Adaptive Regret of Convex and Smooth Functions

Lijun Zhang [1] **Tie-Yan Liu** [2] **Zhi-Hua Zhou** [1]

Abstract

We investigate online convex optimization in changing environments, and choose the adaptive regret as the performance measure. The goal is to achieve a small regret over *every* interval so that the comparator is allowed to change over time. Different from previous works that only utilize the convexity condition, this paper further exploits smoothness to improve the adaptive regret. To this end, we develop novel adaptive algorithms for convex and smooth functions, and establish problem-dependent regret bounds over any interval. Our regret bounds are comparable to existing results in the worst case, and become much tighter when the comparator has a small loss.

1. Introduction

Online convex optimization (OCO) is a powerful learning framework which has both theoretical and practical appeals (Zinkevich, 2003). Given a convex decision set $\mathcal{W}$, the learner is required to select a decision $\mathbf{w}_t \in \mathcal{W}$ in each round t. Then, a convex loss function $f_t : \mathcal{W} \mapsto \mathbb{R}$ is revealed, and the learner suffers loss $f_t(\mathbf{w}_t)$. The goal is to minimize the cumulative loss of the online learner, or equivalently the regret defined as

$$\text{Regret} = \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w} \in \mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w})$$

which is the difference of losses between the learner and the optimal solution in hindsight. In the past decades, various algorithms for minimizing the regret have been developed (Shalev-Shwartz, 2011; Hazan, 2016).

OCO is a natural choice for changing environments in the sense that the loss arrives dynamically. However, in the real-world application, we are also facing another dynamic challenge—the optimal solution may change continuously. For example, in online recommendation, $\mathbf{w}$ models the interest of users, which could evolve over time. In this scenario, regret is no longer a suitable measure of performance, since the online learner is compared against a *fixed* decision. So, the traditional regret is also referred to as *static* regret to emphasize that the comparator is static.

To cope with changing environments, the notion of adaptive regret has been proposed and received considerable interests (Hazan & Seshadhri, 2007; Daniely et al., 2015; Zhang et al., 2018b). The key idea is to minimize the "local" regret

$$\text{Regret}\left([r,s]\right) = \sum_{t=r}^{s} f_t(\mathbf{w}_t) - \min_{\mathbf{w} \in \mathcal{W}} \sum_{t=r}^{s} f_t(\mathbf{w})$$

of *every* interval $[r, s] \subseteq [T]$. Requiring a low regret over any interval essentially means the online learner is evaluated against a changing comparator. For convex functions, the state-of-the-art algorithm achieves an $O(\sqrt{(s-r)\log s})$ regret over any interval $[r, s]$ (Jun et al., 2017), which is close to the minimax regret over a fixed interval (Abernethy et al., 2008). In the studies of static regret, it is well-known that the regret bound can be improved when additional curvatures, such as smoothness, are present (Srebro et al., 2010). Thus, it is natural to ask whether smoothness can also be exploited to enhance the adaptive regret. This paper provides an affirmative answer by developing adaptive algorithms for convex and smooth functions that enjoy tighter bounds.

We remark that directly combining the regret of convex and smooth functions with existing adaptive algorithms does not give a tight adaptive regret, because of the following technical challenges.

- The regret bound for convex and smooth functions requires to know the loss of the optimal decision (Srebro et al., 2010), which is generally unavailable.
- Existing adaptive algorithms have some components, including a meta-algorithm and a set of intervals, that cannot utilize smoothness.

To address the above challenges, we first introduce the scale-free online gradient descent (SOGD), a special case of the scale-free mirror descent (Orabona & Pál, 2018), and demonstrate that SOGD is able to exploit smoothness automatically and does not need any prior knowledge. Then,

[1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China [2]Microsoft Research Asia, Beijing, China. Correspondence to: Lijun Zhang <zhanglj@lamda.nju.edu.cn>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

we develop a Strongly Adaptive algorithm for Convex and Smooth functions (SACS), which runs multiple instances of SOGD over a set of carefully designed intervals, and combines them with an expert-tracking algorithm that can benefit from small losses. Let $L_r^s = \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=r}^{s} f_t(\mathbf{w})$ be the minimal loss over an interval $[r, s]$. Our theoretical analysis demonstrates that the regret of SACS over any interval $[r, s]$ is $O(\sqrt{L_r^s \log s \cdot \log(s-r)})$, which could be much smaller than the existing $O(\sqrt{(s-r)\log s})$ bound when L_r^s is small. Finally, to further improve the performance, we propose a novel way to construct problem-dependent intervals, and attain an $O(\sqrt{L_r^s \log L_1^s \cdot \log L_r^s})$ bound.

2. Related Work

Adaptive regret has been studied in the settings of prediction with expert advice (PEA) and online convex optimization (OCO). Existing algorithms are closely related in the sense that adaptive algorithms designed for OCO are usually built upon those designed for PEA.

In an early study of PEA, Littlestone & Warmuth (1994) develop one variant of weighted majority algorithm for tracking the best expert. One intermediate result, i.e., Lemma 3.1 of Littlestone & Warmuth (1994) provides a mistake bound for any interval, which is analogous to the adaptive regret. The concept of adaptive regret is formally introduced by Hazan & Seshadhri (2007) in the context of OCO. Specifically, Hazan & Seshadhri (2007) introduce the adaptive regret

$$\text{A-Regret}(T) = \max_{[r,s]\subseteq[T]} \text{Regret}\left([r, s]\right) \tag{1}$$

which is the maximum regret over any contiguous interval, and propose a new algorithm named follow the leading history (FLH), which contains 3 parts:

- An expert-algorithm, which is able to minimize the static regret of a given interval;
- A set of intervals, each of which is associated with an expert-algorithm that minimizes the regret of that interval;
- A meta-algorithm, which combines the predictions of active experts in each round.

