Dual Adaptivity: A Universal Algorithm for Minimizing the Adaptive Regret of Convex Functions

Anonymous Author(s)
Affiliation
Address
email

Abstract

To deal with changing environments, a new performance measure—adaptive regret, defined as the maximum static regret over any interval, was proposed in online learning. Under the setting of online convex optimization, several algorithms have been successfully developed to minimize the adaptive regret. However, existing algorithms lack universality in the sense that they can only handle one type of convex functions and need apriori knowledge of parameters. By contrast, there exist universal algorithms, such as MetaGrad, that attain optimal static regret for multiple types of convex functions simultaneously. Along this line of research, this paper presents the *first* universal algorithm for minimizing the adaptive regret of convex functions. Specifically, we borrow the idea of maintaining multiple learning rates in MetaGrad to handle the uncertainty of functions, and utilize the technique of sleeping experts to capture changing environments. In this way, our algorithm automatically adapts to the property of functions (convex, exponentially concave, or strongly convex), as well as the nature of environments (stationary or changing). As a by product, it also allows the type of functions to switch between rounds.

1 Introduction

Online learning aims to make a sequence of accurate decisions given knowledge of answers to previous tasks and possibly additional information [Shalev-Shwartz, 2011]. It is performed in a sequence of consecutive rounds, where at round t the learner is asked to select a decision $\mathbf{w}_t$ from a domain Ω. After submitting the answer, a loss function $f_t : \Omega \mapsto \mathbb{R}$ is revealed and the learner suffers a loss $f_t(\mathbf{w}_t)$. The standard performance measure is the regret [Cesa-Bianchi and Lugosi, 2006]:

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

defined as the difference between the cumulative loss of the online learner and that of the best decision chosen in hindsight. When both the domain Ω and the loss $f_t(\cdot)$ are convex, it becomes online convex optimization (OCO) [Zinkevich, 2003].

In the literature, there exist plenty of algorithms to minimize the regret under the setting of OCO [Hazan, 2016]. However, when the environment undergoes many changes, regret may not be the best measure of performance. That is because regret chooses a fixed comparator, and for the same reason, it is also referred to as *static* regret. To address this limitation, Hazan and Seshadhri [2007] introduce the concept of adaptive regret, which measures the performance with respect to a changing comparator. Following the terminology of Daniely et al. [2015], we define the strongly adaptive

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

regret as the maximum static regret over intervals of length τ, i.e.,

$$\text{SA-Regret}(T,\tau) = \max_{[p,p+\tau-1]\subseteq[T]} \left(\sum_{t=p}^{p+\tau-1} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\Omega} \sum_{t=p}^{p+\tau-1} f_t(\mathbf{w}) \right). \tag{1}$$

Since the seminal work of Hazan and Seshadhri [2007], several algorithms have been successfully developed to minimize the adaptive regret of convex functions, including general convex, exponentially concave (abbr. exp-concave) and strongly convex functions [Hazan and Seshadhri, 2009, Jun et al., 2017a, Zhang et al., 2018]. However, existing methods can only handle one type of convex functions. Furthermore, when facing exp-concave and strongly convex functions, they need to know the moduli of exp-concavity and strong convexity. The lack of universality hinders their applications to real-world problems.

On the other hand, there do exist universal algorithms, such as MetaGrad [van Erven and Koolen, 2016], that attain optimal static regret for multiple types of convex functions simultaneously. This observation motivates us to ask whether it is possible to design a single algorithm to minimize the adaptive regret of multiple types of functions. This is very challenging because the algorithm needs to enjoy dual adaptivity, adaptive to the function type and adaptive to the environment. In this paper, we provide an affirmative answer by developing a Universal algorithm for Minimizing the Adaptive regret (UMA). First, inspired by MetaGrad, UMA maintains multiple learning rates to handle the uncertainty of functions. In this way, it supports multiple types of functions simultaneously and identifies the best learning rate automatically. Second, following existing studies on adaptive regret, UMA deploys sleeping experts [Freund et al., 1997] to minimize the regret over any interval, and thus achieves a small adaptive regret and captures the changing environment.

