plays a feasible point. Part (2) turns out to be important in the proof of Theorem 2.

Next, we show how to bound the regret of Algorithm 1 by the regret of INNER. To this end, let us denote by $\mathcal{R}_{\text{INNER}}[a,b]$ an upper bound on the regret of INNER over the interval $[a,b]$. I.e., assume that $\forall u \in \mathcal{B}(0,1)$,

$$\sum_{t=a}^{b} \ell_t(\overline{x}_t) - \ell_t(u) \leq \mathcal{R}_{\text{INNER}}[a,b].$$

**Lemma 4.** *(Proved in Appendix A.) Consider any time interval $[a,b]$ and let $u$ be any vector in $B(0,1)$. Then the point $x_t$ returned by Algorithm 1 satisfies*

$$\sum_{t=a}^{b} \langle c_t, x_t - u\rangle \leq \mathcal{R}_{\text{INNER}}[a,b] + 2(r_{b+1} - r_a).$$

## 3.2 INNER **algorithm**

The algorithm INNER works by "combining" algorithms INNER$[\delta]$ for different values of $\delta$. So we begin by first describing INNER$[\delta]$ that assumes an auxiliary parameter $\delta$. The combination procedure is described in Section 3.2.1.

**Lemma 5.** *Let $\{\ell_t\}_{t=1}^{T}$ be a sequence of functions such that for all t, $\ell_t$ is $\gamma_t$-Lipschitz and $\alpha_t$ strongly convex, where $\gamma_t > 0$ and $2 \geq \alpha_t \geq \delta\gamma_t^2$ for some* known *parameter $\delta > 0$. There exists an online algorithm* INNER$[\delta]$ *that at each time step outputs a point $\overline{x}_t \in \mathcal{B}(0,1)$, such that for all intervals $[a,b]$ and for all $u \in \mathcal{B}(0,1)$,*

$$\sum_{t=a}^{b} \ell_t(\overline{x}_t) - \ell_t(u) \leq O\left(\frac{\log^2 T}{\delta}\right).$$

*Moreover, for $\ell_t$ defined in Algorithm 1, $\overline{x}_t$ can be computed in time $O(d \log T)$, where $d$ is the dimension of the domain.*

*Proof.* The first observation is that the functions $\ell_t$ have Exp-Concavity parameter at least $\delta$, which is independent of the time $t$. This holds because the exp-concavity parameter of a function that is $G$-Lipschitz and $S$-strongly convex is at least $S/G^2$ (see, e.g., [20]).

Then, we use the results of [22] to show the existence of INNER$[\delta]$ with the desired properties. Specifically, by combining Theorem 4.1 with Theorem 3.1 of [22][2], we have the following.

**Theorem 6.** *Suppose $\delta > 0$ is a known parameter, and suppose $\ell_1, \ell_2, \ldots, \ell_T$ are $\delta$-exp concave loss functions over a (fixed) convex domain. Let $\mathcal{A}$ be an algorithm with the property that its regret over the interval $[1,t]$ is bounded by $O(\log t)/\delta$. Then there exists an algorithm $\mathcal{A}'$ whose run time is a factor $O(\log T)$ worse than that of $\mathcal{A}$, that for every* interval $[a,b]$, *has a regret over that interval upper bounded by $O(\log^2 T/\delta)$.*

In other words, the theorem allows one to move from a bound over a time prefix (that many algorithms enjoy) to a bound over any interval. Note also that for us, it is not important to have a bound that depends only on the length of the interval (this is an important concern in [22]). In summary, we have shown that it suffices to give an algorithm $\mathcal{A}$ whose regret over the time interval $[1,T]$ is $O((\log T)/\delta)$.

For our functions $\ell_t$, such $\mathcal{A}$ can be obtained either using FTRL or online gradient descent by leveraging the strong convexity assumption on $\ell_t$. For concreteness, let $\mathcal{A}$ be the online gradient descent algorithm with the step size $\eta_t = \frac{1}{1+\alpha_{1:t}}$, initialized anywhere in $\mathcal{B}(0,1)$. A standard calculation (see Chapter 3 of [19]) shows that the regret (over the $T$ steps) is bounded by:

$$\sum_{t=1}^{T} \frac{\gamma_t^2}{1+\alpha_{1:t}} \leq \frac{1}{\delta}\sum_{t=1}^{T} \frac{\alpha_t}{1+\alpha_{1:t}}$$
$$\leq \frac{1}{\delta}\sum_{t=1}^{T} \int_{1+\alpha_{1:(t-1)}}^{1+\alpha_{1:t}} \frac{dz}{z} = \frac{\log(1+\alpha_{1:T})}{\delta}.$$

Since $\alpha_t \leq 2$ for all $t$, the regret is at most $O((\log T)/\delta)$, thus completing the proof.

The "moreover" part of the theorem follows once again from Theorem 4.1 of [22] together with the observation that for the functions $\ell_t$ we work with, a single step of online gradient descent takes only $O(d)$ time. □

### 3.2.1 Combining algorithms for different $\delta$

In order to obtain our final guarantee for OLO, we need to apply INNER$[\delta]$ for an unknown value of $\delta$. To achieve this, we argue that as long as we have a bounded range of candidate values, the guarantees can be combined effectively.

**Theorem 7.** *Let $\{\ell_t\}_{t=1}^{T}$ be a sequence of functions that satisfy the hypothesis of Lemma 5 for some (unknown) parameter $\delta$ in the interval $[\frac{1}{\Delta}, 1]$, where $\Delta$ is known. Then*

[2]Note that the algorithms of [22] require a knowledge of $\delta$, which is why we need to as well. We also note that the Theorem numbers above refer to the version of [22] available at `https://eccc.weizmann.ac.il/report/2007/088/download`.

*there exists an online algorithm* INNER *that at each step outputs an* $\overline{x}_t \in \mathcal{B}(0,1)$*, with the property that for all intervals* $[a,b]$ *and for all* $u \in \mathcal{B}(0,1)$*,*

$$\sum_{t=a}^{b} \ell_t(\overline{x}_t) - \ell_t(u) \le O\left(\frac{\log^2 T}{\delta}\right). \tag{1}$$

*For* $\ell_t$ *of the form we consider in Algorithm 1, the running time per step is* $O(d \log T \log \Delta)$*.*

Note that the regret bound only depends on the (unknown) parameter $\delta$ and has no dependence on the "range" $\Delta$.

*Proof.* We consider INNER to be the following meta algorithm. Let $\mathcal{C}$ be the set of candidate $\delta$ values,

$$\mathcal{C} = \{2^{-j} : 0 \le j \le \lceil \log_2 \Delta \rceil\}.$$

The meta algorithm runs the algorithms INNER$[c]$ in parallel for all $c \in \mathcal{C}$. At $t = 1$, INNER returns the output of the INNER$[1]$. As we see the functions $\ell_t$, we keep track of the largest value of $c \in \mathcal{C}$ for which the $\ell_t$ so far satisfy the hypothesis of Lemma 5 with $\delta = c$, and use the output of the corresponding copy INNER$[c]$. Because we run all the algorithms in parallel, the run time bound is easy to see.

