Strongly Adaptive Online Learning

Amit Daniely AMIT.DANIELY@MAIL.HUJI.AC.IL
Alon Gonen ALONGNN@CS.HUJI.AC.IL
Shai Shalev-Shwartz SHAIS@CS.HUJI.AC.IL
The Hebrew University

Abstract

Strongly adaptive algorithms are algorithms whose performance on *every time interval* is close to optimal. We present a reduction that can transform standard low-regret algorithms to strongly adaptive. As a consequence, we derive simple, yet efficient, strongly adaptive algorithms for a handful of problems.

1. Introduction

Coping with changing environments and rapidly adapting to changes is a key component in many tasks. A broker is highly rewarded from rapidly adjusting to new trends. A reliable routing algorithm must respond quickly to congestion. A web advertiser should adjust himself to new ads and to changes in the taste of its users. A politician can also benefit from quickly adjusting to changes in the public opinion. And the list goes on.

Most current algorithms and theoretical analysis focus on relatively stationary environments. In statistical learning, an algorithm should perform well on the training distribution. Even in online learning, an algorithm should usually compete with the best strategy (from a pool), that is fixed and does not change over time.

Our main focus is to investigate to which extent such algorithms can be modified to cope with changing environments.

We consider a general online learning framework that encompasses various online learning problems including prediction with expert advice, online classification, online convex optimization and more. In this framework, a learning scenario is defined by a decision set D, a context space C and a set $\mathcal{L}$ of real-valued loss functions defined over D. The *learner* sequentially observes a context $c_t \in C$ and then picks a decision $x_t \in D$. Next, a loss function $\ell_t \in \mathcal{L}$ is revealed and the learner suffers a loss $\ell_t(x_t)$.

Often, algorithms in such scenarios are evaluated by comparing their performance to the performance of the best strategy from a pool of strategies (usually, this pool is simply all strategies that play the same action all the time). Concretely, the *regret,* $R_{\mathcal{A}}(T)$, of an algorithm $\mathcal{A}$ is defined as its cumulative loss minus the cumulative loss of the best strategy in the pool. The rationale behind this evaluation metric is that one of the strategies in the pool is reasonably good during the *entire* course of the game. However, when the environment is changing, different strategies will be good in different periods. As we do not want to make any assumption on the duration of each of these periods, we would like to guarantee that our algorithm performs well on *every* interval $I = [q, s] \subset [T]$. Clearly, we cannot hope to have a regret bound which is better than what we have for algorithms that are tested only on I. If this barrier is met, we say that the corresponding algorithm is *strongly adaptive*[1].

Surprisingly maybe, our main result shows that for many learning problems strongly adaptive algorithms exist. Concretely, we show a simple "meta-algorithm" that can use any online algorithm (that was possibly designed to have just small standard regret) as a black box, and produces a new algorithm that is designed to have a small regret on every interval. We show that if the original algorithm have a regret bound of $R(T)$, then the produced algorithm has, on every interval $[q, s]$ of size $\tau := |I|$, regret that is very close to $R(\tau)$ (see a precise statement in Section 1.2). Moreover, the running time of the new algorithm at round t is just $O(\log(t))$ times larger than that of the original algorithm. As an immediate corollary we obtain strongly adaptive algorithms for a handful of online problems including prediction with expert advice, online convex optimization, and more.

[1] See a precise definition in Section 1.1. Also, see Section 1.3 for a weaker notion of *adaptive algorithms* that was studied in (Hazan & Seshadhri, 2007).

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

Furthermore, we show that strong adaptivity is stronger than previously suggested adaptivity properties including the adaptivity notion of (Hazan & Seshadhri, 2007) and the tracking notion of (Herbster & Warmuth, 1998). Namely, strongly adaptive algorithms are also adaptive (in the sense of (Hazan & Seshadhri, 2007)), and have a near optimal tracking regret (in the sense of (Herbster & Warmuth, 1998)). We conclude our discussion by showing that strong adaptivity can not be achieved with bandit feedback.

1.1. Problem setting

A Framework for Online Learning

Many learning problems can be described as a repeated game between the learner and the environment, which we describe below.

A *learning scenario* is determined by a triplet $(D, C, \mathcal{L})$, where D is a *decision space*, C is a set of contexts, and $\mathcal{L}$ is a set of *loss* functions from D to $[0, 1]$. Extending the results to general bounded losses is straightforward. The number of rounds, denoted T, is unknown to the learner. At each time $t \in [T]$, the learner sees a context $c_t \in C$, and then chooses an action $x_t \in D$. Simultaneously, the environment chooses a loss function $\ell_t \in \mathcal{L}$. Then, the action x_t is revealed to the environment, and the loss function ℓ_t is revealed to the learner which suffers the loss $\ell_t(x_t)$. We list below some examples of families of learning scenarios.

