CSCI 659 Lecture 5

Spring 2026

Instructor: Haipeng Luo

1 Stronger Regret Measures for Non-Stationary Environments

When introducing the classical static regret definition in Lecture 1, we mentioned that competing with a fixed optimal action in hindsight might not always make sense, especially when the environment is non-stationary such that there is no single fixed action that performs well overall. To see this, consider a simple expert problem with two experts where the first one always suffers loss 0 for the first $T/2$ rounds and loss 1 for the remaining $T/2$ rounds, and the situation for the second expert is exactly reversed. Then, either expert is the overall best one, but with a huge total loss of $T/2$. In this case, even if we have zero regret against the best fixed expert, all we can say is that the loss of our algorithm is bounded by $T/2$, a very disappointing guarantee.

Moreover, this might not be just due to loose regret analysis, and one can show that some algorithms with sublinear regret guarantees indeed suffer linear loss $\Omega(T)$ is this instance. Take Hedge as an example. Observe that by the algorithm's definition, the weight for the first expert is always not smaller than the second one (since its cumulative loss is always not larger). This means that for the second half of the rounds, the loss of the algorithm is at least $1/2$ each round, and thus the cumulative loss of Hedge is at least $T/4$.

How do we address such issues? From an algorithmic perspective, intuitively we need algorithms that can realize the changes in the environment quickly and gradually forget about the outdated data in the past. To guide such algorithm design, we consider the following two stronger regret measures.

Interval regret. For two integers s (staring point) and e (end point) such that $1 \leq s \leq e \leq T$, we use the notation $\mathcal{I} = [s, e]$ to denote the rounds $\{s, s+1, \ldots, e-1, e\}$ and call it an interval. The interval regret with respect to an interval $\mathcal{I}$ is then simply defined as the standard regret restricted on this interval:

$$\mathcal{R}_{\mathcal{I}} = \sum_{t \in \mathcal{I}} f_t(w_t) - \min_{w \in \Omega} \sum_{t \in \mathcal{I}} f_t(w).$$

In other words, $\mathcal{R}_{\mathcal{I}}$ compares the loss of the algorithm on interval $\mathcal{I}$ to the loss of the best fixed point on the same interval. If we know where the starting point s of the interval $\mathcal{I}$ is, then we would simply run a standard algorithm starting from round s and obtain $\mathcal{R}_{\mathcal{I}} = \mathcal{O}(\sqrt{|\mathcal{I}|})$ (omitting other dependence), where we use $|\mathcal{I}|$ to denote the length of interval $\mathcal{I}$ (that is, $e - s + 1$). Of course, the challenge is that we do not know what $\mathcal{I}$ is beforehand, or in other words, we want to design an algorithm with interval regret $\mathcal{R}_{\mathcal{I}} = \mathcal{O}(\sqrt{|\mathcal{I}|})$ *simultaneously* for all $\mathcal{I}$. In the literature, such an algorithm is also sometimes called a *strongly adaptive* algorithm. This is clearly a stronger measure compared to the standard static regret where we only care about $\mathcal{I} = [1, T]$.

Going back to the example discussed earlier. If we have a strongly adaptive algorithm, then we can conclude that the interval regret for the first $T/2$ rounds and the remaining $T/2$ rounds are both of order $\mathcal{O}(\sqrt{T})$. Since the best experts on these two intervals (expert 1 and 2 respectively) both have zero cumulative loss on their respective interval, the cumulative loss of such a strongly adaptive algorithm over T rounds is only $\mathcal{O}(\sqrt{T})$, much better than the $\Omega(T)$ loss suffered by Hedge.

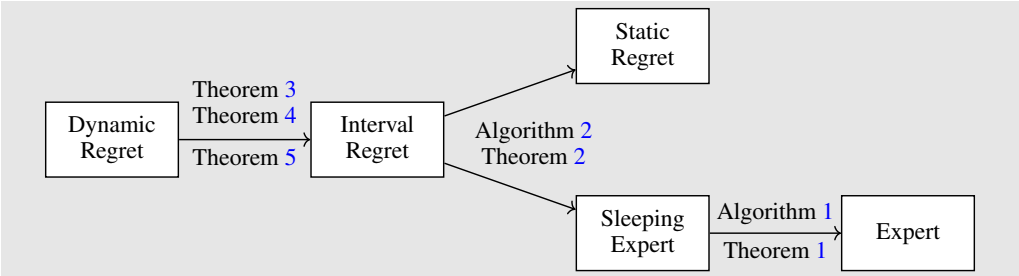
Dynamic regret. While interval regret considers the performance of the algorithm in a local region, dynamic regret measures the global performance of the algorithm over T rounds when compared to a sequence of changing actions. Specifically, the dynamic regret with respect to a comparator sequence $u_1, \dots, u_T \in \Omega$ is defined as