We can analyze the regret of the meta algorithm as follows. Consider any interval $[a, b]$ and any $u \in B(0, 1)$. Suppose we used the output of INNER$[c]$ for some fixed $c \in \mathcal{C}$ throughout $[a, b]$, we can use the bound from Lemma 5 directly. Otherwise, we can split the interval $[a, b]$ into subintervals $[a, a_1], [a_1, a_2], \dots, [a_{r-1}, b]$, where we use the values $c_1, c_2, \dots, c_r$ respectively. We can apply the guarantee from Lemma 5 for INNER$[c_j]$ in the interval $[a_{j-1}, a_j]$, and thus bound the LHS of (1) by

$$O(\log^2 T) \cdot (2^{c_1} + 2^{c_2} + \cdots + 2^{c_r})$$

The largest term dominates the summation, and thus we can bound the sum by $O(\log^2 T) \cdot 2^{c_r}$. Since we assumed that the hypothesis of Lemma 5 holds for some unknown $\delta$ for all $t$, we must have $2^{-c_r} \ge \delta$, and thus (1) follows. ☐

Using these results, we can complete the proofs of Theorems 2 and 1.

*Proof of Theorem 2.* The idea is to combine the bounds from Lemma 4 and Theorem 7. We plug in $\frac{\alpha}{r_{b+1}}$ as the unknown strong convexity parameter in Theorem 7 (this follows from part (2) of Lemma 3). Since $\alpha$ can be assumed to be $> 1/T$ and $r_{b+1} < T$, we can use the value $\Delta = T^2$.[3] Thus, we have that for $x_t$ output by Algorithm 1,

$$\sum_{t=a}^{b} \langle c_t, x_t - u \rangle \le 2(r_{b+1} - r_a) + O\left(\frac{r_{b+1}\log^2 T}{\alpha}\right).$$

[3]This assumes a knowledge of the time horizon, which can be removed via the doubling trick.

The second term clearly dominates, and thus the theorem follows. ☐

*Proof of Theorem 1.* Suppose the optimal $L$-switching strategy plays the vectors $u_1, u_2, \dots, u_L, u_{L+1}$ in the intervals $[1, A_1), [A_1, A_2), \dots, [A_{L-1}, A_L), [A_L, T]$ respectively, we can use the bound from Theorem 2 in each of the intervals to obtain a total regret bound of

$$O\left(\frac{\log^2 T}{\alpha} \cdot (r_{A_1} + \cdots + r_{A_L} + r_T)\right).$$

Naïvely bounding each $r_{A_j}$ by $r_T$ completes the proof. ☐

## 4 Switching Regret for Experts via Sleeping Experts

In this section, we will present a simple algorithm for sleeping experts that incorporates hints. It generalizes the result of [40] and yields an additional "gain" term in the regret bound, analogous to [12]. We then show that it can be used to obtain a switching regret guarantee.

Recall the sleeping experts problem from Section 2, where the learner additionally has access to a hint or prediction about the losses before playing. Let $m_{t,i}$ be the prediction for $\ell_{t,i}$, the loss of expert $i$ at time $t$. Define the *prediction error* as $\varepsilon_{t,i} = \ell_{t,i} - m_{t,i}$ (which can be positive or negative). Let $y_t$ denote the expected loss of the algorithm in the $t$th round, i.e., $y_t = \sum_{i \in A_t} p_{t,i}\ell_{t,i}$ where $p_t$ is the probability distribution over the awake arms at time $t$ used by the algorithm. Finally, define $\mathsf{act}(i)$ as $\{t : i \in A_t\}$, i.e., the rounds in which expert $i$ is awake or *active*. Then we have

**Theorem 8.** *Suppose* $y_t$ *is the expected loss incurred by Algorithm 3 at time t, and let* $p_t$ *be the sampling distribution defined in the algorithm. Then for all* $i \in [M]$*, we have:*

$$\sum_{t \in \mathsf{act}(i)} y_t \le \sum_{t \in \mathsf{act}(i)} \ell_{t,i} + \frac{\log M}{\eta} + \eta \sum_{t \in \mathsf{act}(i)} \varepsilon_{t,i}^2 - \frac{\eta}{4}\sum_t \left(\sum_{i \in A_t} p_{t,i}\varepsilon_{t,i}\right)^2.$$

**Remarks.** Note that while our regret bound has an additional negative term similar to that of [12], their results do not apply to the sleeping setting. This is because, while their main algorithm MsMwC allows for the feasible domain (which they call $\Omega_t$) to vary at every time step, the point they compare with when bounding the regret is assumed to lie in $\bigcap_t \Omega_t$. For our result, we need a bound that only sums over the steps in which the expert being compared to is awake.

On another note, our algorithm is parameterized by $\eta$, which is given. The impossibility theorem of [12] can be

**Algorithm 3** Sleeping Experts with Hints

```
input Set of awake experts $A_t$, predictions $m_{t,i}$ for the awake experts, followed by true losses $\ell_{t,i}$; parameter $\eta$
  Maintain weights $w_{t,i}$ for expert $i$ at time $t$; initialize $w_{1,i} = 1$ for all $i$
  for $t = 1 \dots T$ do
    Receive set $A_t$, predictions $m_{t,i}$ for all $i \in A_t$.
    Choose an expert $i \in A_t$ with probability:
        $p_{t,i} := \frac{w_{t,i} e^{-\eta m_{t,i}}}{\sum_{j \in A_t} w_{t,j} e^{-\eta m_{t,j}}}$
    Observe the true losses $\ell_{t,i}$ and compute the prediction errors $\varepsilon_{t,i}$
    For all $i \in A_t$ set weights for time $(t+1)$ as:
        $w_{t+1,i} = w_{t,i} \cdot e^{-\eta m_{t,i}} (1 - \eta \varepsilon_{t,i})$
    Define $\gamma_t := \frac{\sum_{i \in A_t} w_{t+1,i}}{\sum_{i \in A_t} w_{t,i}}$
    For all $i \notin A_t$, set weights for time $(t+1)$ as $w_{t+1,i} = w_{t,i} \cdot \gamma_t$
  end for
```

used to prove that automatically (and separately) tuning $\eta$ for each arm, which would let us obtain regret that only depends on the $\ell_2$ prediction error for that arm, is impossible.

In Section 4.2 (along with the Supplement), we show how to use Theorem 8 to prove the following:

**Theorem 9.** *Consider the experts with hints problem with $M$ experts, losses $\ell_{t,i}$ and predictions $m_{t,i}$. There exists an online algorithm with running time $O(M \log T)$ per round that achieves the following regret bound: for any sequence $i_1, i_2, \dots, i_T$ of experts with at most $L$ switches, we have*

$$\sum_{t=1}^{T} y_t - \ell_{t,i_t} \le O\left(\sqrt{(L+1)\mathcal{E} \log M \log T}\right),$$

*where $y_t$ is the expected loss of the algorithm at time t and $\mathcal{E}$ is the total squared prediction error along $i_1, \dots, i_T$, i.e., $\mathcal{E} = \sum_{t=1}^{T} (\ell_{t,i_t} - m_{t,i_t})^2$.*

We note that this theorem itself is not novel [12], but we give an alternative proof starting with our result above for sleeping experts.