- **Learning with expert advice (Cesa-Bianchi et al., 1997).** Here, there is no context (formally, C consists of a single element), D is a finite set of size N (each element in this set corresponds to an expert), and $\mathcal{L}$ consists of all functions from D to $[0, 1]$.
- **Online convex optimization (Zinkevich, 2003).** Here, there is no context as well, D is a convex set, and $\mathcal{L}$ is a collection of convex functions from D to $[0, 1]$.
- **Classification.** Here, C is some set, D is a finite set, and $\mathcal{L}$ consists of all functions from D to $\{0, 1\}$ that are indicators of a single element.
- **Regression.** Here, C is a subset of a Euclidean space, $D = [0, 1]$, and $\mathcal{L}$ consists of all functions of the form $\ell(\hat{y}) = (y - \hat{y})^2$ for $y \in [0, 1]$.

A *learning problem* is a quadruple $\mathcal{P} = (D, C, \mathcal{L}, \mathcal{W})$, where $\mathcal{W}$ is a benchmark of *strategies* that is used to evaluate the performance of algorithms. Here, each strategy $w \in \mathcal{W}$ makes a prediction $x_t(w) \in \mathcal{D}$ based on some rule. We assume that the prediction $x_t(w)$ of each strategy is fully determined by the game's history at the time of the prediction. I.e., by $(c_1, \ell_1), \ldots, (c_{t-1}, \ell_{t-1}), c_t$. Usually, $\mathcal{W}$ consists of very simple strategies. For example, in context-less scenarios (like learning with expert advice and online convex optimization), $\mathcal{W}$ is often identified with D, and the strategy corresponding to $x \in D$ simply predicts x at each step. In contextual problems (such as classification and regression), $\mathcal{W}$ is often a collection of functions from C to D (a hypothesis class), and the prediction of the strategy corresponding to $h : C \to D$ at time t is simply $h(c_t)$.

The cumulative loss of $w \in \mathcal{W}$ at time T is $L_w(T) = \sum_{t=1}^{T} \ell_t(x_t(w))$ and the cumulative loss of an algorithm $\mathcal{A}$ is $L_{\mathcal{A}}(T) = \sum_{t=1}^{T} \ell_t(x_t)$. The cumulative regret of $\mathcal{A}$ is $R_{\mathcal{A}}(T) = L_{\mathcal{A}}(T) - \inf_{w \in \mathcal{W}} L_w(T)$. We define the regret, $R_{\mathcal{P}}(T)$, of the learning problem $\mathcal{P}$ as the minimax regret bound. Namely, $R_{\mathcal{P}}(T)$ is the minimal number for which there exists an algorithm $\mathcal{A}$ such that for every environment $R_{\mathcal{A}}(T) \leq R_{\mathcal{P}}(T)$. We say that an algorithm $\mathcal{A}$ has *low regret* if $R_{\mathcal{A}}(T) = O\left(\text{poly}\left(\log T\right) R_{\mathcal{P}}(T)\right)$ for every environment.

We note that both the learner and the environment can make random decisions. In that case, the quantities defined above refer to the expected value of the corresponding terms.

Strongly Adaptive Regret

Let $I = [q, s] := \{q, q + 1, \ldots, s\} \subseteq [T]$. The loss of $w \in \mathcal{W}$ during the interval I is $L_w(I) = \sum_{t=q}^{s} \ell_t(x_t(w))$ and the loss of an algorithm $\mathcal{A}$ during the interval I is $L_{\mathcal{A}}(I) = \sum_{t=q}^{s} \ell_t(x_t)$. The regret of $\mathcal{A}$ during the interval I is $R_{\mathcal{A}}(I) = L_{\mathcal{A}}(I) - \inf_{w \in \mathcal{W}} L_w(I)$. The *strongly adaptive regret* of $\mathcal{A}$ at time T is the function

$$\text{SA-Regret}_{\mathcal{A}}^{T}(\tau) = \max_{I=[q,q+\tau-1] \subset [T]} R_{\mathcal{A}}(I)$$

We say that $\mathcal{A}$ is *strongly adaptive* if for every environment, $\text{SA-Regret}_{\mathcal{A}}^{T}(\tau) = O\left(\text{poly}\left(\log T\right) \cdot R_{\mathcal{P}}(\tau)\right)$.

1.2. Our Results

A strongly adaptive meta-algorithm

Achieving strongly adaptive regret seems more challenging than ensuring low regret. Nevertheless, we show that often, low-regret algorithms can be transformed into a strongly adaptive algorithms with a little extra computational cost.