For exponentially concave (abbr. exp-concave) functions, Hazan & Seshadhri (2007) use online Newton step (Hazan et al., 2007) as the expert-algorithm. For the construction of intervals, they consider two different approaches. In the first approach, the set of intervals is $\{[t, \infty], t \in \mathbb{N}\}$ which means an expert will be initialized at each round t and live forever. In the second approach, the set of intervals is $\{[t, e_t], t \in \mathbb{N}\}$, meaning the expert that becomes active in round t will be removed after e_t. Here, e_t denotes the ending time of the interval started from t, and its value is set according to a data streaming algorithm. Hazan et al. (2007) develop a meta-algorithm based on Fixed-Share (Herbster & Warmuth, 1998), and allow the set of experts to change dynamically.

FLH with the first set of intervals attains an $O(d \log T)$ adaptive regret, where d is the dimensionality, but is inefficient since it maintains t experts in round t. In contrast, FLH with the second set of intervals achieves a higher $O(d \log^2 T)$ bound, but is efficient because it only keeps $O(\log t)$ experts in the t-th round. Thus, we observe that the intervals control the tradeoff between the adaptive regret and the computational cost. György et al. (2012) and Zhang et al. (2018b) have developed new ways to construct intervals which can trade effectiveness for efficiency explicitly. Furthermore, when the function is strongly convex, the dependence on d in the upper bound disappears (Zhang et al., 2018b).

For convex functions, Hazan et al. (2007) modify the FLH algorithm by replacing the expert-algorithm with any low-regret method for convex functions, and introducing a parameter of step size in the meta-algorithm. In this case, the efficient and inefficient versions of FLH achieve $O(\sqrt{T \log^3 T})$ and $O(\sqrt{T \log T})$ adaptive regret bounds, respectively.[1] One limitation of this result is that it does not guarantee to perform well on small intervals, because the upper bounds are meaningless for intervals of size $O(\sqrt{T})$.

The adaptive regret of PEA setting is studied by Adamskiy et al. (2012). Let $L_{t,i}$ be the loss of the i-th expert in round t, and L_t be the loss of the learner, which is generally a convex combination of $L_{t,i}$'s. In this case, the regret over interval $[r, s]$ in (1) becomes

$$\text{Regret}\left([r, s]\right) = \sum_{t=r}^{s} L_t - \min_i \sum_{t=r}^{s} L_{t,i}.$$

They pointed out that the meta-algorithm of Hazan et al. (2007) can be reduced to the Fixed-Share algorithm with a special configuration of parameters. Although Fixed-Share is designed to minimize the tracking regret, Adamskiy et al. (2012) show that it can also minimize the adaptive regret. Combining Hoeffding bound (Cesa-Bianchi & Lugosi, 2006, Lemma 2.2) and (1a) of Adamskiy et al. (2012), it is easy to prove that the adaptive regret of Fixed-Share is $O(\sqrt{T \log NT})$, where N is the number of experts.[2] Unfortunately, it also does not respect short intervals well.

To ensure a good performance on every interval, Daniely et al. (2015) propose the notion of strongly adaptive regret

$$\text{SA-Regret}(T, \tau) = \max_{[s,s+\tau-1]\subseteq[T]} \text{Regret}\left([s, s+\tau-1]\right)$$

[1] As pointed out by Hazan & Seshadhri (2009), online gradient descent with constant step size (Zinkevich, 2003) can also be used to minimize the adaptive regret of convex functions, and the bound is $O(\sqrt{T})$.

[2] We need to use Hoeffding bound to convert the mix loss defined by Adamskiy et al. (2012) to the traditional weighted loss.

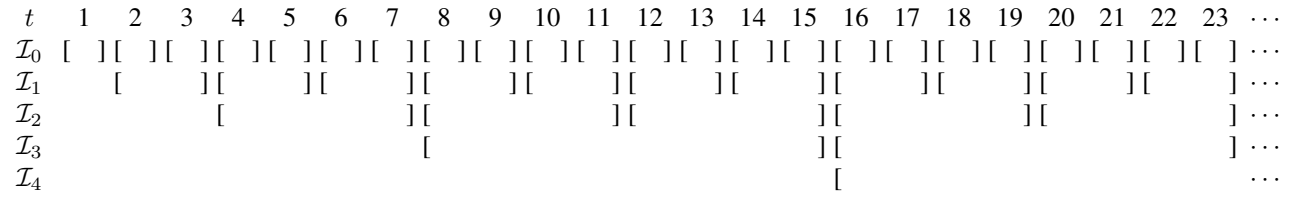

Figure 1. Geometric covering (GC) intervals of Daniely et al. (2015). In the figure, each interval is denoted by [ ].

which emphasizes the dependency on the interval length τ, and investigate both the PEA and OCO settings. The main contribution of that paper is a new meta-algorithm for combining experts, namely strongly adaptive online learner (SAOL), which is similar to the multiplicative weights method (Arora et al., 2012). Furthermore, they also propose a different way to construct the set of intervals as

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

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

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

Following Jun et al. (2017), we refer to $\mathcal{I}$ as geometric covering (GC) intervals and present a graphical illustration in Fig. 1. It is obvious to see that each $\mathcal{I}_k$ is a partition of $\mathbb{N} \setminus \{1, \cdots, 2^k - 1\}$ to consecutive intervals of length 2^k.

In the PEA setting, by using multiplicative weights as the expert-algorithm, Daniely et al. (2015) establish a strongly adaptive regret of $O(\sqrt{\tau \log N} + \log T \sqrt{\tau})$. In the OCO setting, by using online gradient descent as the expert-algorithm, Daniely et al. (2015) establish a strongly adaptive regret of $O(\log T \sqrt{\tau})$. Those rates are further improved by Jun et al. (2017), who develop a new meta-algorithm named as sleeping coin betting (CB). The strongly adaptive regrets of PEA and OCO are improved to $O(\sqrt{\tau \log NT})$ and $O(\sqrt{\tau \log T})$, respectively. Recently, Wang et al. (2018) demonstrate that for minimizing the adaptive regret of convex functions, we can use surrogate loss to reduce the number of gradient evaluations per round from $O(\log T)$ to 1.