$$\mathcal{R}_T(u_1, \dots, u_T) = \sum_{t=1}^{T} (f_t(w_t) - f_t(u_t)).$$

When $u_1 = \cdots = u_T = u$, this clearly recovers the standard static regret. On the other hand, the most ambitious goal would be to compare with the optimal comparator sequence $w_1^\star, \dots, w_T^\star$ where $w_t^\star = \operatorname{argmin}_{w \in \Omega} f_t(w)$ is the optimal action at time t. However, it is not difficult to see that sublinear dynamic regret against such an optimal comparator sequence is impossible in general. To see this, simply consider a 2-expert problem where the environment always assigns loss 0 to the expert that has the larger weight from the algorithm and loss 1 to the other expert. Then at each round, clearly the best expert has loss 0, while the algorithm suffers loss at least $1/2$, leading to $\Omega(T)$ dynamic regret.

Therefore, the objective here is to express the dynamic regret in terms of a certain "complexity" measure of either the comparator sequence or the loss function sequence, such that the algorithm enjoys sublinear dynamic regret whenever such complexity is not too large. We defer the concrete discussion to Section 4.

A reduction roadmap. We will take a reduction approach to derive algorithms with interval regret or dynamic regret guarantees. Specifically, we will reduce the problem of getting dynamic regret bounds to the problem of getting interval regret bounds, and then further reduce the latter to getting static regret bounds plus solving a so-called *sleeping experts* problem, which is finally reduced to solving the standard expert problem. This reduction chain is illustrated in the following picture. It again demonstrates the fundamental role of the expert problem as well as the static regret (even if its definition seems questionable at first glance).



2 Sleeping Experts

We start with a detour and discuss the sleeping expert problem, a variant of the expert problem introduced by Freund et al. [1997] that is itself interesting and useful. In this variant, each expert comes with different expertise and thus can choose to abstain from providing any advice for a given round if they do not feel confident enough. Formally, at round $t = 1, \dots, T$,

1. the environment first decides (possibly adversarially) which of the N experts are *awake* and which are *asleep*: $a_t(i) = 1$ means expert i is awake and $a_t(i) = 0$ means it is asleep;
2. $a_t \in \{0, 1\}^N$ is then revealed to the learner who needs to decide a distribution $p_t \in \Delta(N)$ with the restriction that no weights are put on asleep experts, that is, $p_t(i) = 0$ if $a_t(i) = 0$;
3. the environment decides and reveals the loss for the awake experts, that is, $\ell_t(i) \in [0, 1]$ for all i such that $a_t(i) = 1$.

The regular expert problem is clearly a special case with $a_t(i) = 1$ for all t and i. Because an expert i is not necessarily involved in every round of the game, the regret against this expert, denoted by $R_T(i)$, is now naturally defined only in terms of those rounds when the expert is awake:

$$R_T(i) = \sum_{t \le T : a_t(i) = 1} \langle p_t - e_i, \ell_t \rangle .$$

Note that while we use the notation $\ell_t \in [0,1]^N$, some of its coordinates might not be defined since the corresponding experts are asleep for this round. However, this is not really an issue because p_t is required to put zero weight on those coordinates anyway and thus they make no difference to the inner product $\langle p_t, \ell_t \rangle$.

It is natural to ask for a sleeping expert algorithm with $R_T(i) = \mathcal{O}(\sqrt{|\{t : a_t(i) = 1\}| \ln N})$ for all i. Indeed, this is achievable by reducing the sleeping expert problem to the regular expert problem. Specifically, suppose we are given an regular expert algorithm $\mathcal{E}$ with prediction $\widehat{p}_t$ on round t. To come up with a prediction p_t for the sleeping expert problem so that $p_t(i) = 0$ for all i with $a_t(i) = 0$, a natural idea is to simply ignore the weights in $\widehat{p}_t$ for these asleep experts and renormalize the others, that is, $p_t(i) \propto a_t(i)\widehat{p}_t(i)$.

Next, we need to come up with a loss vector as the feedback to $\mathcal{E}$. For the awake experts, it is natural to just use the same loss from the sleeping expert problem. What about the asleep experts? Suppose that we assign the same value x to all these asleep experts. Then with the goal of forcing the loss of $\mathcal{E}$ for this round to be the same as the loss of the sleeping expert algorithm, we arrive at an equation

$$\sum_{i:a_t(i)=1} \widehat{p}_t(i)\ell_t(i) + \left(\sum_{i:a_t(i)=0} \widehat{p}_t(i)\right) x = \sum_{i:a_t(i)=1} p_t(i)\ell_t(i).$$