Concretely, fix a learning scenario $(D, C, \mathcal{L})$. We derive a strongly adaptive meta-algorithm, that can use any algorithm $\mathcal{B}$ (that presumably have low regret w.r.t. some learning problem) as a black-box. We call our meta-algorithm Strongly Adaptive Online Learner (SAOL). The specific instantiation of SAOL that uses $\mathcal{B}$ as the black box is denoted $\text{SAOL}^{\mathcal{B}}$.

Fix a set $\mathcal{W}$ of strategies and an algorithm $\mathcal{B}$ whose regret

w.r.t. $\mathcal{W}$ satisfies

$$R_{\mathcal{B}}(T) \leq C \cdot T^{\alpha}, \tag{1}$$

where $\alpha \in (0,1)$, and $C > 0$ is some scalar. The properties of SAOL$^{\mathcal{B}}$ are summarized in the theorem below. The description of the algorithm and the proof of Theorem 1 are given in Section 2.

Theorem 1

1. *For every interval $I = [q,s] \subseteq \mathbb{N}$,*

$$R_{SAOL^{\mathcal{B}}}(I) \leq \frac{4}{2^{\alpha}-1} C|I|^{\alpha} + 40\log(s+1)|I|^{\frac{1}{2}} .$$

2. *In particular, if $\alpha \geq \frac{1}{2}$ and $\mathcal{B}$ has low regret, then SAOL$^{\mathcal{B}}$ is strongly adaptive.*

3. *The runtime of SAOL at time t is at most $\log(t+1)$ times the runtime per-iteration of $\mathcal{B}$.*

From part 2, we can derive strongly adaptive algorithms for many online problems. Two examples are outlined below.

- **Prediction with N experts advice.** The Multiplicative Weights (MW) algorithm has regret $\leq 2\sqrt{\ln(N)T}$. Hence, for every $I = [q,s] \subseteq [T]$,

$$R_{\text{SAOL}^{\text{MW}}}(I) = O\left(\left(\sqrt{\log(N)} + \log(s+1)\right)\sqrt{|I|}\right) .$$

- **Online convex optimization with G-Lipschitz loss functions over a convex set $D \subseteq \mathbb{R}^d$ of diameter B.** Online Gradient Descent (OGD) has regret $\leq 3BG\sqrt{T}$. Hence, for every $I = [q,s] \subseteq [T]$,

$$R_{\text{SAOL}^{\text{OGD}}}(I) = O\left((BG + \log(s+1))\sqrt{|I|}\right) .$$

Comparison to (weak) adaptivity and tracking

Several alternative measures for coping with changing environment were proposed in the literature. The two that are most related to our work are *tracking regret* (Herbster & Warmuth, 1998) and *adaptive regret* (Hazan & Seshadhri, 2007) (other notions are briefly discussed in Section 1.3).

Adaptivity, as defined in (Hazan & Seshadhri, 2007), is a weaker requirement than strong adaptivity. The adaptive regret of a learner $\mathcal{A}$ at time T is $\max_{I \subseteq [T]} R_{\mathcal{A}}(I)$. An algorithm is called *adaptive* if its adaptive regret is $O\left(\text{poly}\left(\log T\right) R_{\mathcal{P}}(T)\right)$. For online convex optimization problems for which there exists an algorithm with regret bound $R(T)$, (Hazan & Seshadhri, 2007) derived an efficient algorithm whose adaptive regret is at most $R(T)\log(T) + O\left(\sqrt{T\log^3(T)}\right)$, thus establishing adaptive algorithms for many online convex optimization problems. For the case where the loss functions are α-exp concave, they showed an algorithm with adaptive regret $O(\frac{1}{\alpha}\log^2(T))$ (we note that according to our definition this algorithm is in fact strongly adaptive). A main difference between adaptivity and strong adaptivity, is that in many problems, adaptive algorithms are not guaranteed to perform well on small intervals. For example, for many problems including online convex optimization and learning with expert advice, the best possible adaptive regret is $\Omega(\sqrt{T})$. Such a bound is meaningless for intervals of size $O(\sqrt{T})$. We note that in many scenarios (e.g. routing, paging, news headlines promotion) it is highly desired to perform well even on very small intervals.