Finally, we note that adaptive regret is closely related to the tracking regret in PEA (Herbster & Warmuth, 1998; György et al., 2012; Cesa-bianchi et al., 2012) and dynamic regret in OCO (Hall & Willett, 2013; Jadbabaie et al., 2015; Mokhtari et al., 2016; Yang et al., 2016; Zhang et al., 2017; 2018a). Specifically, from adaptive regret, we can derive a tight bound for the tracking regret (Jun et al., 2017) and a special form of dynamic regret (Zhang et al., 2018b).

3. Main Results

We first investigate how to utilize smoothness to improve the static regret, then develop a strongly adaptive algorithm for convex and smooth functions, and finally propose data-dependent intervals to further strengthen the performances. All the proofs can be found in the full paper (Zhang et al., 2019).

3.1. Scale-free Online Gradient Descent (SOGD)

We introduce common assumptions used in our paper.

Assumption 1 *The domain $\mathcal{W}$ is convex, and its diameter is bounded by D, i.e.,*

$$\max_{\mathbf{w}, \mathbf{w}' \in \mathcal{W}} \|\mathbf{w} - \mathbf{w}'\|_2 \leq D. \tag{2}$$

Assumption 2 *All the online functions are convex and non-negative.*

Assumption 3 *All the online functions are H-smooth over $\mathcal{W}$, that is,*

$$\|\nabla f_t(\mathbf{w}) - \nabla f_t(\mathbf{w}')\| \leq H \|\mathbf{w} - \mathbf{w}'\| \tag{3}$$

for all $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$, $t \in [T]$.

Note that in Assumption 2, we require the online function to be nonnegative outside the domain $\mathcal{W}$. This is a precondition for establishing the self-bounding property of smooth functions, which can be exploited to deliver a tight regret bound. Specifically, Srebro et al. (2010) consider online gradient descent with constant step size:

$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}} \left[\mathbf{w}_t - \eta \nabla f_t(\mathbf{w}_t) \right], \ \forall t \geq 1$$

where $\mathbf{w}_1 \in \mathcal{W}$ and $\Pi_{\mathcal{W}}[\cdot]$ denotes the projection onto the nearest point in $\mathcal{W}$, and prove the following regret bound (Srebro et al., 2010, Theorem 2).

Theorem 1 *Let $B \geq 0$ and $L \geq 0$ be two constants, set the step size in OGD as*

$$\eta = \frac{1}{HB^2 + \sqrt{H^2 B^4 + HB^2 L}},$$

and $\mathbf{w}_1 = 0$. Under Assumptions 2 and 3, we have

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \leq 4HB^2 + 2\sqrt{HB^2 L}$$

for any $\mathbf{w} \in \mathcal{W}$ such that $\frac{\|\mathbf{w}\|^2}{2} \leq B^2$, and $\sum_{t=1}^{T} f_t(\mathbf{w}) \leq L$.

The above theorem indicates that under the smoothness condition, the regret bound could be tighter if the cumulative loss of the comparator $\mathbf{w}$ is small, Specifically, when $L = o(T)$, the regret bound becomes $o(\sqrt{T})$, thus improves the minimax rate of online convex optimization (Abernethy et al., 2008). However, one limitation of Theorem 1 is that the step size depends on the bound L on the loss in hindsight.

The standard way to address the above problem is the "doubling trick" (Cesa-Bianchi & Lugosi, 2006), but it requires the online learner to evaluate the minimal cumulative loss on the fly, which is computationally expensive. Instead, we make use of the scale-free mirror descent algorithm of Orabona & Pál (2018) and set the step size of the t-th iteration as

$$\eta_t = \frac{\alpha}{\sqrt{\delta + \sum_{i=1}^{t} \|\nabla f_i(\mathbf{w}_i)\|^2}} \tag{4}$$

where the parameter $\delta > 0$ is introduced to avoid being divided by 0, and $\alpha > 0$ is used to fine-tune the upper bound. We note that the step size in (4) is similar to the self-confident tuning originally proposed for online linear regression (Auer et al., 2002), and later extended to self-bounded functions (Shalev-Shwartz, 2007, Theorem 2). The new algorithm is named as scale-free online gradient descent (SOGD), and summarized in Algorithm 1.

Next, we prove the regret bound of SOGD in the following theorem, which demonstrates that SOGD can make use of smoothness automatically.

Theorem 2 *Set $\delta > 0$ and $\alpha = D/\sqrt{2}$ in Algorithm 1. Under Assumptions 1, 2 and 3, SOGD satisfies*

$$\begin{aligned}&\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \\ \leq & 8HD^2 + D\sqrt{2\delta + 8H\sum_{t=1}^{T} f_t(\mathbf{w})}\end{aligned}$$

for any $\mathbf{w} \in \mathcal{W}$.

Remark: First, comparing Theorem 2 with Theorem 1, we observe that the regret bound of SOGD is of the same order as that of SGD with optimal parameters. Second, because the step size of SOGD is automatically tuned during the learning process, it is equipped with an anytime regret bound, i.e., its regret bound holds for any T. This nice property of SOGD will be utilized to simplify the design of adaptive algorithms.

3.2. A Strongly Adaptive Algorithm

Similar to previous studies (Hazan & Seshadhri, 2007; Daniely et al., 2015; Jun et al., 2017), our strongly adaptive algorithm contains 3 components: an expert-algorithm, a set of intervals, and a meta-algorithm.