## 4.1 Algorithm for sleeping experts with hints and its analysis

Algorithm 3 consists of weights $w_{t,i}$ (for arm $i$ at time $t$) that are used to define the sampling probabilities. While the update rule can be viewed as mirror descent with a correction term (as in [40]), we will state it simply as a hybrid between the hedge and the multiplicative weight update rules. This view allows us to incorporate the *fixed share* idea [23] for handling sleeping experts.

The analysis proceeds by considering the standard potential $\Phi_t = \sum_{i \in [M]} w_{t,i}$, and then showing the following lemma.

**Lemma 10.** *Consider any time step $t$, and let $\delta_t := \sum_{i \in A_t} p_{t,i} \varepsilon_{t,i}$. Then we have the following two inequalities*

$$\frac{\Phi_{t+1}}{\Phi_t} \le e^{-\eta y_t - \eta^2 \delta_t^2}, \tag{2}$$

$$\forall i \in A_t, \quad \frac{w_{t+1,i}}{w_{t,i}} \ge e^{-\eta \ell_{t,i} - \eta^2 \varepsilon_{t,i}^2}. \tag{3}$$

The proof is via elementary calculation, but it is important to note the additional quadratic term obtained in (2).

*Proof.* From the way $w_{t+1,i}$ are defined for $i \notin A_t$ (see Algorithm 3), we have that

$$\frac{\Phi_{t+1}}{\Phi_t} = \gamma_t = \frac{\sum_{i \in A_t} w_{t+1,i}}{\sum_{i \in A_t} w_{t,i}}.$$

Thus, using the definition of $p_{t,i}$, we obtain

$$\frac{\Phi_{t+1}}{\Phi_t} = \frac{\sum_{i \in A_t} p_{t,i} (1 - \eta \varepsilon_{t,i})}{\sum_{i \in A_t} p_{t,i} e^{\eta m_{t,i}}}.$$

Next, since the $p_{t,i}$ add up to 1 by definition, we have that the numerator is equal to

$$1 - \eta \sum_{i \in A_t} p_{t,i} \varepsilon_{t,i} = 1 - \eta \delta_t \le e^{-\eta \delta_t - \frac{\eta^2}{4} \delta_t^2}.$$

For the last inequality, we used the fact that for $|t| < 1$, $1 - t \le e^{-t - \frac{t^2}{4}}$. The denominator can now be bounded using the convexity of the exponential.

$$\sum_{i \in A_t} p_{t,i} e^{\eta m_{t,i}} \ge e^{\eta \sum_{i \in A_t} p_{t,i} m_{t,i}}.$$

Noting that $\ell_{t,i} = m_{t,i} + \varepsilon_{t,i}$, we have

$$\frac{\sum_{i \in A_t} w_{t+1,i}}{\sum_{i \in A_t} w_{t,i}} \le e^{-\eta \sum_{i \in A_t} p_{t,i} \ell_{t,i} - \frac{\eta^2}{4} \delta_t^2},$$

which completes the proof of (2).

To see (3), we simply use the fact that $1 - t \ge e^{-t - t^2}$ for $|t| \le 1/2$ to conclude that

$$\frac{w_{t+1}}{w_t} = e^{-\eta m_{t,i}} (1 - \eta \varepsilon_{t,i}) \ge e^{\eta m_{t,i} - \eta \varepsilon_{t,i} - \eta^2 \varepsilon_{t,i}^2}.$$

This completes the proof, since $m_{t,i} + \varepsilon_{t,i} = \ell_{t,i}$. □

Theorem 8 now follows from the Lemma as follows.

*Proof of Theorem 8.* Since all the $w_{t,i}$ are non-negative, we have that for any $i$, $w_{T+1,i} \le \Phi_{T+1}$. Thus, we can write

$$w_{1,i} \prod_{t=1}^{T} \frac{w_{t+1,i}}{w_{t,i}} \le \Phi_1 \prod_{t=1}^{T} \frac{\Phi_{t+1}}{\Phi_t}. \tag{4}$$

This is because the LHS simplifies to $w_{T+1,i}$ and the RHS to $\Phi_{T+1}$. Now by construction, if $t \notin \mathsf{act}(i)$, we have $w_{t+1,i}/w_{t,i} = \gamma_t = \Phi_{t+1}/\Phi_t$, and thus the terms cancel out. Thus, using the initialization $w_{1,i} = 1$ for all $i$, the inequality above is equivalent to

$$\prod_{t\in\mathsf{act}(i)} \frac{w_{t+1,i}}{w_{t,i}} \le M \prod_{t\in\mathsf{act}(i)} \frac{\Phi_{t+1}}{\Phi_t}$$
$$\le M e^{-\sum_{t\in\mathsf{act}(i)}(\eta y_t + \frac{\eta^2}{4}\delta_t^2)},$$

where the last step uses (2). Now using (3), we have

$$e^{-\sum_{t\in\mathsf{act}(i)}(\eta \ell_{t,i} + \eta^2\varepsilon_{t,i}^2)} \le M \cdot e^{-\sum_{t\in\mathsf{act}(i)}(\eta y_t + \frac{\eta^2}{4}\delta_t^2)}.$$

Simplifying this yields the theorem. □

### 4.2 Switching regret

Using Theorem 8, we show the following lemma (proof in Appendix B.1).

**Lemma 11.** *Consider the experts with hints problem with $M, m_{t,i}, \ell_{t,i}$ and $\varepsilon_{t,i}$ as defined in Theorem 9. There exists an online algorithm parameterized by $\eta > 0$ with running time per round $O(M \log T)$, that at every step produces a probability distribution $p_t$ over experts and satisfies the following regret bound: for any L-switch sequence $i_1, i_2, \ldots, i_T$,*

$$\sum_{t=1}^{T} y_t - \ell_{t,i_t} \le \frac{(1+L)\log M \log T}{\eta} + \eta\mathcal{E}$$
$$- \frac{\eta}{4}\sum_t \Big( \sum_{i\in[M]} p_{t,i}\varepsilon_{t,i} \Big)^2,$$

*where $y_t$ is the expected loss at time t and $\mathcal{E}$ is the total squared prediction error along $i_1, \ldots, i_T$.*

This can then be combined with a *multi-scale multiplicative weight* procedure from [9, 12] to obtain Theorem 9. The details are deferred to Appendix B.2.

## 5 Lower Bounds for Bandits

Unlike the full information settings above, we show that switching regret guarantees are impossible in an information theoretical sense for the classical non-stochastic multi-armed bandit problem.

We show two lower bounds. In the first, we show that obtaining a regret bound that only depends on the prediction error "along the optimal path" as in Theorem 9 is impossible. More concretely, we show in Theorem 12 that in order to compete with a strategy that (say) plays arm-1 for a certain time period and switches to arm-2 at time $t$, we need good predictions for the loss of arm-2 at times $< t$. Second, we show that obtaining a regret guarantee that only depends on the "sum of squared prediction errors" is impossible if we wish to obtain switching regret (Theorem 13). Note that all the known optimism bounds [40, 41] are of this nature. Our results may be viewed as saying that obtaining *switching regret analogs* of these results is impossible – even with two switches! These results add to the existing negative results for bandits, including the impossibility of automatic "learning rate tuning" [12] and multi-scale multiplicative weights [9].