The problem of "tracking the best expert" was studied in (Herbster & Warmuth, 1998) (see also, (Bousquet & Warmuth, 2003)). In that problem, originally formulated for the learning with expert advice problem, learning algorithms are compared to all strategies that shift from one expert to another a bounded number of times. They derived an efficient algorithm, named Fixed-Share, which attains near-optimal regret bound of $\sqrt{Tm(\log(T)+\log(N))}$ versus the best strategy that shifts between $\leq m$ experts. (Interestingly, a recent work (Cesa-Bianchi et al., 2012) showed that the Fixed-Share algorithm is in fact (weakly) adaptive). As we show in Section 3, strongly adaptive algorithms enjoy near-optimal tracking regret in the experts problem, and in fact, in many other problems (e.g., online convex optimization). We note that as with (weakly) adaptive algorithms, algorithms with optimal tracking regret are not guaranteed to perform well on small intervals.

Strong adaptivity with bandit feedback

In the so-called bandit setting, the loss functions ℓ_t is not exposed to the learner. Rather, the learner just gets to see the loss, $\ell_t(x_t)$, that he has suffered. In Section 4 we prove that there are no strongly adaptive algorithms that can cope with bandit feedback. Even in the simple experts problem we show that for every $\epsilon > 0$, there is no algorithm whose strongly adaptive regret is $O\left(|I|^{1-\epsilon} \cdot \text{poly}\left(\log T\right)\right)$. Investigating possible alternative notions and/or weaker guarantees in the bandit setting is mostly left for future work.

1.3. Related Work

Maybe the most relevant previous work, from which we borrow many of our techniques is (Blum & Mansour, 2007). They focused on the expert setting and proposed a strengthened notion of regret using time selection functions, which are functions from the time interval $[T]$ to $[0,1]$. The regret of a learner $\mathcal{A}$ with respect

to a time selection function I is defined by $R_{\mathcal{A}}^{I}(T) = \max_{i\in[N]}\left(\sum_{t=1}^{T} I(t)\ell_t(x_t) - \sum_{t=1}^{T} I(t)\ell_t(i)\right)$, where $\ell_t(i)$ is the loss of expert i at time t. This setting can be viewed as a generalization of the sleeping expert setting (Freund et al., 1997). For a fixed set $\mathcal{I}$ consisting of M time selection functions, they proved a regret bound of $O(\sqrt{L_{\min,I}\log(NM)) + \log(NM)})$ with[2] respect to each time selection function $I \in \mathcal{I}$. We observe that if we let $\mathcal{I}$ be the set of all indicator functions of intervals (note that $|\mathcal{I}| = \binom{T}{2} = \Theta(T^2)$), we obtain a strongly adaptive algorithm for learning with expert advice. However, the (multiplicative) computational overhead of our algorithm (w.r.t. the standard MW algorithm) at time t is $\Theta(\log(t))$, whereas the computational overhead of their algorithm is $\Theta(T^2)$. Furthermore, our setting is much more general than the expert setting.

Another related, but somewhat orthogonal line of work (Zinkevich, 2003; Hall & Willett, 2013; Rakhlin & Sridharan, 2013; Jadbabaie et al., 2015) studies *drifting environments*. The focus of those papers is on scenarios where the environment is changing slowly over time.

2. Reducing Adaptive Regret to Standard Regret

In this section we present our strongly adaptive meta-algorithm, named *Strongly Adaptive Online Learner* (SAOL). For the rest of this section we fix a learning scenario $(D, C, \mathcal{L})$ and an algorithm $\mathcal{B}$ that operates in this scenario (think of $\mathcal{B}$ as a low regret algorithm).

We first give a high level description of SAOL. The basic idea is to run an instance of $\mathcal{B}$ on each interval I from an appropriately chosen set of intervals, denoted $\mathcal{I}$. The instance corresponding to I is denoted $\mathcal{B}_I$, and can be thought as an expert that gives his advice for the best action at each time slot in I. The algorithm weights the various $\mathcal{B}_I$'s according to their performance in the past, in a way that instances with better performance get more weight. The exact weighting is a variant of the multiplicative weights rule. At each step, SAOL picks at random one of the $\mathcal{B}_I$'s and follows his advice. The probability of choosing each $\mathcal{B}_I$ is proportional to its weight. Next, we give more details.

The choice of $\mathcal{I}$. As in the MW algorithm, the weighting procedure is used to ensure that SAOL performs optimally for every $I \in \mathcal{I}$. Therefore, the choice of $\mathcal{I}$ exhibits the following tradeoff. On one hand, $\mathcal{I}$ should be large, since we want that optimal performance on intervals in $\mathcal{I}$ will result in an optimal performance on *every interval*. On the other hand, we would like to keep $\mathcal{I}$ small, since running many instances of $\mathcal{B}$ in parallel will result with a large computational cost. To balance these desires, we let