Algorithm 1 Scale-free online gradient descent (SOGD)

Input: parameters δ and α
Initialize $\mathbf{w}_1 \in \mathcal{W}$ arbitrarily
for $t = 1$ **to** T **do**
  Submit $\mathbf{w}_t$ and then receive function $f_t(\cdot)$
  Suffer loss $f_t(\mathbf{w}_t)$ and set η_t as (4)
  Update the decision according to

$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}\left[\mathbf{w}_t - \eta_t \nabla f_t(\mathbf{w}_t)\right]$$

end for

3.2.1. The Procedure

For the expert-algorithm, we choose the scale-free online gradient descent (SOGD) in Algorithm 1, since it can utilize smoothness to improve the regret bound. For the set of intervals, we can directly re-use the GC intervals of Daniely et al. (2015). However, because an instance of SOGD will be created for each interval and SOGD has an anytime regret bound, we can further simplify GC intervals based on the following observation: For intervals with the same starting point, we only need to keep the longest one, since the expert associated with this interval can replace others.

Take the set of intervals $\{[4,4],[4,5],[4,7]\}$ in Fig. 1 as an example, and denote the expert associated with interval I as E_I. The expert $E_{[4,7]}$ performs exactly the same as the expert $E_{[4,4]}$ in round 4, and exactly the same as the expert $E_{[4,5]}$ in rounds 4 and 5. Thus, we can use $E_{[4,7]}$ to replace $E_{[4,4]}$ and $E_{[4,5]}$ in any place (algorithm or analysis) they appear. Mathematically, our compact geometric covering (CGC) intervals are defined as

$$\mathcal{C} = \bigcup_{k \in \mathbb{N} \cup \{0\}} \mathcal{C}_k \tag{5}$$

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

$$\mathcal{C}_k = \left\{[i \cdot 2^k, (i+1) \cdot 2^k - 1] : i \text{ is odd}\right\}.$$

A graphical illustration of CGC intervals is given in Fig. 2. Comparing Fig. 1 with Fig. 2, the main difference is that CGC only adds 1 interval in each round, while CG may add multiple intervals in each round.

Finally, we need to specify the meta-algorithm. One may attempt to use the SAOL of Daniely et al. (2015) or the sleeping CB of Jun et al. (2017). However, neither of them meets our requirements, because their meta-regret depends on the length of the interval and cannot benefit from small losses of experts. Instead, we choose a recently developed expert-tracking procedure—AdaNormalHedge (Luo & Schapire, 2015) as the meta-algorithm because

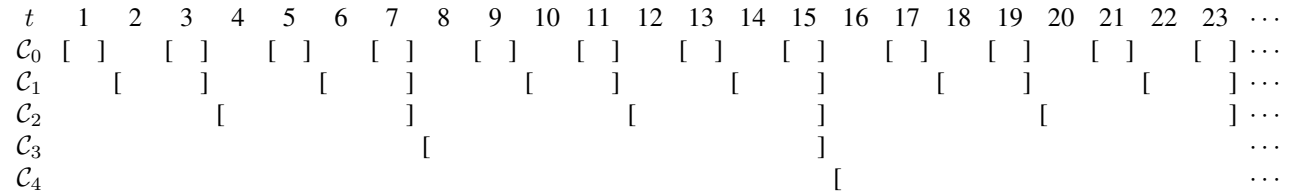

Figure 2. Compact geometric covering (CGC) intervals. In the figure, each interval is denoted by [ ].

(i) it achieves a small regret when the comparator has a small loss, thus can be naturally combined with SOGD which enjoys a similar property;
(ii) it supports the sleeping expert problem, and thus the number of experts can vary over time.

The key ingredients of AdaNormalHedge are a potential function:

$$\Phi(R, C) = \exp\left(\frac{[R]_+^2}{3C}\right)$$

where $[x]_+ = \max(0, x)$ and $\Phi(0, 0)$ is defined to be 1, and a weight function with respect to this potential:

$$w(R, C) = \frac{1}{2}\big(\Phi(R + 1, C + 1) - \Phi(R - 1, C + 1)\big).$$

In the t-th round, AdaNormalHedge assigns a weight $p_{t,i}$ to an expert E_i according to

$$p_{t,i} \propto w(R_{t-1,i}, C_{t-1,i})$$

where $R_{t-1,i}$ is the regret with respect to E_i over the first $t-1$ iterations, and $C_{t-1,i}$ is the sum of the absolute value of the instantaneous regret over the first $t-1$ iterations.

Putting everything together, we present our Strongly Adaptive algorithm for Convex and Smooth functions (SACS) in Algorithm 2. For each interval $[i, j] \in \mathcal{C}$, we will create an expert $E_{[i,j]}$ which is active during the interval $[i, j]$. Note that in our CGC intervals, the starting point of each interval is *unique*. So, to simplify notations, we use E_i as a shorthand of $E_{[i,j]}$.

On the t-round, we first create an expert E_t by running an instance of SOGD (Step 2) and add it to the set of active experts, denoted by $\mathcal{S}_t$ (Step 3). In Step 4, we receive the prediction $\mathbf{w}_{t,i}$ of each $E_i \in \mathcal{S}_t$, and assign the following weight to E_i

$$p_{t,i} = \frac{w(R_{t-1,i}, C_{t-1,i})}{\sum_{E_i \in \mathcal{S}_t} w(R_{t-1,i}, C_{t-1,i})} \tag{6}$$

where

$$R_{t-1,i} = \sum_{u=i}^{t-1} f_u(\mathbf{w}_u) - f_u(\mathbf{w}_{u,i}),$$
$$C_{t-1,i} = \sum_{u=i}^{t-1} \left|f_u(\mathbf{w}_u) - f_u(\mathbf{w}_{u,i})\right|.$$

Algorithm 2 Strongly Adaptive algorithm for Convex and Smooth functions (SACS)