**Theorem 12.** *There exists a distribution over instances for the problem of bandits with hints with the following properties: (1) The prediction error* along *the best one-switch strategy in hindsight is zero; and (2) Any online algorithm has an expected regret $\Omega(\sqrt{T})$.*

*Proof.* Consider a setting with two arms, denoted 1 and 2, and suppose there are $T$ rounds in total. Suppose for the first arm, we have $\ell_{t,1} = m_{t,1} = 1/2$ for all $t$. For the second arm, suppose that $m_{t,2} = 0$ for all $t$. For the true loss, suppose that we choose a random $t_0 \in [T]$, and we set $\ell_{t,2} = 1$ for $t < t_0$ and $\ell_{t,2} = 0$ for $t \ge t_0$.

This defines a distribution over instances, parameterized by $t_0$ (call this $\mathcal{D}_\mathcal{I}$). For each instance in the distribution, the best one-switch strategy in hindsight is to play arm-1 for the first $t_0 - 1$ steps, then switch to arm-2. Along this strategy, the prediction error is zero.

Now, we claim that for any online algorithm $\mathcal{A}$ for solving the problem, the expected regret on an instance sampled from this distribution is $\Omega(\sqrt{T})$. Since we are giving an explicit distribution over the instances, Yao's minimax theorem lets us assume that $\mathcal{A}$ is deterministic. Any such algorithm can be viewed as a decision tree, where at each step it pulls an arm, and branches based on the observed cost. As the first arm is fixed as are the predictions for the second arm, the only time the algorithm learns about $t_0$ is by pulling arm-2. Thus, we define $Y_t$ as a binary variable that indicates if the algorithm pulls arm-2 at time $t$ *assuming* that it has *not seen* a cost of 0 on arm-2 so far. (If $t_0 > t$, this is guaranteed to be well defined.)

*Case 1.* We have $\sum_{t=1}^{T/2} Y_t \ge \sqrt{T}/2$. In this case, in all the instances in $\mathcal{D}_\mathcal{I}$ with $t_0 > T/2$, the algorithm $\mathcal{A}$ ends up pulling arm-2 at least $\sqrt{T}/2$ times (thus incurring a loss of 1 as opposed to $1/2$ attainable by pulling arm-1). Thus the regret is $\ge \sqrt{T}/4$. Since the case $t_0 \ge T/2$ occurs with probability $1/2$, the expected regret overall is $\ge \sqrt{T}/8$.

*Case 2.* $\sum_{t=1}^{T/2} Y_t < \sqrt{T}/2$. Intuitively, in this case, arm-2 is not pulled frequently enough, thus in the case when $t_0 < T/2$, there is a good chance of *missing* the switch to zero loss for arm-2 by a length $\Omega(\sqrt{T})$. To formalize this, let $r = \sqrt{T}/2$. Define $s_i = Y_i + Y_{i+1} + \cdots + Y_{i+r-1}$. Now consider the sum $A = s_1 + s_2 + \cdots + s_{T/2-r}$. Each $Y_i$ appears at most $r$ times in this summation, and thus $A \le$

$r(Y_1 + Y_2 + \cdots + Y_{T/2}) \le r\sqrt{T}/2$. By our choice of $r$, the RHS is $< T/4$. Thus, $s_1 + s_2 + \cdots + s_{T/2-r} < T/4$, and thus for at least $T/4 - r$ indices $i < T/2 - r$, the algorithm does not play arm-2 in the interval $[i, i + r - 1]$ assuming it has not seen a cost of 0 on arm-2 before $i$. Let us call such an index $i$ "good". Thus, conditioned on $t_0 \in [i, i + r - 1]$, the expected regret for a good $i$ is $\ge r/2$. Since there are $T/4 - r > T/6$ good indices $i$, if we denote by $\mathcal{E}_i$ the event that $t_0 \in [i, i + r - 1]$, the probability that $\mathcal{E}_i$ occurs for some good $i$ is $\Omega(1)$. Thus, the expected regret overall is $\Omega(\sqrt{T})$. □

The next theorem shows that even if the sum of squared prediction errors is small (known to be bounded by 1) for *all* the arms —not just for arms along the optimal path— a poly($T$) regret is unavoidable.

**Theorem 13.** *There exists a distribution over instances for bandits with hints, such that: (a) the total squared prediction error of each of the arms is bounded by* 1*, and (b) any online algorithm has an expected regret* $\Omega(T^{1/10})$.

The proof uses a slightly more involved instance where the optimum solution has $L = 2$ switches. We have two arms; for arm-1, we have $m_{t,1} = \ell_{t,1} = 1/\sqrt{T}$ for all $t$. For arm-2, we set $m_{t,2} = C/\sqrt{T}$ for all $t$, for a parameter $C$ that we will choose later. Then we pick a sub-interval $J$ of length $\frac{T}{C^2}$ uniformly at random in the full interval $[1, T]$, and for all $t \notin J$, we set $\ell_{t,2} = C/\sqrt{T}$ (so the prediction is perfect), and for $t \in J$, we set $\ell_{t,2} = 0$. Note that the total squared prediction error is $\sum_{t \in J} \frac{C^2}{T} = 1$.

The analysis for this instance follows an argument similar to that for Theorem 12 and we defer the details to Appendix C. It is natural to ask if the lower bound in Theorem 13 can be improved to $\Omega(\sqrt{T})$ using $L = O(1)$. We leave this as an interesting open question.

## 6 Conclusion

In summary, our main algorithmic contributions are the design of a new adaptive regret algorithm for online linear optimization (Section 3) and a sleeping experts algorithm (Section 4), both of which exploit hints about the loss vectors. These algorithms are used to obtain switching regret bounds. Conceptually, these results together with our lower bounds help us conclude that for online learning problems, obtaining "best of both worlds" guarantees that can exploit hints against a non-stationary comparator crucially requires the decision maker to have full information feedback. Our results also raise the question of whether hints can similarly improve regret bounds for other settings, such as that of *dynamic* regret, or for contextual variants of online learning. These are interesting avenues for future research.

**Acknowledgements**

Aditya Bhaskara acknowledges support from NSF grant CCF-2047288. Kamesh Munagala acknowledges support from NSF grant CCF-2113798.

## References

[1] Keerti Anand, Rong Ge, and Debmalya Panigrahi. Customizing ml predictions for online algorithms. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, 2020.

[2] Antonios Antoniadis, Christian Coester, Marek Elias, Adam Polak, and Bertrand Simon. Online metric algorithms with untrusted predictions. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, 2020.

[3] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2-3):235–256, 2002.

[4] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002.

[5] Aditya Bhaskara, Ashok Cutkosky, Ravi Kumar, and Manish Purohit. Online learning with imperfect hints. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 822–831. PMLR, 2020.

[6] Aditya Bhaskara, Ashok Cutkosky, Ravi Kumar, and Manish Purohit. Logarithmic regret from sublinear hints. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[7] David Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6(1):1 – 8, 1956.

[8] Avrim Blum and Yishay Mansour. From external to internal regret. *Journal of Machine Learning Research*, 8(47):1307–1324, 2007.