$$\mathcal{I} = \bigcup_{k\in\mathbb{N}\cup\{0\}} \mathcal{I}_k \,,$$

where for all $k \in \mathbb{N}\cup\{0\}$,

$$\mathcal{I}_k = \{[i\cdot 2^k, (i+1)\cdot 2^k - 1] \,:\, i \in \mathbb{N}\}.$$

That is, each $\mathcal{I}_k$ is a partition of $\mathbb{N}\setminus\{1,\ldots,2^k\}$ to consecutive intervals of length 2^k. We denote by

$$\text{ACTIVE}(t) := \{I \in \mathcal{I} \,:\, t \in I\} \,,$$

the set of active intervals at time t. By the definition of $\mathcal{I}_k$, for every $t \le 2^k$ we have that no interval in $\mathcal{I}_k$ contains t, while for every $t > 2^k$ we have that a single interval in $\mathcal{I}_k$ contains t. Therefore,

$$|\text{ACTIVE}(t)| = \lfloor \log(t) \rfloor + 1 \,.$$

It follows that the running time of SAOL at time t is at most $(\log(t)+1)$ times larger than the running time of $\mathcal{B}$. On the other hand, as we show in the proof, we can cover every interval by intervals from $\mathcal{I}$, in a way that will guarantee small regret on the covered interval, provided that we have small regret on the covering intervals.

The weighting method. Let $x_t = x_t(I)$ be the action taken by $\mathcal{B}_I$ at time t. The instantaneous regret of SAOL w.r.t. $\mathcal{B}_I$ at time t is $r_t(I) = \ell_t(x_t) - \ell_t(x_t(I))$. As explained above, SAOL maintains weights over the $\mathcal{B}_I$'s. For $I = [q, s]$, the weight of $\mathcal{B}_I$ at time t is denoted $w_t(I)$. For $t < q$, $\mathcal{B}_I$ is not active yet, so we let $w_t(I) = 0$. At the "entry" time, $t = q$, we set $w_t(I) = \eta_I$ where

$$\eta_I := \min\left\{1/2, 1/\sqrt{|I|}\right\}.$$

The weight at time $t \in (q, s]$ is the previous weight times $(1 + \eta_I \cdot r_{t-1}(I))$. Overall, we have

$$w_t(I) = \begin{cases} 0 & t \notin I \\ \eta_I & t = q \\ w_{t-1}(I)(1 + \eta_I \cdot r_{t-1}(I)) & t \in (q, s] \end{cases} \tag{2}$$

Note that the regret is always between $[-1, 1]$, and $\eta_I \in (0, 1)$, therefore weights are always positive during the lifetime of the corresponding expert. Also, the weight of B_I decreases (increases) if its loss is higher (lower) than the predicted loss.

The overall weight at time t is defined by

$$W_t := \sum_{I\in\mathcal{I}} w_t(I) = \sum_{I\in\text{ACTIVE}(t)} w_t(I).$$

Finally, a probability distribution over the experts at time t is defined by

$$p_t(I) = \frac{w_t(I)}{W_t} \,.$$

[2] where $L_{min,I} = \min_i \sum_{t=1}^{T} I(t)\ell_t(i)$

Note that the probability mass assigned to any inactive instance is zero. The probability distribution p_t determines the action of SAOL at time t. Namely, we have $x_t = x_t(I)$ with probability $p_t(I)$. A pseudo-code of SAOL is detailed in Algorithm 1.

Algorithm 1 Strongly Adaptive Online Learner (with blackbox algorithm $\mathcal{B}$)

Initialize: $w_1(I) = \begin{cases} 1/2 & I = [1,1] \\ 0 & \text{o.w.} \end{cases}$
for $t = 1$ **to** T **do**
  Let $W_t = \sum_{I \in \text{ACTIVE}(t)} w_t(I)$
  Choose $I \in \text{ACTIVE}(t)$ w.p. $p_t(I) = \frac{w_t(I)}{W_t}$
  Predict $x_t(I)$
  Update weights according to Equation (2)
end for

2.1. Proof Sketch of Theorem 1

In this section we sketch the proof of Theorem 1. A full proof is detailed in Appendix 1. The analysis of SAOL is divided into two parts. The first challenge is to prove the theorem for the intervals in $\mathcal{I}$ (see Lemma 2). Then, the theorem should be extended to any interval (end of Appendix 1).

Let us start with the first task. Our first observation is that for every interval I, the regret of SAOL during the interval I is equal to

$$\text{(SAOL's regret relatively to } \mathcal{B}_I + \text{the regret of } \mathcal{B}_I) \quad (3)$$

(during the interval I). Since the regret of $\mathcal{B}_I$ during the interval I is already guaranteed to be small (Equation (1)), the problem of ensuring low regret during each of the intervals in $\mathcal{I}$ is reduced to the problem of ensuring low regret with respect to each of the $\mathcal{B}_I$'s.