1: **for** $t = 1$ **to** T **do**
2: Initialize an expert E_t by invoking SOGD in Algorithm 1 and set $R_{t-1,t} = C_{t-1,t} = 0$
3: Add E_t to the set of active experts

$$\mathcal{S}_t = \mathcal{S}_{t-1} \cup \{E_t\}$$

4: Receive the prediction $\mathbf{w}_{t,i}$ of each expert $E_i \in \mathcal{S}_t$, and calculate its weight $p_{t,i}$ according to (6)
5: Submit $\mathbf{w}_t$ defined in (7) and then receive $f_t(\cdot)$
6: Remove experts whose ending times are t

$$\mathcal{S}_t = \mathcal{S}_t \setminus \{E_i | [i, t] \in \mathcal{C}\}$$

7: For each $E_i \in \mathcal{S}_t$, update

$$R_{t,i} = R_{t-1,i} + f_t(\mathbf{w}_t) - f_t(\mathbf{w}_{t,i}),$$
$$C_{t,i} = C_{t-1,i} + |f_t(\mathbf{w}_t) - f_t(\mathbf{w}_{t,i})|$$

8: Pass $f_t(\cdot)$ to each expert $E_i \in \mathcal{S}_t$
9: **end for**

In Step 5, SACS submits the weighted average of $\mathbf{w}_{t,i}$

$$\mathbf{w}_t = \sum_{E_i \in \mathcal{S}_t} p_{t,i} \mathbf{w}_{t,i} \tag{7}$$

as the output, and suffers loss $f_t(\mathbf{w}_t)$. In Step 6, we remove all the experts whose ending times are t, and in Step 7, we update the parameters of each remaining expert. Finally, we pass the loss function $f_t(\cdot)$ to all experts in $\mathcal{S}_t$ so that they can update their predictions for the $(t+1)$-th round (Step 8).

3.2.2. Theoretical Guarantees

In the following, we present theoretical guarantees of SACS. To simplify our presentations, we assume all the convex functions are bounded by 1.

Assumption 4 *The value of each function belongs to* $[0, 1]$*, i.e.,*

$$0 \leq f_t(\mathbf{w}) \leq 1, \ \forall \mathbf{w} \in \mathcal{W}, t \in [T].$$

As long as the loss functions are bounded, they can always

be scaled and restricted to $[0, 1]$.

We start with the meta-regret of SACS with respect to an expert E_i.

Lemma 1 *Under Assumptions 2 and 4, for any interval $[i, j] \in \mathcal{C}$, and any $t \in [i, j]$, SACS satisfies*

$$\sum_{u=i}^{t} [f_u(\mathbf{w}_u) - f_u(\mathbf{w}_{u,i})] \leq c(t) + \sqrt{2c(t) \sum_{u=i}^{t} f_u(\mathbf{w}_{u,i})}$$

where $c(t) = 3 \ln(4t^2)$.

Remark: First, compared with the meta-regret of SAOL (Daniely et al., 2015) and sleeping CB (Jun et al., 2017), the main advantage of SACS is that its upper bound depends on the cumulative loss of the expert, which could be much tighter when the problem is easy. Second, the theoretical guarantee of SACS is an anytime regret bound, since the upper bound holds for any $t \in [i, j]$.

Combining Lemma 1 with the regret bound of SOGD in Theorem 2, we immediately obtain the following regret bound of SACS over any interval $[i, j] \in \mathcal{C}$.

Lemma 2 *Under Assumptions 1, 2, 3 and 4, for any interval $[i, j] \in \mathcal{C}$, any $t \in [i, j]$, and any $\mathbf{w} \in \mathcal{W}$, SACS satisfies*

$$\sum_{u=i}^{t} [f_u(\mathbf{w}_u) - f_u(\mathbf{w})] \leq a(t) + \sqrt{b(t) \sum_{u=i}^{t} f_u(\mathbf{w})}$$

where

$$a(t) = \frac{9}{2} \ln(4t^2) + 18HD^2 + 2D\sqrt{2\delta} \tag{8}$$

and

$$b(t) = 24 \ln(4t^2) + 16HD^2. \tag{9}$$

By utilizing the special structure of the interval set $\mathcal{C}$, we extend Lemma 2 to any interval $[r, s] \subseteq [T]$.

Theorem 3 *Under Assumptions 1, 2, 3 and 4, for any interval $[r, s] \subseteq [T]$ and any $\mathbf{w} \in \mathcal{W}$, SACS satisfies*

$$\begin{aligned}
&\sum_{t=r}^{s} [f_t(\mathbf{w}_t) - f_t(\mathbf{w})] \\
\leq & va(s) + \sqrt{vb(s) \sum_{t=r}^{s} f_t(\mathbf{w})} \\
= & O\left(\sqrt{\left(\sum_{t=r}^{s} f_t(\mathbf{w}) \right) \log s \cdot \log(s - r)} \right)
\end{aligned}$$

where $v \leq \lceil \log_2(s - r + 2) \rceil$, $a(\cdot)$ and $b(\cdot)$ are respectively defined in (8) and (9).

Remark: In the literature, the best adaptive regret for convex functions is $O(\sqrt{(s - r) \log s})$ of Jun et al. (2017). Although our upper bound in Theorem 3 has an additional dependence on $\sqrt{\log(s - r)}$, it replaces the interval length $s - r$ with the cumulative loss over that interval, i.e., $\sum_{t=r}^{s} f_t(\mathbf{w})$. As a result, our bound could be much tighter when the comparator has a small loss. Whether the additional $\sqrt{\log(s - r)}$ factor can be removed remains an open problem to us, and we leave it as a future work.

3.3. Problem-dependent Intervals

We can refer to our result in Theorem 3 as a *problem-dependent* bound, since the dominant factor $\sqrt{\sum_{t=r}^{s} f_t(\mathbf{w})}$ depends on the problem, which has a similar spirit with the *data-dependent* bound of Adagrad (Duchi et al., 2011). One unsatisfactory point of Theorem 3 is that the logarithmic factor $\log s \cdot \log(s - r)$, although non-dominant, is problem-independent. In this section, we discuss how to make SACS fully problem-dependent.

The problem-independent factor appears because CGC intervals, as well as CG intervals, are problem-independent. To address this limitation, we propose a problem-dependent way to generate intervals dynamically. The basic idea is to run an instance of SOGD, and restart the algorithm when the cumulative loss is larger than some threshold. The time points when SOGD restarts will be used as the starting rounds of intervals.