Using the definition of p_t and solving for x gives

$$\begin{aligned} x &= \frac{\sum_{i:a_t(i)=1} \left(p_t(i) - \widehat{p}_t(i)\right) \ell_t(i)}{\sum_{i:a_t(i)=0} \widehat{p}_t(i)} = \frac{\left(\frac{1}{\sum_{i:a_t(i)=1} \widehat{p}_t(i)} - 1\right) \sum_{i:a_t(i)=1} \widehat{p}_t(i)\ell_t(i)}{1 - \sum_{i:a_t(i)=1} \widehat{p}_t(i)} \\ &= \frac{1}{\sum_{i:a_t(i)=1} \widehat{p}_t(i)} \sum_{i:a_t(i)=1} \widehat{p}_t(i)\ell_t(i) = \sum_{i:a_t(i)=1} p_t(i)\ell_t(i), \end{aligned}$$

showing that the "fake" loss of asleep experts should be exactly the loss of the sleeping expert algorithm! As discussed earlier, the value of $\ell_t(i)$ for $a_t(i) = 0$ does not matter in the sleeping expert problem. We will therefore overload the notation ℓ_t to denote the loss vector for both the regular expert problem and the sleeping expert problem, where $\ell_t(i) = \sum_{j:a_t(j)=1} p_t(j)\ell_t(j) = \langle p_t, \ell_t \rangle$ if $a_t(i) = 0$, and otherwise is the loss revealed by the environment. The complete reduction is shown in the following algorithm, followed by its formal guarantee.

Algorithm 1: Reduction from Sleeping Experts to Regular Experts

Input: a regular expert algorithm $\mathcal{E}$
for $t = 1, \ldots, T$ **do**
  let $\widehat{p}_t \in \Delta(N)$ be the decision of $\mathcal{E}$ at round t
  observe a_t from the environment
  play p_t such that $p_t(i) \propto a_t(i)\widehat{p}_t(i)$
  observe $\ell_t(i)$ for all i such that $a_t(i) = 1$
  set $\ell_t(i) = \langle p_t, \ell_t \rangle$ for all i such that $a_t(i) = 0$
  feed ℓ_t to $\mathcal{E}$

Theorem 1. *If the regular expert algorithm $\mathcal{E}$ enjoys the optimal regret bound $\sum_{t=1}^T \langle \widehat{p}_t - e_i, \ell_t \rangle = \mathcal{O}(\sqrt{T \ln N})$ for all i and T, then Algorithm 1 ensures $R_T(i) = \sum_{t:a_t(i)=1} \langle p_t - e_i, \ell_t \rangle = \mathcal{O}(\sqrt{T \ln N})$ for all i and T as well. On the other hand, if $\mathcal{E}$ satisfies an adaptive regret bound*

$$\sum_{t=1}^T \langle \widehat{p}_t - e_i, \ell_t \rangle = \mathcal{O}\left(\sqrt{\sum_{t=1}^T \langle \widehat{p}_t - e_i, \ell_t \rangle^2 \ln N}\right) \tag{1}$$

for all i and T, then Algorithm 1 ensures for all i and T:

$$R_T(i) = \mathcal{O}(\sqrt{|\{t : a_t(i) = 1\}| \ln N}). \tag{2}$$

Proof. The design of Algorithm 1 ensures $\langle p_t, \ell_t \rangle = \langle \widehat{p}_t, \ell_t \rangle$ for all t and also $\langle p_t - e_i, \ell_t \rangle = 0$ for t with $a_t(i) = 0$. Therefore, we can connect the regret in these two problems as:

$$R_T(i) = \sum_{t:a_t(i)=1} \langle p_t - e_i, \ell_t \rangle = \sum_{t=1}^{T} \langle p_t - e_i, \ell_t \rangle = \sum_{t=1}^{T} \langle \widehat{p}_t - e_i, \ell_t \rangle .$$

The first statement of the theorem thus follows immediately. The second statement is also clear after realizing

$$\sum_{t=1}^{T} \langle \widehat{p}_t - e_i, \ell_t \rangle^2 = \sum_{t=1}^{T} \langle p_t - e_i, \ell_t \rangle^2 = \sum_{t:a_t(i)=1} \langle p_t - e_i, \ell_t \rangle^2 \le |\{t : a_t(i) = 1\}|,$$

where the inequality uses the fact $\langle p_t - e_i, \ell_t \rangle \in [-1, 1]$. □

This shows that to obtain the ideal regret guarantee for the sleeping expert problem, we need slightly more than a minimax optimal expert algorithm in the reduction, that is, we need an adaptive regret bound (1). Note that we indeed have such adaptive algorithms — in the last lecture, we discussed one already, and in fact, there are several more in the literature. Recall that we also discussed two interesting consequences of (1) last time: small-loss bound and almost constant regret for a stochastic setting. One can verify that the same consequences also hold for $R_T(i)$ in this sleeping expert problem.

3 From Sleeping Experts to Interval Regret

We are now ready to introduce a general mechanism to derive strongly adaptive algorithms (that is, $\mathcal{R}_{\mathcal{I}} = \mathcal{O}(\sqrt{|\mathcal{I}|})$ or at least $\mathcal{R}_{\mathcal{I}} = \widetilde{\mathcal{O}}(\sqrt{|\mathcal{I}|})$ for any interval $\mathcal{I}$) in the OCO setting,[1] given any OCO algorithm $\mathcal{A}$ with static regret $\mathcal{O}(\sqrt{T})$ for all T. As mentioned earlier, if the interval $\mathcal{I} = [s, e]$ was known, or in fact only the starting point s was known, then one could simply run $\mathcal{A}$ starting from round s. Since we now want to consider all different starting points, a natural idea is to start a new instance of $\mathcal{A}$ at the beginning of every round, and to combine the actions selected by these different instances to come up with the final action.