We next prove that the regret of SAOL with respect to the $\mathcal{B}_I$'s is small. Our analysis is similar to the proof of (Blum & Mansour, 2007)[Theorem 16]. Both of these proofs are similar to the analysis of the Multiplicative Weights Update (MW) method. The main idea is to define a potential function and relate it both to the loss of the learner and the loss of the best expert.

To this end, we start by defining pseudo-weights over the experts (the B_I's). With a slight abuse of notation, we define $I(t) = \mathbf{1}_{[t \in I]}$. For any $I = [q,s] \in \mathcal{I}$, the pseudo-weight of $\mathcal{B}_I$ is defined by:

$$\tilde{w}_t(I) = \begin{cases} 0 & t < q \\ 1 & t = q \\ \tilde{w}_{t-1}(I) \cdot (1 + \eta_I \cdot r_{t-1}(I)) & q < t \leq s+1 \\ \tilde{w}_s(I) & t > s+1 \end{cases}$$

Note that

$$w_t(I) = \eta_I \cdot I(t) \cdot \tilde{w}_t(I) \ .$$

The potential function we consider is the overall pseudo-weight at time t, $\tilde{W}_t = \sum_{I \in \mathcal{I}} \tilde{w}_t(I)$. The following lemma, whose proof is given in the appendix, is a useful consequence of our definitions.

Lemma 1 *For every $t \geq 1$,*

$$\tilde{W}_t \leq t(\log(t) + 1) \ .$$

Through straightforward calculations, we conclude the proof of Theorem 1 for any interval in $\mathcal{I}$.

Lemma 2 *For every $I = [q,s] \in \mathcal{I}$,*

$$\sum_{t=q}^{s} r_t(I) \leq 5 \log(s+1) \sqrt{|I|} \ .$$

Hence, according to Equation (3),

$$R_{SAOL^{\mathcal{B}}}(I) \leq C \cdot |I|^{\alpha} + 5 \log(s+1) \sqrt{|I|}$$

The proof is given in the appendix.

The extension of the theorem to any interval relies on some useful properties of the set $\mathcal{I}$ (see Lemma 1.1 in the appendix). Roughly speaking, any interval $I \subseteq [T]$ can be partitioned into two sequences of intervals from $\mathcal{I}$, such that the lengths of the intervals in each sequence decay at an exponential rate (Lemma 1.2 in the appendix). The theorem now follows by bounding the regret during the interval I by the sum of the regrets during the intervals in the above two sequences, and by using the fact that the lengths decay exponentially.

3. Strongly Adaptive Regret Is Stronger Than Tracking Regret

In this section we relate the notion of strong adaptivity to that of tracking regret, and show that algorithms with small strongly adaptive regret also have small tracking regret. Let us briefly review the problem of tracking. For simplicity, we focus on context-less learning problems, and on the case where the set of strategies coincides with the decision space (though the result can be straightforwardly generalized). Fix a decision space D and a family $\mathcal{L}$ of loss functions. A compound action is a sequence $\sigma = (\sigma_1, \ldots, \sigma_T) \in D^T$. Since there is no hope in competing w.r.t. all sequences[3], a typical restriction of the problem is to bound the number of switches in each sequence. For a positive integer m,

[3] It is easy to prove a lower bound of order T for this problem

the class of compound actions with at most m switches is defined by

$$B_m = \left\{ \sigma \in D^T : s(\sigma) := \sum_{t=1}^{T-1} \mathbf{1}_{[\sigma_{t+1} \neq \sigma_t]} \leq m \right\} . \quad (4)$$

The notions of loss and regret naturally extend to this setting. For example, the cumulative loss of a compound action $\sigma \in B_m$ is defined by $L_\sigma(T) = \sum_{t=1}^{T} \ell_t(\sigma_t)$. The tracking regret of an algorithm $\mathcal{A}$ w.r.t. the class B_m is defined by

$$\text{Tracking-Regret}_{\mathcal{A}}^{m}(T) = L_{\mathcal{A}}(T) - \inf_{\sigma \in B_m} L_\sigma(T) .$$

The following theorem bounds the tracking regret of algorithms with bounds on the strongly adaptive regret. In particular, of SAOL.