Specifically, we set $s_1 = 1$ and run an instance of SOGD. Let $s_1 + \alpha$ be the round such that the cumulative loss becomes larger than a threshold C. Then, we set $s_2 = s_1 + \alpha + 1$ and restart SOGD in round s_2. Repeating this process, we can generate a sequence of points $s_1, s_2, s_3, \ldots$ which is referred to as *markers*. Our problem-dependent geometric covering (PGC) intervals are constructed based on markers:

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

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

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

Similarly, we can also compact PGC intervals by removing overlapping intervals with the same starting point. The compact problem-dependent geometric covering (CPGC) intervals are given by

$$\widetilde{\mathcal{C}} = \bigcup_{k \in \mathbb{N} \cup \{0\}} \widetilde{\mathcal{C}}_k \tag{10}$$

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

$$\widetilde{\mathcal{C}}_k = \left\{ [s_{i \cdot 2^k}, s_{(i+1) \cdot 2^k} - 1] : i \text{ is odd} \right\}.$$

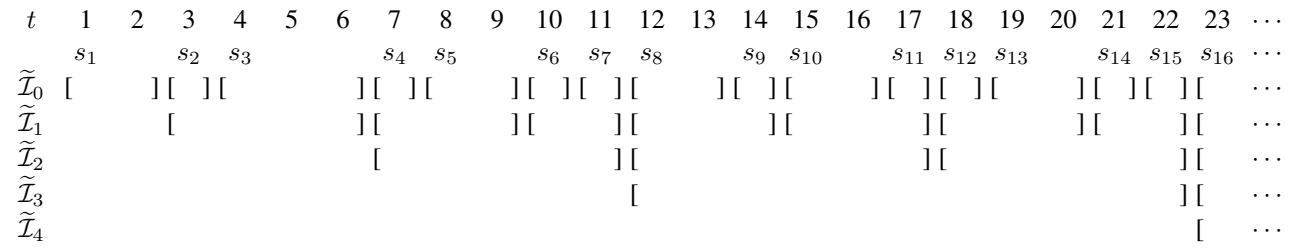

Figure 3. Problem-dependent geometric covering (PGC) intervals. In the figure, each interval is denoted by [ ].

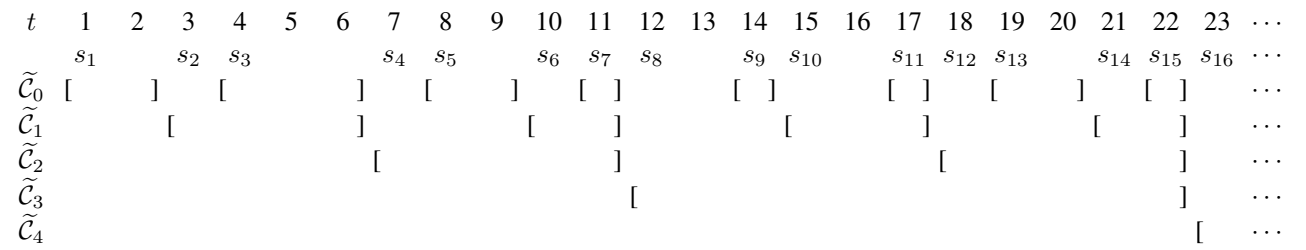

Figure 4. Compact problem-dependent geometric covering (CPGC) intervals. In the figure, each interval is denoted by [ ].

We provide graphical illustrations of PGC intervals and CPGC intervals in Fig. 3 and Fig. 4, respectively.

To see the difference between problem-independent and problem-dependent intervals, let's compare Fig. 1 of GC intervals and Fig. 3 of PGC intervals. We have the following observations.

- In the former one, intervals belong to the same level, i.e., $\mathcal{I}_k$, are of the same length, while in the latter one, intervals belong to the same level, i.e., $\widetilde{\mathcal{I}}_k$, are of different lengths.
- In the former one, an interval is created for each round. By contrast, in the latter one, an interval is created only at markers. Thus, the number of problem-dependent intervals is smaller than that of problem-independent intervals.

We then incorporate CPGC intervals into our SACS algorithm, and summarize the procedure in Algorithm 3. The new algorithm is a bit more complex than the original one in Algorithm 2 because we need to construct CPGC intervals on the fly.

Next, we explain the main differences. To generate CPGC intervals dynamically, we introduce a Boolean variable $NewInterval$ to indicate whether a new interval should be created, m to denote the total number of intervals created so far, and n to denote the index of the latest interval. In each round t, if $NewInterval$ is true, we will create a new expert E_t, add it to the active set, and then reset the indicator (Steps 5 to 7). We also increase the total number of intervals by 1 in Step 8, and note that the m-th marker $s_m = t$. Let $m = i \cdot 2^k$, where i is odd and $k \in \mathbb{N}$. According to the definition of CPGC intervals, $E_t = E_{s_m}$ is active during the interval $[s_{i \cdot 2^k}, s_{(i+1) \cdot 2^k} - 1]$. So, it should be removed before the $s_{(i+1) \cdot 2^k}$-th round. However, the value of $s_{(i+1) \cdot 2^k}$ is *unknown* in the t-th round, so we cannot tag the ending time to E_t. As an alternative, we record the value of $(i+1) \cdot 2^k$, denoted by g_t (Step 9), and remove E_t when m is going to reach g_t (Step 18).

To generate the next marker s_{m+1}, we keep track of the index of the latest expert (Step 10), and record its cumulative loss (Steps 11 and 15). When the cumulative loss is larger than the threshold C (Step 16), we set the indicator $NewInterval$ to be true (Step 17) and remove all the experts whose ending times are $s_{m+1} - 1$ (Step 18). All the other steps are identical to those in Algorithm 2.

We present theoretical guarantees of Algorithm 3. As before, we first prove the meta-regret.