This idea can be exactly captured by the sleeping expert problem: each instance of $\mathcal{A}$ is an expert, and the instance that starts at time t (denoted by $\mathcal{A}^t$) is asleep for the first $t-1$ rounds and awake for the rest of the problem. At time t, all the t awake instances $\mathcal{A}^1, \dots, \mathcal{A}^t$ propose their action, denoted by $w_t^1, w_t^2, \dots, w_t^t$ respectively. Then, we use a sleeping expert algorithm to come up with a distribution p_t over these awake experts, and finally select the action that is the average of $w_t^1, w_t^2, \dots, w_t^t$ weighted by $p_t(1), \dots, p_t(t)$, that is, their convex combination with respect to p_t (one can also randomly follow one of these actions according to p_t). At the end of round t, the loss for expert $\mathcal{A}^i$ ($i \le t$) is naturally $f_t(w_t^i)$, the loss of the action proposed by $\mathcal{A}^i$. The full reduction is summarized below, followed by its formal guarantee.

Algorithm 2: Strongly Adaptive Algorithm via Sleeping Expert

Input: an OCO algorithm $\mathcal{A}$ with a static regret guarantee, a sleeping expert algorithm $\mathcal{S}$
for $t = 1, \dots, T$ **do**
- start a new instance of $\mathcal{A}$, called $\mathcal{A}^t$
- obtain actions $w_t^1, \dots, w_t^t \in \Omega$ proposed by $\mathcal{A}^1, \dots, \mathcal{A}^t$ respectively
- feed a_t to $\mathcal{S}$ where $a_t(i) = 1$ for $i \le t$ and $a_t(i) = 0$ for $i > t$
- obtain distribution p_t from $\mathcal{S}$
- select final action $w_t = \sum_{i=1}^{t} p_t(i) w_t^i$
- observe loss function f_t and suffer loss $f_t(w_t)$
- feed f_t to $\mathcal{A}^1, \dots, \mathcal{A}^t$
- feed ℓ_t to $\mathcal{S}$ where $\ell_t(i) = f_t(w_t^i)$ for $i \le t$.

[1] Recall that the notation $\widetilde{\mathcal{O}}(\cdot)$ hides all the factors that are logarithmic or polylogarithmic in T.

Theorem 2. *If the OCO algorithm $\mathcal{A}$ ensures $\mathcal{R}_T = \mathcal{O}(\sqrt{T})$ for all T and the sleeping expert algorithm $\mathcal{S}$ ensures Eq. (2), then Algorithm 2 ensures $\mathcal{R}_{\mathcal{I}} = \mathcal{O}(\sqrt{|\mathcal{I}| \ln T})$ for all interval $\mathcal{I}$.*

Proof. By the construction, we have for any $u \in \Omega$ and interval $\mathcal{I} = [s, e]$,

$$\begin{aligned}
\sum_{t \in \mathcal{I}} (f_t(w_t) - f_t(u)) &\leq \sum_{t \in \mathcal{I}} \left(\sum_{i=1}^{t} p_t(i) f_t(w_t^i) - f_t(u) \right) && \text{(Jensen's inequality)} \\
&= \sum_{t \in \mathcal{I}} (\langle p_t, \ell_t \rangle - \ell_t(s)) + \sum_{t \in \mathcal{I}} (f_t(w_t^s) - f_t(u)) \\
&= R_e(s) + \mathcal{O}(\sqrt{|\mathcal{I}|}) && \text{(by the guarantee of } \mathcal{A}^s) \\
&= \mathcal{O}(\sqrt{|\mathcal{I}| \ln T}) + \mathcal{O}(\sqrt{|\mathcal{I}|}), && \text{(by the guarantee of } \mathcal{S})
\end{aligned}$$

completing the proof. □

We have thus shown that a strongly adaptive algorithm can be constructed via a standard algorithm and another sleeping expert algorithm, with only a small price of an additional $\sqrt{\ln T}$ factor in the regret (that is known to be necessary). Note that the improved sleeping expert guarantee Eq. (2) plays a key role here; indeed, if one only has the weaker guarantee stated in the first part of Theorem 1, then Algorithm 2 only guarantees $\mathcal{R}_{\mathcal{I}} = \mathcal{O}(\sqrt{T \ln T})$ for all $\mathcal{I}$ (known as *weakly adaptive*). Also, the logarithmic dependence on the number of experts is clearly critical here as well, since the total number of experts is T in this case.