[9] Sebastien Bubeck, Nikhil R. Devanur, Zhiyi Huang, and Rad Niazadeh. Online auctions and multi-scale online learning. EC '17, page 497–514, New York, NY, USA, 2017. Association for Computing Machinery.

[10] Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[11] Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games.* Cambridge University Press, 2006.

[12] Liyu Chen, Haipeng Luo, and Chen-Yu Wei. Impossible tuning made possible: A new expert algorithm and its applications. In Mikhail Belkin and Samory Kpotufe, editors, *Conference on Learning Theory, COLT 2021, 15-19 August 2021, Boulder, Colorado, USA*, volume 134 of *Proceedings of Machine Learning Research*, pages 1216–1259. PMLR, 2021.

[13] Amit Daniely, Alon Gonen, and Shai Shalev-Shwartz. Strongly adaptive online learning. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1405–1411, Lille, France, 07–09 Jul 2015. PMLR.

[14] Ofer Dekel, arthur flajolet, Nika Haghtalab, and Patrick Jaillet. Online learning with a hint. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[15] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[16] Yoav Freund, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, page 334–343, 1997.

[17] Sreenivas Gollapudi and Debmalya Panigrahi. Online algorithms for rent-or-buy with expert advice. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 2319–2327, 2019.

[18] James Hannan. Approximation to Bayes risk in repeated play. In *Contributions to the Theory of Games (AM-39), Volume III*, pages 97–140. Princeton University Press, 2016.

[19] Elad Hazan. Introduction to online convex optimization. *CoRR*, abs/1909.05207, 2019.

[20] Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

[21] Elad Hazan and Nimrod Megiddo. Online learning with prior knowledge. In *Proceedings of the 20th Annual Conference on Learning Theory*, COLT'07, page 499–513, 2007.

[22] Elad Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 393–400, New York, NY, USA, 2009. Association for Computing Machinery.

[23] Mark Herbster and Manfred Warmuth. Tracking the best expert. In Armand Prieditis and Stuart Russell, editors, *Machine Learning Proceedings 1995*, pages 286–294. Morgan Kaufmann, San Francisco (CA), 1995.

[24] Mark Herbster and Manfred K. Warmuth. Tracking the best expert. *Mach. Learn.*, 32(2):151–178, aug 1998.

[25] Chen-Yu Hsu, Piotr Indyk, Dina Katabi, and Ali Vakilian. Learning-based frequency estimation algorithms. In *International Conference on Learning Representations*, 2019.

[26] Ruitong Huang, Tor Lattimore, András György, and Csaba Szepesvári. Following the leader and fast rates in online linear prediction: Curved constraint sets and other regularities. *JMLR*, 18(145):1–31, 2017.

[27] Ali Jadbabaie, Alexander Rakhlin, Shahin Shahrampour, and Karthik Sridharan. Online Optimization : Competing with Dynamic Comparators. In Guy Lebanon and S. V. N. Vishwanathan, editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 398–406, San Diego, California, USA, 09–12 May 2015. PMLR.

[28] Zhihao Jiang, Debmalya Panigrahi, and Kevin Sun. Online algorithms for weighted caching with predictions. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020*, 2020.

[29] T.L Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

[30] Silvio Lattanzi, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Online scheduling via learned weights. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1859–1877. SIAM, 2020.

[31] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 256–261. IEEE Computer Society, 1989.

[32] Haipeng Luo and Robert E. Schapire. Achieving all with no parameters: Adanormalhedge. In Peter Grünwald, Elad Hazan, and Satyen Kale, editors, *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*, volume 40 of *JMLR Workshop and Conference Proceedings*, pages 1286–1304. JMLR.org, 2015.

[33] Thodoris Lykouris and Sergei Vassilvtiskii. Competitive caching with machine learned advice. In *International Conference on Machine Learning*, pages 3302–3311, 2018.

[34] Andres Muñoz Medina and Sergei Vassilvitskii. Revenue optimization with approximate bid predictions. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 1858–1866, 2017.

[35] Michael Mitzenmacher. A model for learned bloom filters and optimizing by sandwiching. In *Advances in Neural Information Processing Systems*, pages 464–473, 2018.

[36] Michael Mitzenmacher. Scheduling with predictions and the price of misprediction. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 14:1–14:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[37] Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ml predictions. In *Advances in Neural Information Processing Systems*, pages 9661–9670, 2018.

[38] Alexander Rakhlin and Karthik Sridharan. Online learning with predictable sequences. In Shai Shalev-Shwartz and Ingo Steinwart, editors, *COLT 2013 - The 26th Annual Conference on Learning Theory, June 12-14, 2013, Princeton University, NJ, USA*, volume 30 of *JMLR Workshop and Conference Proceedings*, pages 993–1019. JMLR.org, 2013.

[39] Dhruv Rohatgi. Near-optimal bounds for online caching with machine learned advice. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1834–1845. SIAM, 2020.

[40] Jacob Steinhardt and Percy Liang. Adaptivity and optimism: An improved exponentiated gradient algorithm. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1593–1601. JMLR.org, 2014.

[41] Chen-Yu Wei and Haipeng Luo. More adaptive algorithms for adversarial bandits. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*, volume 75 of *Proceedings of Machine Learning Research*, pages 1263–1291. PMLR, 2018.

[42] Lijun Zhang, Shiyin Lu, and Zhi-Hua Zhou. Adaptive online learning in dynamic environments. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pages 1323–1333, 2018.

[43] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pages 928–936. AAAI Press, 2003.

## A Missing Proofs from Section 3

*Proof of Lemma 3.* Part (1) is a straightforward computation which uses the fact that $\|h_t\| \le 1$; see Lemma 3.2 of [5].

Part (2) follows from the fact that $\max(\alpha \|c_t\|^2, \langle c_t, h_t\rangle) \le 1$ (because $\alpha, \|c_t\|, \|h_t\|$ are all $\le 1$), and the facts that $r_t \ge 1$ and $\|x\|^2$ is 2-Lipschitz on the unit ball.

To see part (3), we observe that if the hint $h_t$ is good, then the $\max()$ term equals $\langle c_t, h_t\rangle$, and thus (3) follows from the definition of $x_t$ in the algorithm.

If the hint $h_t$ is good, then $z_t = 0$, thus part (4) follows from part (3). If $h_t$ is bad, then the $\max()$ term becomes $\alpha \|c_t\|^2$, thus

$$\begin{aligned}\langle c_t, x_t\rangle - \ell_t(\overline{x}_t) &= \left(\frac{\|\overline{x}_t\|^2 - 1}{2r_t}\right)(\langle c_t, h_t\rangle - \alpha \|c_t\|^2) \\ &= \frac{z_t(1 - \|x_t\|^2)}{2r_t} \le \frac{z_t}{2r_t}.\end{aligned}$$

This completes the proof of part (4). ☐

*Proof of Lemma 4.* First, observe that because of the non-negativity of the $\max()$ term in the definition of $\ell_t$, for all $u \in B(0,1)$, we have $\ell_t(u) \ge \langle c_t, u\rangle$. This means that we can use part (4) of Lemma 3 to conclude that for any round $t$,

$$\langle c_t, x_t - u\rangle \le \ell_t(\overline{x}_t) - \ell_t(u) + \frac{z_t}{2r_t}. \tag{5}$$