Lemma 3 *Suppose*

$$C \geq 20HD^2 + 2D\sqrt{2\delta}. \tag{11}$$

Under Assumptions 2 and 4, for any interval $[i, j] \in \widetilde{\mathcal{C}}$, and any $t \in [i, j]$, SACS with CPGC intervals satisfies

$$\sum_{u=i}^{t} [f_u(\mathbf{w}_u) - f_u(\mathbf{w}_{u,i})] \leq \tilde{c}(t) + \sqrt{2\tilde{c}(t) \sum_{u=i}^{t} f_u(\mathbf{w}_{u,i})}$$

where

$$\tilde{c}(t) \leq 3 \ln \left(1 + \frac{4}{C} \sum_{u=1}^{t} f_u(\mathbf{w}) \right) + 3 \ln \frac{5 + 3\ln(1+t)}{2}. \tag{12}$$

Remark: Following previous studies (Chernov & Vovk, 2010; Luo & Schapire, 2015), we treat the double logarithmic factor in $\tilde{c}(t)$ as a constant. Compared with

Algorithm 3 SACS with CPGC intervals

Input: Parameter C
Initialize indicator $NewInterval = true$, the total number of intervals $m = 0$, the index of the latest interval $n = 0$
for $t = 1$ **to** T **do**
  if $NewInterval$ is $true$ **then**
    Initialize an expert E_t by invoking SOGD in Algorithm 1 and set $R_{t-1,t} = C_{t-1,t} = 0$
    Add E_t to the set of active experts

$$\mathcal{S}_t = \mathcal{S}_{t-1} \cup \{E_t\}$$

    Reset the indicator $NewInterval = false$
    Update the total number of intervals $m = m + 1$
    Set $g_t = j$ such that $[m, j-1] \in \mathcal{C}$
    Record the index of the latest expert $n = t$
    Initialize the cumulative loss $L_{t-1} = 0$
  end if
  Receive the prediction $\mathbf{w}_{t,i}$ of each expert $E_i \in \mathcal{S}_t$, and calculate its weight $p_{t,i}$ according to (6)
  Submit $\mathbf{w}_t$ defined in (7) and then receive $f_t(\cdot)$
  Update the cumulative loss of the latest expert E_n

$$L_t = L_{t-1} + f_t(\mathbf{w}_{t,n})$$

  if $L_t > C$ **then**
    Set the indicator $NewInterval = true$
    Remove experts whose ending times are $t + 1$

$$\mathcal{S}_t = \mathcal{S}_t \setminus \{E_i | g_i = m + 1\}$$

  end if
  For each $E_i \in \mathcal{S}_t$, update

$$\begin{aligned} R_{t,i} =& R_{t-1,i} + f_t(\mathbf{w}_t) - f_t(\mathbf{w}_{t,i}), \\ C_{t,i} =& C_{t-1,i} + |f_t(\mathbf{w}_t) - f_t(\mathbf{w}_{t,i})| \end{aligned}$$

  Pass $f_t(\cdot)$ to each expert $E_i \in \mathcal{S}_t$
end for

Lemma 1, the main advantage is that $c(t)$ is replaced with a problem-dependent term $\tilde{c}(t)$.

Based on Lemma 3 and Theorem 2, we prove a counterpart of Lemma 2, which bounds the regret over any interval in $\widetilde{\mathcal{C}}$.

Lemma 4 *Under condition (11) and Assumptions 1, 2, 3 and 4, for any interval $[i, j] \in \widetilde{\mathcal{C}}$, any $t \in [i, j]$, and any $\mathbf{w} \in \mathcal{W}$, SACS with CPGC intervals satisfies*

$$\sum_{u=i}^{t} [f_u(\mathbf{w}_u) - f_u(\mathbf{w})] \leq \tilde{a}(t) + \sqrt{\tilde{b}(t) \sum_{u=i}^{t} f_u(\mathbf{w})}$$

where

$$\tilde{a}(t) = \frac{3}{2}\tilde{c}(t) + 18HD^2 + 2D\sqrt{2\delta}, \tag{13}$$

$$\tilde{b}(t) = 8\tilde{c}(t) + 16HD^2, \tag{14}$$

and $\tilde{c}(t)$ conforms to (12).

Finally, we extend Lemma 4 to any interval $[r, s] \subseteq [T]$.

Theorem 4 *Under condition (11) and Assumptions 1, 2, 3 and 4, for any interval $[r, s] \subseteq [T]$ and any $\mathbf{w} \in \mathcal{W}$, SACS with CPGC intervals satisfies*

$$\begin{aligned} &\sum_{t=r}^{s} [f_t(\mathbf{w}_t) - f_t(\mathbf{w})] \\ \leq& 2(C+1) + \frac{3}{2}\tilde{c}(s) + v\tilde{a}(s) + \sqrt{v\tilde{b}(s) \sum_{t=r}^{s} f_t(\mathbf{w})} \\ =& O\left(\sqrt{\left(\sum_{t=r}^{s} f_t(\mathbf{w})\right) \log \sum_{t=1}^{s} f_t(\mathbf{w}) \cdot \log \sum_{t=r}^{s} f_t(\mathbf{w})}\right) \end{aligned}$$

where

$$v \leq \left\lceil \log_2 \left(2 + \frac{4}{C} \sum_{t=r}^{s} f_t(\mathbf{w})\right) \right\rceil$$

$\tilde{a}(\cdot)$, $\tilde{b}(\cdot)$ and $\tilde{c}(\cdot)$ are respectively defined in (13), (14), and (12).

Remark: Compared with the upper bound in Theorem 3, we observe that the problem-independent term $\log s \cdot \log(s - r)$ is improved to $\log \sum_{t=1}^{s} f_t(\mathbf{w}) \cdot \log \sum_{t=r}^{s} f_t(\mathbf{w})$. As a result, our SACS with CPGC intervals becomes fully problem-dependent.