The main advantage of UMA is that it attains second-order regret bounds over any interval. As a result, it can minimize the adaptive regret of general convex functions, and automatically take advantage of easier functions whenever possible. Specifically, UMA attains $O(\sqrt{\tau \log T})$, $O(\frac{d}{\alpha} \log \tau \log T)$ and $O(\frac{1}{\lambda} \log \tau \log T)$ strongly adaptive regrets for general convex, α-exp-concave and λ-strongly convex functions respectively, where d is the dimensionality. All of these bounds match the state-of-the-art results on adaptive regret [Jun et al., 2017a, Zhang et al., 2018] exactly. Furthermore, UMA can also handle the case that the type of functions changes between rounds. For example, suppose the online functions are general convex during interval I_1, then become α-exp-concave in I_2, and finally switch to λ-strongly convex in I_3. When facing this function sequence, UMA achieves $O(\sqrt{|I_1| \log T})$, $O(\frac{d}{\alpha} \log |I_2| \log T)$ and $O(\frac{1}{\lambda} \log |I_3| \log T)$ regrets over intervals I_1, I_2 and I_3, respectively.

2 Related work

We briefly review related work on static regret and adaptive regret, under the setting of OCO.

2.1 Static regret

To minimize the static regret of general convex functions, online gradient descent (OGD) with step size $\eta_t = O(1/\sqrt{t})$ achieves an $O(\sqrt{T})$ bound [Zinkevich, 2003]. If all the online functions are λ-strongly convex, OGD with step size $\eta_t = O(1/[\lambda t])$ attains an $O(\frac{1}{\lambda} \log T)$ bound [Shalev-Shwartz et al., 2007]. When the functions are α-exp-concave, online Newton step (ONS), with knowledge of α, enjoys an $O(\frac{d}{\alpha} \log T)$ bound, where d is the dimensionality [Hazan et al., 2007]. These regret bounds are minimax optimal for the corresponding type of functions [Abernethy et al., 2008], but choosing the optimal algorithm for a specific problem requires domain knowledge.

The study of universal algorithms for OCO stems from the adaptive online gradient descent (AOGD) [Bartlett et al., 2008] and its proximal extension [Do et al., 2009]. The key idea of AOGD is to add a quadratic regularization term to the loss. Bartlett et al. [2008] demonstrate that AOGD is able to interpolate between the $O(\sqrt{T})$ bound of general convex functions and the $O(\log T)$ bound of strongly convex functions. Furthermore, it allows the online function to switch between general convex and strongly convex. However, AOGD has two restrictions:

- It needs to calculate the modulus of strong convexity on the fly, which is a nontrivial task;
- It does not support exp-concave functions explicitly, and thus can only achieve a suboptimal $O(\sqrt{T})$ regret for this type of functions.

Another milestone is the multiple eta gradient algorithm (MetaGrad) [van Erven and Koolen, 2016, Mhammedi et al., 2019], which adapts to a much broader class of functions, including convex and exp-concave functions. MetaGrad achieves $O(\sqrt{T \log \log T})$ and $O(\frac{d}{\alpha} \log T)$ regret bounds for general convex and α-exp-concave functions, respectively. However, it suffers the following two limitations:

- MetaGrad treats strongly convex functions as exp-concave, and thus only gives a suboptimal $O(\frac{d}{\lambda} \log T)$ regret for λ-strongly convex functions;
- It assumes the type of online functions, as well as the associated parameter, does not change between rounds.

The first limitation of MetaGrad has been addressed by Wang et al. [2019], who develop a universal algorithm named as multiple sub-algorithms and learning rates (Maler). It attains $O(\sqrt{T})$, $O(\frac{d}{\alpha} \log T)$ and $O(\frac{1}{\lambda} \log T)$ regret bounds for general convex, α-exp-concave, and λ-strongly convex functions, respectively. Furthermore, Wang et al. [2020] extend Maler to make use of smoothness. However, the second limitation remains there.