Computational efficiency. While for regret we only pay logarithmic dependence on the number of experts, the running time of $\mathcal{S}$ does still depend linearly on this number, an issue that is fundamental for the expert problem as we discussed before. Although the number of experts is only T (instead of something exponentially large as in the combinatorial example discussed in Lecture 2), this still makes the algorithm much less practical and also destroys the efficiency of most OCO algorithms whose update time per round is independent of T.

Fortunately, it turns out that we can improve the running time to $\mathcal{O}(\ln T)$ per round, without sacrificing anything for the regret. The idea is to kill some instances of $\mathcal{A}$ when they have lived long enough in some sense so that at each time there are only $\mathcal{O}(\ln T)$ instances alive. Killing an instance can be easily incorporated into the sleeping expert model by just putting the corresponding expert to sleep forever. The key is to do this in a careful way so that the regret is almost not affected, and there are in fact several different ways to do so; see for example [Hazan and Seshadhri, 2007].

4 From Interval Regret to Dynamic Regret

Finally, we discuss how an interval regret guarantee automatically implies several different dynamic regret guarantees. Recall that for dynamic regret $\mathcal{R}_T(u_1, \ldots, u_T) = \sum_{t=1}^{T} (f_t(w_t) - f_t(u_t))$, our goal is to express it in terms of some complexity measure of either the comparator sequence $u_1, \ldots, u_T$, or the loss function sequence $f_1, \ldots, f_T$. We discuss three such examples below.

Switching/tracking regret. The simplest case is to measure the complexity of $u_1, \ldots, u_T$ by how many times it switches from one action to another, in which case dynamic regret is also called switching regret or tracking regret. Formally, let $S \geq 1$ be such that $\sum_{t=2}^{T} \mathbf{1}\{u_t \neq u_{t-1}\} = S - 1$, that is, the number of switches plus one. In other words, we allow the benchmark to divide the total T rounds into S disjoint intervals, and to select any fixed action on each interval. Based on this interpretation, we immediately obtain the following result.

Theorem 3. *If an algorithm ensures $\mathcal{R}_{\mathcal{I}} = \widetilde{\mathcal{O}}(\sqrt{|\mathcal{I}|})$ for all interval $\mathcal{I}$, then it also ensures $\mathcal{R}_T(u_1, \ldots, u_T) = \widetilde{\mathcal{O}}(\sqrt{ST})$ for any $S \in [T]$ and any comparator sequence with $S - 1$ switches.*

Proof. Such a comparator sequence naturally divides the T rounds into S intervals $\mathcal{I}_1, \ldots, \mathcal{I}_S$ such that the comparator stays the same on each interval. Then, we simply apply the interval regret

guarantee on each interval:

$$\mathcal{R}_T(u_1,\dots,u_T) = \sum_{m=1}^{S}\sum_{t\in\mathcal{I}_m}(f_t(w_t)-f_t(u_t)) = \widetilde{\mathcal{O}}\left(\sum_{m=1}^{S}\sqrt{|\mathcal{I}_m|}\right)$$
$$\le \widetilde{\mathcal{O}}\left(\sqrt{S\sum_{m=1}^{S}|\mathcal{I}_m|}\right) = \widetilde{\mathcal{O}}(\sqrt{ST})$$

where the inequality is by Cauchy-Schwarz inequality. □

Therefore, as long as S is sublinear in T, the switching regret is also sublinear. Such a low-switching benchmark can be significantly better than a no-switching benchmark — just consider the 2-expert example discussed in Section 1 again, where one switch already leads to $\Omega(T)$ difference in the total loss of the benchmark. Generally, switching regret is suitable for environments that are close to being piecewise stationary, and there are indeed many such real-life examples. For instance, in the context of a recommendation system, users' preferences usually exhibit a certain stationary trend within a period (say a month or a season).

Question 1. *If the interval regret guarantee is instead $\mathcal{R}_\mathcal{I} = \widetilde{\mathcal{O}}(\sqrt{T})$ for all interval $\mathcal{I}$, what would be the corresponding switching regret guarantee?*

Variation of the loss sequence. In the second example, we allow the competitor sequence to be arbitrary (so we might just as well pick $u_t = w_t^\star = \min_w f_t(w)$, leading to the strongest benchmark), but measure the complexity using the variation of the loss function sequence $f_1,\dots,f_T$, defined as

$$V_T = \sum_{t=2}^{T}\max_{w\in\Omega}|f_t(w)-f_{t-1}(w)|,$$

that is, the cumulative differences between two consecutive loss functions (measured by their largest difference at the same point). A small variation implies that the environment is slowly drifting over time, making learning possible even if we want to compete with the best action at each round. This measure is quite similar to the path length of the loss sequence discussed in the last two lectures: previously, our goal was to derive a static regret bound in terms of the path length, with the hope of getting an $o(\sqrt{T})$ bound whenever the environment is slowly changing, and now our goal is to derive a dynamic regret bound in terms of the variation, with the goal of getting an $o(T)$ bound whenever the environment is slowly changing.