4. Conclusion and Future Work

In this paper, we propose a Strongly Adaptive algorithm for Convex and Smooth functions (SACS), which combines the strength of online gradient descent (OGD), geometric covering (GC) intervals, and AdaNormalHedge. Let $L_r^s(\mathbf{w})$ be the cumulative loss of a comparator $\mathbf{w}$ over an interval $[r, s]$. Theoretical analysis shows that the regret of SACS over any $[r, s]$ with respect to any $\mathbf{w}$ is $O(\sqrt{L_r^s(\mathbf{w}) \log s \cdot \log(s - r)})$, which could be much smaller than the state-of-the-art result (Jun et al., 2017) when $L_r^s(\mathbf{w})$ is small. Furthermore, we propose to construct problem-dependent intervals, and improve the regret bound to $O(\sqrt{L_r^s(\mathbf{w}) \log L_1^s(\mathbf{w}) \cdot \log L_r^s(\mathbf{w})})$.

One future work is to extend our results to exp-concave functions. Note that the static regret of exp-concave functions can be improved by smoothness (Orabona et al., 2012). Thus, it is possible to improve the adaptive regret of exp-concave functions by combining the regret bound of Orabona et al. (2012) and our problem-dependent intervals.

Acknowledgements

This work was partially supported by the National Key R&D Program of China (2018YFB1004300), NSFC (61751306), JiangsuSF (BK20160658), YESS (2017QNRC001), Microsoft Research Asia, and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

Abernethy, J., Bartlett, P. L., Rakhlin, A., and Tewari, A. Optimal stragies and minimax lower bounds for online convex games. In *Proceedings of the 21st Annual Conference on Learning Theory*, pp. 415–423, 2008.

Adamskiy, D., Koolen, W. M., Chernov, A., and Vovk, V. A closer look at adaptive regret. In *Proceedings of the 23rd International Conference on Algorithmic Learning Theory*, pp. 290–304, 2012.

Arora, S., Hazan, E., and Kale, S. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.

Auer, P., Cesa-Bianchi, N., and Gentile, C. Adaptive and self-confident on-line learning algorithms. *Journal of Computer and System Sciences*, 64(1):48–75, 2002.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Cesa-bianchi, N., Gaillard, P., Lugosi, G., and Stoltz, G. Mirror descent meets fixed share (and feels no regret). In *Advances in Neural Information Processing Systems 25*, pp. 980–988, 2012.

Chernov, A. and Vovk, V. Prediction with advice of unknown number of experts. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, pp. 117–125, 2010.

Daniely, A., Gonen, A., and Shalev-Shwartz, S. Strongly adaptive online learning. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1405–1411, 2015.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

György, A., Linder, T., and Lugosi, G. Efficient tracking of large classes of experts. *IEEE Transactions on Information Theory*, 58(11):6709–6725, 2012.

Hall, E. C. and Willett, R. M. Dynamical models and tracking regret in online convex programming. In *Proceedings of the 30th International Conference on Machine Learning*, pp. 579–587, 2013.

Hazan, E. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

Hazan, E. and Seshadhri, C. Adaptive algorithms for online decision problems. *Electronic Colloquium on Computational Complexity*, 88, 2007.

Hazan, E. and Seshadhri, C. Efficient learning algorithms for changing environments. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 393–400, 2009.

Hazan, E., Agarwal, A., and Kale, S. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

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

Jadbabaie, A., Rakhlin, A., Shahrampour, S., and Sridharan, K. Online optimization: Competing with dynamic comparators. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, pp. 398–406, 2015.

Jun, K.-S., Orabona, F., Wright, S., and Willett, R. Improved strongly adaptive online learning using coin betting. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 943–951, 2017.

Littlestone, N. and Warmuth, M. K. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

Luo, H. and Schapire, R. E. Achieving all with no parameters: Adanormalhedge. In *Proceedings of The 28th Conference on Learning Theory*, pp. 1286–1304, 2015.

Mokhtari, A., Shahrampour, S., Jadbabaie, A., and Ribeiro, A. Online optimization in dynamic environments: Improved regret rates for strongly convex problems. In *Proceedings of the 55th IEEE Conference on Decision and Control*, pp. 7195–7201, 2016.

Orabona, F. and Pál, D. Scale-free online learning. *Theoretical Computer Science*, 716:50–69, 2018.

Orabona, F., Cesa-Bianchi, N., and Gentile, C. Beyond logarithmic bounds in online learning. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pp. 823–831, 2012.

Shalev-Shwartz, S. *Online Learning: Theory, Algorithms, and Applications*. PhD thesis, The Hebrew University of Jerusalem, 2007.

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

Srebro, N., Sridharan, K., and Tewari, A. Smoothness, low-noise and fast rates. In *Advances in Neural Information Processing Systems 23*, pp. 2199–2207, 2010.

Wang, G., Zhao, D., and Zhang, L. Minimizing adaptive regret with one gradient per iteration. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018.

Yang, T., Zhang, L., Jin, R., and Yi, J. Tracking slowly moving clairvoyant: Optimal dynamic regret of online learning with true and noisy gradient. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 449–457, 2016.

Zhang, L., Yang, T., Yi, J., Jin, R., and Zhou, Z.-H. Improved dynamic regret for non-degenerate functions. In *Advances in Neural Information Processing Systems 30*, pp. 732–741, 2017.

Zhang, L., Lu, S., and Zhou, Z.-H. Adaptive online learning in dynamic environments. In *Advances in Neural Information Processing Systems 31*, pp. 1330–1340, 2018a.

Zhang, L., Yang, T., Jin, R., and Zhou, Z.-H. Dynamic regret of strongly adaptive methods. In *Proceedings of the 35th International Conference on Machine Learning*, 2018b.

Zhang, L., Liu, T.-Y., and Zhou, Z.-H. Adaptive regret of convex and smooth functions. *ArXiv e-prints*, arXiv:1904.11681, 2019.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pp. 928–936, 2003.