Theorem 2 *Let $\mathcal{A}$ be a learning algorithm with SA-Regret$_{\mathcal{A}}(\tau) \leq C\tau^\alpha$. Then,*

$$\textit{Tracking-Regret}_{\mathcal{A}}^{m}(T) \leq CT^\alpha m^{1-\alpha}$$

Proof Let $\sigma \in B_m$. Let $I_1, \ldots, I_m$ be the intervals that correspond to σ. Clearly, the tracking regret w.r.t. σ is bounded by the sum of the regrets of during the intervals $I_1, \ldots, I_m$. Hence, and using Hölder's inequality, we have

$$\begin{aligned} L_{\mathcal{A}}(T) - L_\sigma(T) &\leq \sum_{i=1}^{m} R_{\mathcal{A}}(I_i) \\ &\leq C \sum_{i=1}^{m} |I_i|^\alpha \\ &\leq C \left(\sum_{i=1}^{m} 1^{\frac{1}{1-\alpha}} \right)^{1-\alpha} \left(\sum_{i=1}^{m} |I_i| \right)^{\alpha} \\ &\leq C m^{1-\alpha} T^\alpha \end{aligned}$$

Recall that for the problem of prediction with expert advice, the strongly adaptive regret of SAOL (with, say, Multiplicative Weights as a black box) is $O\left((\sqrt{\ln(N)} + \log(T))\sqrt{\tau}\right)$. Hence, we obtain a tracking bound of $O\left((\sqrt{\ln(N)} + \log(T))\sqrt{mT}\right)$. Up to a $\sqrt{\log(T)}$ factor, this bound is asymptotically equivalent to the bound of the Fixed-Share Algorithm of (Herbster & Warmuth, 1998)[4]. Also, up to $\log(T)$ factor, the bound is optimal. One advantage of SAOL over Fixed-Share is that SAOL is parameter-free. In particular, SAOL does not need to know[5] m.

[4]For the comparison, we rely on a simplified form of the bound of the Fixed-Share algorithm. This simplified form can be found, for example, in http://web.eecs.umich.edu/~jabernet/eecs598course/web/notes/lec5_091813.pdf

[5]The parameters of Fixed-Share do depend on m

4. Strongly Adaptive Regret in The Bandit Setting

In this section we consider the challenge of achieving adaptivity in the bandit setting. Following our notation, in the bandit setting, only the loss incured by the learner, $\ell_t(x_t)$, is revealed at the end of each round (rather than the loss function, ℓ_t). For many online learning problems for which there exists an efficient low-regret algorithm in the full information model, a simple reduction from the bandit setting to the full information setting (for example, see (Shalev-Shwartz, 2011)[Theorem 4.1]) yields an efficient low-regret bandit algorithm. Furthermore, it is often the case that the dependence of the regret on T is not affected by the lack of information. For example, for the Multi-armed bandit (MAB) problem (Auer et al., 2002) (which is the bandit version of the the problem of prediction with expert advice), the above reduction yields an algorithm with near optimal regret bound of $2\sqrt{TN \log N}$.

A natural question is whether adaptivity can be achieved with bandit feedback. Few positive results are known. For example, applying the aforementioned reduction to the Fixed-Share algorithm results with an efficient bandit learner whose tracking regret is $O\left(\sqrt{Tm(\ln(N) + \ln(T))N}\right)$.

The next theorem shows that with bandit feedback there are no algorithms with non-trivial bounds on the strongly adaptive regret. We focus on the MAB problem with two arms (experts) but it is easy to generalize the result to any nondegenerate online problem. Recall that for this problem we do not have a context, $\mathcal{W} = D = \{e_1, e_2\}$ and $\mathcal{L} = [0,1]^{\mathcal{D}}$.

Theorem 3 *For all $\epsilon > 0$, there is no algorithm for MAB with strongly adaptive regret of $O\left(\tau^{1-\epsilon}\text{poly}\left(\log T\right)\right)$.*

The idea of the proof is simple. Suppose toward a contradiction that $\mathcal{A}$ is an algorithm with strongly adaptive regret of $O\left(\tau^{1-\epsilon}\text{poly}\left(\log T\right)\right)$. This means that the regret of $\mathcal{A}$ on every interval I of length $T^{\frac{\epsilon}{2}}$ is non trivial (i.e. $o(|I|)$). Intuitively, this means that both arms must be inspected at least once during I. Suppose now that one of the arms is always superior to the second (say, has loss zero while the other has loss one). By the above argument, the algorithm will still inspect the bad arm at least once in every $T^{\frac{\epsilon}{2}}$ time slots. Those inspections will result in a regret of $\frac{T}{T^{\frac{\epsilon}{2}}} = T^{1-\frac{\epsilon}{2}}$. This, however, is a contradiction, since the strongly adaptive regret bound implies that the standard regret of $\mathcal{A}$ is $o\left(T^{1-\frac{\epsilon}{2}}\right)$.