Interestingly, this problem is again already solved by using a strongly adaptive algorithm.

Theorem 4. *If an algorithm ensures $\mathcal{R}_\mathcal{I} = \widetilde{\mathcal{O}}(\sqrt{|\mathcal{I}|})$ for all interval $\mathcal{I}$, then it also ensures*

$$\mathcal{R}_T(w_1^\star,\dots,w_T^\star) = \widetilde{\mathcal{O}}\left(\sqrt{T}+T^{2/3}V_T^{1/3}\right)$$

where $w_t^\star = \operatorname{argmin}_{w\in\Omega} f_t(w)$.

Proof. Let $\mathcal{I}_1 = [s_1,e_1],\dots,\mathcal{I}_M = [s_M,e_M]$ be any partition of the entire T rounds, and

$$V_{\mathcal{I}_m} = \sum_{t=s_m+1}^{e_m}\max_{w\in\Omega}|f_t(w)-f_{t-1}(w)|$$

be the variation of interval $\mathcal{I}_m$. For any $m\in[M]$, we can bound the regret on this interval as:

$$\sum_{t\in\mathcal{I}_m}(f_t(w_t)-f_t(w_t^\star)) = \sum_{t\in\mathcal{I}_m}(f_t(w_t)-f_t(w_{s_m}^\star)) + \sum_{t\in\mathcal{I}_m}(f_t(w_{s_m}^\star)-f_t(w_t^\star))$$
$$\le \widetilde{\mathcal{O}}(\sqrt{|\mathcal{I}_m|}) + 2|\mathcal{I}_m|V_{\mathcal{I}_m},$$

where the last step is by the interval regret guarantee of the algorithm and the fact

$$f_t(w_{s_m}^\star)-f_t(w_t^\star) \le f_t(w_{s_m}^\star)-f_{s_m}(w_{s_m}^\star)+f_{s_m}(w_t^\star)-f_t(w_t^\star) \qquad \text{(by optimality of } w_{s_m}^\star\text{)}$$

$$= \sum_{\tau=s_m+1}^{t} \left(f_\tau(w^\star_{s_m}) - f_{\tau-1}(w^\star_{s_m})\right) + \sum_{\tau=s_m+1}^{t} \left(f_{\tau-1}(w^\star_t) - f_\tau(w^\star_t)\right)$$
$$\le 2 \sum_{\tau=s_m+1}^{t} \max_{w\in\Omega} |f_\tau(w) - f_{\tau-1}(w)| \le 2V_{\mathcal{I}_m}.$$

Therefore, the overall dynamic regret is bounded by

$$\begin{aligned}
\mathcal{R}_T(w^\star_1, \dots, w^\star_T) &\le \sum_{m=1}^{M} \widetilde{\mathcal{O}}(\sqrt{|\mathcal{I}_m|}) + 2|\mathcal{I}_m| V_{\mathcal{I}_m} \\
&\le \widetilde{\mathcal{O}}(\sqrt{MT}) + 2 \max_m |\mathcal{I}_m| \sum_{m=1}^{M} V_{\mathcal{I}_m} \qquad \text{(Cauchy-Schwarz inequality)} \\
&\le \widetilde{\mathcal{O}}(\sqrt{MT}) + 2 \max_m |\mathcal{I}_m| V_T.
\end{aligned}$$

Finally we just need to balance M and $\max_m |\mathcal{I}_m|$. For a fixed M, it is clear that if we want to minimize $\max_m |\mathcal{I}_m|$, we should divide the entire T rounds (almost) evenly into M intervals so that $\max_m |\mathcal{I}_m| = \mathcal{O}(T/M)$ and $\mathcal{R}_T(w^\star_1, \dots, w^\star_T) = \widetilde{\mathcal{O}}(\sqrt{MT} + TV_T/M)$. Setting M (optimally) to $\max\{1, \lfloor T^{1/3} V_T^{2/3} \rfloor\}$ finishes the proof (note that M only appears in the analysis). □

Question 2. *Once again, if the interval regret guarantee is instead $\mathcal{R}_\mathcal{I} = \widetilde{\mathcal{O}}(\sqrt{T})$ for all interval $\mathcal{I}$, what would be the corresponding dynamic regret guarantee?*

Therefore, as long as V_T is sublinear, the dynamic regret is sublinear. When V_T is linear, the bound is vacuous, but this is consistent with our earlier discussion that it is impossible to always enjoy $o(T)$ dynamic regret. We make two more remarks on this dynamic regret bound:

First, notice that if f_t is completely revealed at the end of each round, then there is in fact a trivial algorithm with a better dynamic regret bound: simply pick w_t as $w^\star_{t-1}$, that is, best respond to the most recently observed loss function. Its dynamic regret is:

$$\sum_{t=1}^{T} (f_t(w^\star_{t-1}) - f_t(w^\star_t)) \le \sum_{t=1}^{T} (f_t(w^\star_{t-1}) - f_{t-1}(w^\star_{t-1}) + f_{t-1}(w^\star_t) - f_t(w^\star_t)) \le 2V_T,$$

which is even better than $\widetilde{\mathcal{O}}(T^{2/3} V_T^{1/3})$. However, this naive approach heavily relies on the fact that f_t is completely revealed, while the algorithms we have been discussing only need the gradient $\nabla f_t(w_t)$ (see Lecture 2). In fact, even if we only observe a noisy version of $\nabla f_t(w_t)$ with a bounded variance, these algorithms still work as well, while it is easy to see that a small amount of noise can completely destroy the naive approach. Indeed, with the presence of noise, the bound stated in Theorem 4 is worst-case optimal as shown in [Besbes et al., 2015]. Even ignoring this issue, the naive approach is also bad in the sense that it has no static regret guarantee at all.

Second, the switching regret bound from Theorem 3 and the dynamic regret bound from Theorem 4 are generally not comparable. However, the key is that a strongly adaptive algorithm *simultaneously* enjoys both bounds. In fact, a simple rewriting of these results shows that a strongly adaptive algorithm makes sure that its total loss $\sum_{t=1}^{T} f_t(w_t)$ is at most

$$\min \left\{ \sum_{t=1}^{T} \min_{w\in\Omega} f_t(w) + \widetilde{\mathcal{O}}\left(\sqrt{T} + T^{2/3} V_T^{1/3}\right), \min_S \min_{\substack{u_1,\dots,u_T \\ \text{w. } S-1 \text{ switches}}} \sum_{t=1}^{T} f_t(u_t) + \widetilde{\mathcal{O}}\left(\sqrt{ST}\right) \right\}.$$

Variation of the comparator sequence. Switching regret measures the complexity of the comparator sequence by counting the number of hard switches, so even a tiny change between u_t and u_{t-1} contributes the same amount of complexity as a bigger change, which does not sound very reasonable. Ideally, we want to measure the complexity in a smoother way, such as by $\sum_{t=1}^{T} \|u_t - u_{t-1}\|$, which clearly recovers and generalizes the switching regret. This is indeed possible via a strongly adaptive algorithm again. For simplicity, in what follows we only focus on the expert problem (a special of OCO) and we define $u_0 = \mathbf{0}$ (all-zero vector) for convenience.

Theorem 5. *For the expert problem, if an algorithm ensures $\mathcal{R}_{\mathcal{I}} = \widetilde{\mathcal{O}}(\sqrt{|\mathcal{I}|})$ for all interval $\mathcal{I}$, then it also ensures $\mathcal{R}_T(u_1, \dots, u_T) = \widetilde{\mathcal{O}}\left(\sqrt{TA_T}\right)$ for any $u_1, \dots, u_T \in \Delta(N)$ where $A_T = \sum_{t=1}^T \sum_{i=1}^N [u_t(i) - u_{t-1}(i)]_+ = \frac{1}{2}\sum_{t=1}^T \|u_t - u_{t-1}\|_1$ and $[\cdot]_+$ is a shorthand for $\max\{\cdot, 0\}$.*

Proof. The dynamic regret $\mathcal{R}_T(u_1, \dots, u_T)$ can be written as $\sum_{t=1}^T \sum_{i=1}^N u_t(i) r_t(i)$ where $r_t(i) = \langle p_t - e_i, \ell_t \rangle$ is the instantaneous regret. In the rest of the proof, we will fix an expert i and prove $\sum_{t=1}^T u_t(i) r_t(i) = \widetilde{\mathcal{O}}\left(\sqrt{\sum_{t=1}^T [u_t(i) - u_{t-1}(i)]_+} \sqrt{\sum_{t=1}^T u_t(i)}\right)$, which then completes the proof after summing over i and applying Cauchy-Schwarz inequality.

The idea is to rewrite $\sum_{t=1}^T u_t(i) r_t(i)$ as a a weighted sum of interval regret $\sum_{m=1}^M \alpha_m \mathcal{R}_{\mathcal{I}_m}$, for some intervals $\mathcal{I}_1, \dots, \mathcal{I}_M$ (not necessarily a partition of $[1, T]$) and some corresponding weights $\alpha_1, \dots, \alpha_M > 0$. This rewriting holds as long as $u_t(i) = \sum_{m=1}^M \mathbf{1}\{t \in \mathcal{I}_m\}\alpha_m$ for any t, that is, each $u_t(i)$ can be decomposed as the sum of the weights of the intervals that cover t, because

$$\sum_{t=1}^T u_t(i) r_t(i) = \sum_{t=1}^T \sum_{m=1}^M \mathbf{1}\{t \in \mathcal{I}_m\}\alpha_m r_t(i) = \sum_{m=1}^M \alpha_m \sum_{t=1}^T \mathbf{1}\{t \in \mathcal{I}_m\} r_t(i) \le \sum_{m=1}^M \alpha_m \mathcal{R}_{\mathcal{I}_m}.$$