2.2 Adaptive regret

Adaptive regret has been studied in the setting of prediction with expert advice [Littlestone and Warmuth, 1994, Freund et al., 1997, Adamskiy et al., 2012, György et al., 2012, Luo and Schapire, 2015] and OCO. In this section, we focus on the related work in the latter one.

Adaptive regret is introduced by Hazan and Seshadhri [2007], and later refined by Daniely et al. [2015]. We refer to the definition of Hazan and Seshadhri as weakly adaptive regret:

$$\text{WA-Regret}(T) = \max_{[p,q] \subseteq [T]} \left(\sum_{t=p}^{q} f_t(\mathbf{w}_t) - \min_{\mathbf{w} \in \Omega} \sum_{t=p}^{q} f_t(\mathbf{w}) \right).$$

For α-exp-concave functions, Hazan and Seshadhri [2007] propose an adaptive algorithm named as Follow-the-Leading-History (FLH). FLH restarts a copy of ONS in each round as an expert, and chooses the best one using expert-tracking algorithms. The meta-algorithm used to track the best expert is inspired by the Fixed-Share algorithm [Herbster and Warmuth, 1998]. While FLH is equipped with an $O(\frac{d}{\alpha} \log T)$ weakly adaptive regret, it is computationally expensive since it needs to maintain t experts in the t-th iteration. To reduce the computational cost, Hazan and Seshadhri [2007] further prune the number of experts based on a data streaming algorithm. In this way, FLH only keeps $O(\log t)$ experts, at the price of an $O(\frac{d}{\alpha} \log^2 T)$ weakly adaptive regret.

For general convex functions, we can use OGD as the expert-algorithm in FLH. Hazan and Seshadhri [2007] prove that FLH and its efficient variant attain $O(\sqrt{T \log T})$ and $O(\sqrt{T \log^3 T})$ weakly adaptive regrets, respectively. This result reveals a limitation of weakly adaptive regret—it does not respect short intervals well. For example, the $O(\sqrt{T \log T})$ regret bound is meaningless for intervals of length $O(\sqrt{T})$. To address this limitation, Daniely et al. [2015] introduce the strongly adaptive regret which takes the interval length as a parameter, as shown in (1). They propose a novel meta-algorithm, named as Strongly Adaptive Online Learner (SAOL). SAOL also maintains $O(\log t)$ experts in the t-th round, and achieves an $O(\sqrt{\tau} \log T)$ strongly adaptive regret for convex functions. Later, Jun et al. [2017a] develop a new meta-algorithm named as sleeping coin betting (SCB), and improve the strongly adaptive regret bound to $O(\sqrt{\tau \log T})$. Very recently, Cutkosky [2020] proposes a novel strongly adaptive method, which can guarantee the $O(\sqrt{\tau \log T})$ bound in the worst case, while achieving tighter results when the square norms of gradients are small.

For λ-strongly convex functions, Zhang et al. [2018] point out that we can replace ONS with OGD, and obtain an $O(\frac{1}{\lambda} \log T)$ weakly adaptive regret. They also demonstrate that the number of active experts can be reduced from t to $O(\log t)$, at a cost of an additional $\log T$ factor in the regret.

3 Main results

We first present necessary preliminaries, including assumptions, definitions and the technical challenge, then provide our universal algorithm and its theoretical guarantee. Due to the limitation of space, we provide all the proofs in the supplementary.

3.1 Preliminaries

We introduce two common assumptions used in the study of OCO [Hazan, 2016].

Assumption 1 *The diameter of the domain Ω is bounded by D, i.e.,*

$$\max_{\mathbf{x},\mathbf{y}\in\Omega} \|\mathbf{x}-\mathbf{y}\| \leq D. \tag{2}$$

Assumption 2 *The gradients of all the online functions are bounded by G, i.e.,*

$$\max_{\mathbf{w}\in\Omega} \|\nabla f_t(\mathbf{w})\| \leq G, \ \forall t \in [T]. \tag{3}$$

Next, we state definitions of strong convexity and exp-concavity [Boyd and Vandenberghe, 2004, Cesa-Bianchi and Lugosi, 2006].