We now use the definition of $r_t$ to observe that

$$\frac{z_t}{2r_t} = \frac{z_t}{2\sqrt{1 + z_{1:t-1}}}.$$

Because each $z_t \in [0, 2)$, we have that $2\sqrt{1 + z_{1:t-1}} \ge \sqrt{4 + z_{1:t-1}} > \sqrt{1 + z_{1:t}}$. This lets us bound $\frac{z_t}{2r_t}$ as a telescoping sum.

$$\frac{z_t}{2r_t} < \frac{z_t}{\sqrt{1 + z_{1:t}}} \le 2\left(\sqrt{1 + z_{1:t}} - \sqrt{1 + z_{1:t-1}}\right) = 2(r_{t+1} - r_t).$$

The last equality uses the definition of $r_t$ from the algorithm. Thus, we can sum (5) over $t \in [a, b]$ to obtain the conclusion of the lemma. ☐

## B Missing Proofs from Section 4

### B.1 Proof of Lemma 11

The proof follows along the lines of [32, 22], and additionally incorporates the negative term from Theorem 8.

*Proof.* Consider the following instance of the sleeping experts problem: there are $MT$ experts, parameterized as $(j, t)$, where $j \in [M]$ and $t \in [T]$.[4] The expert $(j, t)$ becomes active in round $t$, and goes back to sleep at round $\gamma(t)$, for a carefully defined function $\gamma$. At all steps $t'$ in which $(j, t)$ is active, we define the loss of this expert to be $\ell_{t',j}$ and the prediction is $m_{t',j}$. The following key properties are ensured for $\eta$:

- $\gamma(t)$ depends only on $t$ (and not on parameters like the time horizon). Indeed, the function used by [22] (attributed to Woodruff) is simply $\gamma(t) = t + 2^{\beta(t)} - 1$, where $2^{\beta(t)}$ is the largest power of 2 dividing $t$.
- At any time $t$, at most $\lceil \log t\rceil$ of the intervals $[t', \gamma(t')]$ for $t' \le t$ contain $t$. (This ensures that only $O(M \log t)$ experts are awake at any time $t$.)
- Finally, *any* interval $[a, b]$ can be covered by a sequence of intervals of the form, $[t_1, \gamma(t_1)], [t_2, \gamma(t_2)], \ldots, [t_r, \gamma(t_r)]$, where $t_1 = a$, $t_i = \gamma(t_{i-1}) + 1$ (i.e., the union of the intervals is a contiguous interval in time), $\gamma(t_r) = b$, and most importantly, $r = O(\log b)$.

[4]We do not need to know $T$ beforehand. As in the procedures of [22], weights of 'incoming' experts can be added as time progresses.

Using $\gamma(t)$ defined above, we run Algorithm 3 with the $MT$ experts. Since only $O(M \log t)$ experts are awake at time $t$, the algorithm can be implemented efficiently — in time $O(M \log T)$ per step. At any step, the algorithm plays one of the awake experts $(j, t')$. Since the losses and predictions only depend on $j$, we define $p_{t,j}$ to be the *total* probability of Algorithm 3 playing an arm of the form $(j, t')$ at time $t$. For convenience, let us also define $\delta_t = \sum_{i \in [M]} p_{t,i} \varepsilon_{t,i}$.

Then, we can apply the guarantee of Theorem 8 to conclude that for any interval $[t', \gamma(t')]$ and for any expert $i$,

$$\sum_{t \in [t', \gamma(t')]} y_t - \ell_{t,i} \le \frac{\log M}{\eta} + \eta \sum_{t \in [t', \gamma(t')]} \left( \varepsilon_{t,i}^2 - \frac{1}{4} \delta_t^2 \right).$$

Thus, for any interval $[a, b] \subseteq [1, T]$, by partitioning $[a, b]$ into $O(\log b)$ intervals as in the third bullet above, we have that for any $i \in [M]$,

$$\sum_{t \in [a,b]} y_t - \ell_{t,i} \le \frac{\log b \log M}{\eta} + \eta \sum_{t \in [a,b]} \left( \varepsilon_{t,i}^2 - \frac{1}{4} \delta_t^2 \right). \tag{6}$$

Thus, if we are to compare with a sequence of experts $i_1, i_2, \dots, i_T$ where the sequence has at most $L$ switches, we can sum up (6) over the $(L + 1)$ intervals that have the same $i_t$ value, and thus the algorithm satisfies

$$\sum_{t=1}^{T} y_t - \ell_{t,i_t} \le \frac{(1 + L) \log T \log M}{\eta} + \eta \sum_{t} \left( \varepsilon_{t,i_t}^2 - \frac{1}{4} \delta_t^2 \right).$$

This completes the proof. Note that the algorithm does not need to know the value of $L$, since we only used (6) which holds for any interval. Thus the guarantee holds simultaneously for all $L$. □

### B.2 Multi-scale multiplicative weights with Hints

**Multiscale MW.** While Lemma 11 yields a switching regret guarantee for any given $\eta$, we need to tune $\eta$ appropriately (specifically, set it to $(L\mathcal{E})^{-1/2}$) to obtain Theorem 9. Naïvely, this is impossible without knowing $L\mathcal{E}$. The following theorem was shown in [12], and it allows one to overcome the so-called *impossible tuning* problem. The idea is to use a combination procedure reminiscent of several prior works. Specifically, it is a variant of the Multiscale Multiplicative Weights (MSMW) procedure of [9] modified to incorporate hints as well as a hybrid update (similar to Algorithm 3).

**Theorem 14.** *Given learning rates $\eta_1, \eta_2, \dots, \eta_M$ and an initial pribability distribution $p_1$ (in the $M$-dimensional probability simplex), there exists an algorithm for the experts problem with hints that has the following guarantee: for all $i \in [M]$,*

$$\sum_{t=1}^{T} y_t \le \sum_{t=1}^{T} \left( \ell_{t,i} + \eta_i \varepsilon_{t,i}^2 \right) + \frac{1}{\eta_i} \log \frac{1}{p_{1,i}} + \sum_{j} \frac{p_{1,j}}{\eta_j},$$

*where $y_t$ is the expected loss incurred by the algorithm at time $t$.*

Lemma 11 then allows us to complete the proof of Theorem 9, as in [12]. The details of this step are presented in Appendix B.3. We now proceed to prove Theorem 14.

**Algorithm (Multi-scale MW with Hints).** The algorithm performs a multiplicative weight style update, but using different learning rates for the different arms. The challenge turns out to be the normalization step (in order to obtain a probability distribution from the weights). [9] introduced a novel approach, by defining a new *projection* step: given some $u \in \mathbb{R}^M$ with nonnegative entries and learning rates $\eta_1, \eta_2, \dots, \eta_M$, first they find a $\lambda \in \mathbb{R}$ such that $\sum_i u_i e^{\eta_i \lambda} = 1$. Then, $\Pi(u)$ is simply the vector whose $i$th coordinate is $u_i e^{\eta_i \lambda}$. As discussed in [9], the operation can be viewed as projection onto the probability simplex with respect to a suitably defined divergence function. Note that if all the $\eta_i$ are equal, $\Pi(u)$'s $i$th coordinate is simply $u_i / (\sum_j u_j)$, which is the classic normalization.