This idea is formalized in the following lemma. It implies Theorem 3 as for $\mathcal{A}$ with strongly adaptive regret of $O\left(\tau^{1-\epsilon}\text{poly}\left(\log T\right)\right)$ we can take $k = O\left(T^{1-\frac{\epsilon}{2}}\right)$ and reach a contradiction as the lemma implies that on some

segment I of size $\frac{T}{k} = \Omega\left(T^{\frac{\epsilon}{2}}\right)$, the regret of $\mathcal{A}$ is $\Omega\left(T^{\frac{\epsilon}{2}}\right)$ which grows faster than $|I|^{1-\epsilon}\text{poly}(\log T)$

Lemma 3 *Let $\mathcal{A}$ be an algorithm with regret bounded*

$$R_{\mathcal{A}}(T) \leq k = k(T) \ ,$$

Then, there exists an interval $I \subseteq [T]$ of size $\Omega(T/k)$ with

$$R_{\mathcal{A}}(I) = \Omega(|I|) \ .$$

Proof Assume for simplicity that $4k$ divides T. Consider the environment E^0 , in which $\forall t,\ \ell_t(e_1) = 0.5, \ell_t(e_2) = 1$. Let $U \subset [T]$ be the (possibly random) set of time slots in which the algorithm chooses e_2 when the environment is E^0. Since the regret is at most k, we have $\mathbb{E}[|U|] \leq 2k$. It follows that for some segment $I \subset [T]$ of size $\geq \frac{T}{4k}$ we have $\mathbb{E}[|U \cap I|] \leq \frac{1}{2}$. Indeed, otherwise, if $[T] = I_1 \cup \ldots \cup I_{4k}$ is the partition of the interval $[T]$ into $4k$ disjoint and consecutive intervals of size $\frac{T}{4k}$ we will have $\mathbb{E}[|U|] = \sum_{j=1}^{4k} \mathbb{E}[|U \cap I_j|] > 2k$.

Now, since $|U \cap I|$ is a non-negative integer, w.p. $\geq \frac{1}{2}$ we have $|U \cap I| = 0$. Namely, w.p. $\geq \frac{1}{2}$ $\mathcal{A}$ does not inspect e_2 during the interval I when it runs against E^0. Consider now the environment E that is identical to E^0, besides that $\forall t \in I,\ l_t(e_2) = 0$. By the argument above, w.p. $\geq \frac{1}{2}$, the operation of $\mathcal{A}$ on E is identical to its operation on E^0. In particular, the regret on I when $\mathcal{A}$ plays against E is, w.p. $\geq \frac{1}{2}$, $\frac{|I|}{2}$, and in total, $\geq \frac{1}{2} \cdot \frac{1}{2} \cdot |I|$.

Acknowledgments

We thank Yishay Mansour and Sergiu Hart for helpful discussions. This work is supported by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI). A. Daniely is supported by the Google Europe Fellowship in Learning Theory.

References

Auer, Peter, Cesa-Bianchi, Nicolo, Freund, Yoav, and Schapire, Robert E. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

Blum, Avrim and Mansour, Yishay. From external to internal regret. *Journal of Machine Learning*, 2007.

Bousquet, Olivier and Warmuth, Manfred K. Tracking a small set of experts by mixing past posteriors. *The Journal of Machine Learning Research*, 3:363–396, 2003.

Cesa-Bianchi, Nicolo, Freund, Yoav, Haussler, David, Helmbold, David P, Schapire, Robert E, and Warmuth, Manfred K. How to use expert advice. *Journal of the ACM (JACM)*, 44(3):427–485, 1997.

Cesa-Bianchi, Nicolo, Gaillard, Pierre, Lugosi, Gábor, and Stoltz, Gilles. A new look at shifting regret. *CoRR, abs/1202.3323*, 2012.

Freund, Yoav, Schapire, Robert E, Singer, Yoram, and Warmuth, Manfred K. Using and combining predictors that specialize. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pp. 334–343. ACM, 1997.

Hall, Eric C and Willett, Rebecca M. Online optimization in dynamic environments. *arXiv preprint arXiv:1307.5944*, 2013.

Hazan, Elad and Seshadhri, C. Adaptive algorithms for online decision problems. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 14, 2007.

Herbster, Mark and Warmuth, Manfred K. Tracking the best expert. *Machine Learning*, 32(2):151–178, 1998.

Jadbabaie, Ali, Rakhlin, Alexander, Shahrampour, Shahin, and Sridharan, Karthik. Online optimization: Competing with dynamic comparators. *arXiv preprint arXiv:1501.06225*, 2015.

Rakhlin, Sasha and Sridharan, Karthik. Optimization, learning, and games with predictable sequences. In *Advances in Neural Information Processing Systems*, pp. 3066–3074, 2013.

Shalev-Shwartz, Shai. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

Zinkevich, Martin. Online convex programming and generalized infinitesimal gradient ascent. 2003.