Applying the interval regret guarantee, we then have

$$\begin{aligned}
\sum_{t=1}^T u_t(i) r_t(i) &= \widetilde{\mathcal{O}}\left(\sum_{m=1}^M \alpha_m \sqrt{|\mathcal{I}_m|}\right) \le \widetilde{\mathcal{O}}\left(\sqrt{\sum_{m=1}^M \alpha_m} \sqrt{\sum_{m=1}^M \alpha_m |\mathcal{I}_m|}\right) \qquad \text{(Cauchy-Schwarz)} \\
&= \widetilde{\mathcal{O}}\left(\sqrt{\sum_{m=1}^M \alpha_m} \sqrt{\sum_{m=1}^M \alpha_m \sum_{t=1}^T \mathbf{1}\{t \in \mathcal{I}_m\}}\right) = \widetilde{\mathcal{O}}\left(\sqrt{\sum_{m=1}^M \alpha_m} \sqrt{\sum_{t=1}^T u_t(i)}\right).
\end{aligned}$$

It thus remains to find α_m and $\mathcal{I}_m$ to minimize $\sum_{m=1}^M \alpha_m$ subject to the constraint $u_t(i) = \sum_{m=1}^M \mathbf{1}\{t \in \mathcal{I}_m\}\alpha_m$. The optimal construction is described below. We will skip its optimality proof here (see [Luo and Schapire, 2015] if interested), but it suffices to prove that this construction indeeds leads to $\sum_{m=1}^M \alpha_m = \sum_{t=1}^T [u_t(i) - u_{t-1}(i)]_+$.

For conciseness, we drop the index i in what follows. The construction is recursive. First, we find $t^\star \in \operatorname{argmin}_t u_t$ and create an interval $\mathcal{I}_1 = [1, T]$ with weight $\alpha_1 = u_{t^\star}$. Then we recursively perform the same construction for the inputs $u_1 - u_{t^\star}, \dots, u_{t^\star - 1} - u_{t^\star}$ and $u_{t^\star + 1} - u_{t^\star}, \dots, u_T - u_{t^\star}$ respectively until there are no non-zero inputs left. Let $h(u_1, \dots, u_T)$ denote the sum of the weights of this construction. We now use an induction (on the length of the input T) to prove $h(u_1, \dots, u_T) = \sum_{t=1}^T [u_t - u_{t-1}]_+ = u_1 + \sum_{t=2}^T [u_t - u_{t-1}]_+$. The base case $T = 1$ holds trivially. Suppose that the statement holds for any input length smaller than T. Then we have

$$\begin{aligned}
h(u_1, \dots, u_T) &= u_{t^\star} + h(u_1 - u_{t^\star}, \dots, u_{t^\star - 1} - u_{t^\star}) + h(u_{t^\star + 1} - u_{t^\star}, \dots, u_T - u_{t^\star}) \\
&= u_{t^\star} + (u_1 - u_{t^\star}) + \sum_{t=2}^{t^\star - 1} [u_t - u_{t-1}]_+ + (u_{t^\star + 1} - u_{t^\star}) + \sum_{t=t^\star + 2}^{T} [u_t - u_{t-1}]_+ \\
&= u_1 + \sum_{t=2}^{t^\star - 1} [u_t - u_{t-1}]_+ + [u_{t^\star + 1} - u_{t^\star}]_+ + \sum_{t=t^\star + 2}^{T} [u_t - u_{t-1}]_+ = \sum_{t=1}^{T} [u_t - u_{t-1}]_+.
\end{aligned}$$

where the last step is by $[u_{t^\star} - u_{t^\star - 1}]_+ = 0$. This finishes the proof. □

Once again, as long as A_T is sublinear, the dynamic regret is sublinear. Note that this bound is also not directly comparable to the one in Theorem 4, and a strongly adaptive algorithm enjoys both of them simultaneously, leading to the following upper bound on the total loss of the algorithm:

$$\min\left\{\sum_{t=1}^T \min_{i \in [N]} \ell_t(i) + \widetilde{\mathcal{O}}\left(\sqrt{T} + T^{2/3} V_T^{1/3}\right), \min_{u_1, \dots, u_T \in \Delta(N)} \sum_{t=1}^T \langle u_t, \ell_t \rangle + \widetilde{\mathcal{O}}\left(\sqrt{TA_T}\right)\right\},$$

where V_T in this case simplifies to $\sum_{t=2}^T \|\ell_t - \ell_{t-1}\|_\infty$.

References

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Non-stationary stochastic optimization. *Operations Research*, 63(5):1227–1244, 2015.

Yoav Freund, Robert E Schapire, Yoram Singer, and Manfred K Warmuth. Using and combining predictors that specialize. In *29th Annual ACM Symposium on the Theory of Computing*, pages 334–343. ACM, 1997.

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

Haipeng Luo and Robert E. Schapire. Achieving All with No Parameters: AdaNormalHedge. In *28th Annual Conference on Learning Theory*, 2015.