To incorporate hints and obtain the type of bound we desire, we perform a hybrid update analogous to Algorithm 3. Formally, the algorithm is as follows.

- Initialize $p_{0,i}$ using the given values.
- For $t = 1, 2, \dots,$ do the following:

1. set $q'_{t,i} = p_{t,i}e^{-\eta_i m_{t,i}}$ for all $i$
2. define $q_t = \Pi(q'_t)$
3. choose an arm according to $q_t$, receive loss vector $\ell_t$
4. set $p'_{t+1,i} = q'_{t,i}(1 - \eta_i \varepsilon_{t,i})$
5. define $p_{t+1} = \Pi(p'_{t+1})$

We now show a lemma that relates the variation in $p_{t,i}$ values to the losses.

**Lemma 15.** *For any t, define $\lambda_t = \frac{1}{\eta_i}\log\frac{p_{t+1,i}}{p'_{t+1,i}}$ (note that this is independent of i by the definition of the projection). Then we have the following inequalities:*

1. *The expected loss of the algorithm $y_t$ satisfies*
$$y_t \le \lambda_t + \sum_i \frac{p_{t,i} - p_{t+1,i}}{\eta_i}.$$
2. *For any expert i and time t, we have*
$$\lambda_t \le \frac{1}{\eta_i}\log\frac{p_{t+1,i}}{p_{t,i}} + \ell_{t,i} + \eta_i \varepsilon_{t,i}^2.$$

*Proof.* Let us first show part (1). Recall that

$$\lambda_t = \frac{1}{\eta_i}\log\frac{p_{t+1,i}}{p'_{t+1,i}} = \frac{1}{\eta_i}\log\frac{p_{t+1,i}}{q_{t,i}(1-\eta_i\varepsilon_{t,i})} + \frac{1}{\eta_i}\log\frac{q_{t,i}}{q'_{t,i}}. \tag{7}$$

Now, both the terms on the RHS are independent of $i$ (since $q_{t,i}$ is also defined as a projection). This means that for *any* probability distribution $\rho$ over $[M]$, $\frac{1}{\eta_i}\log\frac{q_{t,i}}{q'_{t,i}} = \sum_j \rho_j \frac{1}{\eta_j}\log\frac{q_{t,j}}{q'_{t,j}}$. Choosing $\rho$ to be $q_t$ itself,

$$\frac{1}{\eta_i}\log\frac{q_{t,i}}{q'_{t,i}} = \sum_j \frac{q_{t,j}}{\eta_j}\log\frac{q_{t,j}}{q'_{t,j}} = \sum_j -\frac{q_{t,j}}{\eta_j}\log\frac{q'_{t,j}}{q_{t,j}}.$$

Next, note that $q'_{t,j} = p_{t,j}e^{-\eta_j m_{t,j}}$, and thus

$$\sum_j -\frac{q_{t,j}}{\eta_j}\log\frac{q'_{t,j}}{q_{t,j}} = \sum_j -\frac{q_{t,j}}{\eta_j}\log\frac{p_{t,j}}{q_{t,j}} + \sum_j q_{t,j}m_{t,j}.$$

Next, using the inequality $\log x \le (x-1)$, we obtain

$$\sum_j -\frac{q_{t,j}}{\eta_j}\log\frac{p_{t,j}}{q_{t,j}} \ge \sum_j -\frac{q_{t,j}}{\eta_j}\left(\frac{p_{t,j}}{q_{t,j}} - 1\right) = \sum_j \frac{q_{t,j} - p_{t,j}}{\eta_j}.$$

Similarly, we can manipulate the first term on the RHS of (7). We have

$$\frac{1}{\eta_i}\log\frac{p_{t+1,i}}{q_{t,i}(1-\eta_i\varepsilon_{t,i})} = \sum_j -\frac{p_{t+1,j}}{\eta_j}\log\frac{q_{t,j}(1-\eta_j\varepsilon_{t,j})}{p_{t+1,j}}.$$

Once again, using $\log x \le (x-1)$, we have

$$\frac{1}{\eta_i}\log\frac{p_{t+1,i}}{q_{t,i}(1-\eta_i\varepsilon_{t,i})} \ge \sum_j -\frac{q_{t,j}(1-\eta_j\varepsilon_{t,j}) - p_{t+1,j}}{\eta_j} = \sum_j q_{t,j}\varepsilon_{t,j} + \frac{p_{t+1,j} - q_{t,j}}{\eta_j}.$$

Plugging these back into (7), we have

$$\lambda_t \ge \sum_j q_{t,j}(m_{t,j} + \varepsilon_{t,j}) + \frac{p_{t+1,j} - p_{t,j}}{\eta_j}.$$

Since $\ell_{t,j} = m_{t,j} + \varepsilon_{t,j}$ and the algorithm plays experts using the distribution $q_{t,j}$, this completes the proof of (1).

To see part (2) of the lemma, we simply note that by definition, for any $i$,

$$\lambda_t = \frac{1}{\eta_i} \log \frac{p_{t+1,i}}{p'_{t+1,i}} = \frac{1}{\eta_i} \left( \log \frac{p_{t+1,i}}{p_{t,i}} + \log e^{\eta_i m_{t,i}} - \log(1 - \eta_i \varepsilon_{t,i}) \right).$$

Using the inequality $\log(1-t) \geq -t - t^2$ for $|t| \leq 1/2$, we have

$$\lambda_t \leq \frac{1}{\eta_i} \log \frac{p_{t+1,i}}{p_{t,i}} + m_{t,i} + \varepsilon_{t,i} + \eta_i \varepsilon_{t,i}^2.$$

Noticing that $\ell_{t,i} = m_{t,i} + \varepsilon_{t,i}$, this completes the proof. □

The lemma implies Theorem 14 as follows.

*Proof of Theorem 14.* We can combine the two parts of Lemma 15 to conclude that at any time step $t$ and for any expert $i$,

$$y_t \leq \ell_{t,i} + \eta_i \varepsilon_{t,i}^2 + \frac{1}{\eta_i} \log \frac{p_{t+1,i}}{p_{t,i}} + \sum_{j \in [M]} \frac{p_{t,j} - p_{t+1,j}}{\eta_j}.$$

Summing this over $t$, we get

$$\sum_{t=1}^{T} y_t \leq \sum_{t=1}^{T} (\ell_{t,i} + \eta_i \varepsilon_{t,i}^2) + \frac{1}{\eta_i} \log \frac{p_{T+1,i}}{p_{1,i}} + \sum_{j \in [M]} \frac{p_{1,j} - p_{T+1,j}}{\eta_j}.$$

Since $p_{T+1,i} \in [0,1]$, we have

$$\sum_{t=1}^{T} y_t \leq \sum_{t=1}^{T} (\ell_{t,i} + \eta_i \varepsilon_{t,i}^2) + \frac{1}{\eta_i} \log \frac{1}{p_{1,i}} + \sum_{j \in [M]} \frac{p_{1,j}}{\eta_j}.$$

This completes the proof of the theorem. □

## B.3 Proof of Theorem 9

The proof is overall similar to that of [12], except that we use Lemma 11 as the main subroutine.

*Proof of Theorem 9.* The idea is to run the algorithm from Theorem 14 using experts that correspond to running our "known $\eta$" algorithm (from Lemma 11) with different values of $\eta$. Let us call these the *outer* and *inner* algorithms respectively.

Formally, the parameters of the outer algorithm are as follows: define $\eta_k = \frac{1}{2^k}$, for $k \in \{1, 2, \dots, r\}$, where $r = 2 \log T$. The outer algorithm will be the one from Theorem 14, and will have $r$ experts, with learning rate $\eta_k$ for the $k$th expert. The $k$th expert will correspond to an inner algorithm (from Theorem 11) with $\eta = 4\eta_k$. The initial probabilities will be set as $p_{1,k} = c\eta_k$, where $c = \frac{1}{\eta_1 + \cdots + \eta_r}$. [Note that $c \in [1/2, 1]$.]

We denote by $p_t^{(k)}$ the distribution over $[M]$ produced by the $k$th expert (i.e., the $k$th inner algorithm). The loss incurred by this algorithm is thus $\langle p_t^{(k)}, \ell_t \rangle =: \alpha_{t,k}$. The inner algorithms also receive the prediction vector $m_t$ before playing; we define $\beta_{t,k} := \langle p_t^{(k)}, m_t \rangle$, which we give as the predicted loss for the $k$th expert to the outer algorithm.

From Theorem 14, we have that for every $k$, if $\gamma_t$ denotes the expected loss of the outer algorithm, then for all $k \in [r]$,

$$\sum_t \gamma_t \leq \sum_{t,k} \alpha_{t,k} + \frac{1}{\eta_k} \log \frac{1}{p_{1,k}} + \sum_{j \in [r]} \frac{p_{1,j}}{\eta_j} + \sum_t \eta_k (\alpha_{t,k} - \beta_{t,k})^2. \tag{8}$$

By our choice of parameters, the second term on the RHS is $\leq \frac{2 \log T}{\eta_k}$, and the third term is simply $cr \leq 2c \log T \leq \frac{\log T}{\eta_k}$, since $c \leq 1$ and $\eta_k \geq 1/2$ for all $k$. Now consider the first term. By the regret bound from Lemma 11, we have that for any comparator sequence $i_1, i_2, \dots, i_T$ with at most $L$ switches,

$$\sum_t \alpha_{t,k} \leq \sum_t \ell_{t,i_t} + \frac{(1+L) \log T \log M}{4\eta_k} + 4\eta_k \sum_t \left( \varepsilon_{t,i_t}^2 - \frac{1}{4} \delta_t^2 \right),$$

where $\delta_t$ is the expected prediction error, which is exactly $\langle p_t^{(k)}, \ell_t - m_t \rangle = (\alpha_{t,k} - \beta_{t,k})$. Thus,

$$\sum_t \alpha_{t,k} \le \sum_t \ell_{t,i_t} + \frac{(1+L)\log T \log M}{4\eta_k} + 4\eta_k \mathcal{E} - \eta_k \sum_t (\alpha_{t,k} - \beta_{t,k})^2.$$

(Where $\mathcal{E}$ is as defined in the statement of Theorem 9.) The key observation introduced in [12] is that the last term above exactly cancels out the last term in (8). Thus, plugging all of these bounds into (8),

$$\sum_t \gamma_t \le \ell_{t,i_t} + \frac{(1+L)\log T \log M}{4\eta_k} + 4\eta_k \mathcal{E} + \frac{3\log T}{\eta_k}.$$

By the guarantee of 14, this bound holds for all $k \in [r]$, in particular, setting $\eta_k$ as the inverse of the closest power of 2 to $\sqrt{(1+L)\mathcal{E}\log T \log M}$ (which lies in the range $[1, T^2]$), we obtain the desired result. □

## C Missing Proofs from Section 5

*Proof of Theorem 13.* Recall the lower bound example: we have two arms and $L = 2$. For arm-1, we have $m_{t,1} = \ell_{t,1} = 1/\sqrt{T}$ for all $t$. For arm-2, we set $m_{t,2} = C/\sqrt{T}$ for all $t$, for a parameter $C$ that we will choose later. Then we pick a sub-interval $J$ of length $\frac{T}{C^2}$ uniformly at random in the full interval $[1, T]$, and for all $t \notin J$, we set $\ell_{t,2} = C/\sqrt{T}$ (so the prediction is perfect), and for $t \in J$, we set $\ell_{t,2} = 0$. Note that the total squared prediction error is $\sum_{t \in J} \frac{C^2}{T} = 1$.

Because of the random choice of $J$, the above defines a distribution over instances, parametrized by the start of the interval $J$, which we denote by $t_0$ as in Theorem 12. For any such instance, the optimal two-switch strategy is to play arm-2 for the rounds in $J$ and arm-1 outside of it.

Now consider any deterministic algorithm $\mathcal{A}$ (by Yao's theorem, this can be assumed w.l.o.g.). The predictions $\mathcal{A}$ sees for arm-2 are fixed (and it knows the behavior of arm-1 fully), so it can only get information about $J$ by pulling arm-2. Again, we define $p_t$ to be the binary variable that indicates if arm-2 will be pulled at time $t$, assuming that the algorithm has not observed a 0 cost on arm-2 so far. We consider the following two cases.

*Case 1.* $p_1 + p_2 + \cdots + p_{T/2} \ge \frac{C^2}{8}$.

In this case, in all the instances for which $t_0 \ge T/2$, we end up incurring a cost that is at least $\frac{C^2}{8} \cdot \frac{(C-1)}{\sqrt{T}}$ more than the optimal 2-switch strategy in hindsight (which would have played arm-1 in all rounds $\le T/2$). Now for a random instance in $\mathcal{D}$ we have $t_0 > T/2$ with probability $1/2$, thus the expected regret is $\Omega\left(\frac{C^3}{\sqrt{T}}\right)$.

*Case 2.* $p_1 + p_2 + \cdots + p_{T/2} < \frac{C^2}{8}$.

Here we wish to show that we "miss" the entire interval $J$ with a constant probability. Formally, let $r = T/C^2$, the length of $J$. As before, let $s_i = p_i + p_{i+1} + \cdots + p_{i+r-1}$. Using an argument similar to that in Theorem 12, we have $\sum_{i=1}^{T/2-r} s_i \le r \sum_{i=1}^{T/2} p_i < r \cdot \frac{C^2}{8} = \frac{T}{8}$. Thus, at least $T/3 - r$ of the $s_i$ must be $< 1$ (as otherwise, $T/6$ of the $s_i$ will be $\ge 1$, contradicting the bound on the sum). Now if $t_0 = i$ for any $i$ with $s_i = 0$, the algorithm will end up not pulling arm-2 in the entire interval $[i, i+r-1]$, thus incurring a regret of $\frac{1}{\sqrt{T}} \cdot \frac{T}{C^2}$. Thus, in this case, the expected regret is $\Omega\left(\frac{\sqrt{T}}{C^2}\right)$.

*Setting C.* To get the bounds in the two cases to match, we set $C = T^{1/5}$, resulting in a regret bound of $\Omega(T^{1/10})$, thereby completing the proof of the theorem. □