Definition 1 *A function $f : \Omega \mapsto \mathbb{R}$ is λ-strongly convex if*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle + \frac{\lambda}{2}\|\mathbf{y}-\mathbf{x}\|^2, \ \forall \mathbf{x},\mathbf{y} \in \Omega.$$

Definition 2 *A function $f : \Omega \mapsto \mathbb{R}$ is α-exp-concave if $\exp(-\alpha f(\cdot))$ is concave over Ω.*

The following property of exp-concave functions will be used later [Hazan et al., 2007, Lemma 3].

Lemma 1 *For a function $f : \Omega \mapsto \mathbb{R}$, where Ω has diameter D, such that $\forall \mathbf{w} \in \Omega$, $\|\nabla f(\mathbf{w})\| \leq G$ and $\exp(-\alpha f(\cdot))$ is concave, the following holds for $\beta = \frac{1}{2}\min\{\frac{1}{4GD}, \alpha\}$:*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle + \frac{\beta}{2}\langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle^2, \ \forall \mathbf{x},\mathbf{y} \in \Omega.$$

3.1.1 Technical challenge

Before introducing the proposed algorithm, we discuss the technical challenge of minimizing the adaptive regret of multiple types of convex functions simultaneously. All the existing adaptive algorithms [Hazan and Seshadhri, 2007, Daniely et al., 2015, Jun et al., 2017a] share the same framework and contain 3 components:

- An expert-algorithm, which is able to minimize the static regret of a specific type of function;
- 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.

To design a universal algorithm, a straightforward way is to use a universal method for static regret, such as MetaGrad [van Erven and Koolen, 2016], as the expert-algorithm. In this way, the expert-algorithm is able to handle the uncertainty of functions. However, the challenge lies in the design of the meta-algorithm, because the meta-algorithms used by previous studies also lack universality. For example, the meta-algorithm of Hazan and Seshadhri [2007] is able to deliver a tight meta-regret for exp-concave functions, but a loose one for general convex functions. Similarly, meta-algorithms of Daniely et al. [2015] and Jun et al. [2017a] incur at least $\Theta(\sqrt{\tau})$ meta-regret for intervals of length τ, which is tolerable for convex functions but suboptimal for exp-concave functions.

Instead of using MetaGrad as a black-box subroutine, we dig into this algorithm and modify it to minimize the adaptive regret directly. MetaGrad itself is a two-layer algorithm, which runs multiple expert-algorithms, each with a different learning rate, and combines them with a meta-algorithm named as Tilted Exponentially Weighted Average (TEWA). To address the aforementioned challenge, we extend TEWA to support sleeping experts so that it can minimize the adaptive regret. The advantage of TEWA is that its meta-regret only depends on the number of experts instead of the length of the interval, e.g., Lemma 4 of van Erven and Koolen [2016], and thus does not affect the optimality of the regret. The extension of TEWA to sleeping experts is the main technical contribution of this paper.

3.2 A parameter-free and adaptive algorithm for exp-concave functions

Recall that our goal is to design a universal algorithm for minimizing the adaptive regret of general convex, exp-concave, and strongly convex functions simultaneously. However, to facilitate understanding, we start with a simpler question: How to minimize the adaptive regret of exp-concave functions, without knowing the modulus of exp-concavity?

3.2.1 Review of MetaGrad

The reason that MetaGrad can minimize the regret of α-exp-concave functions without knowing the value of α is because it enjoys a second-order regret bound [Gaillard et al., 2014]:

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \leq \sum_{t=1}^{T} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle = O\left(\sqrt{V_T d \log T} + d \log T\right) \tag{4}$$

where $V_T = \sum_{t=1}^{T} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle^2$. Besides, Lemma 1 implies

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \leq \sum_{t=1}^{T} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle - \frac{\beta}{2} \sum_{t=1}^{T} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle^2. \tag{5}$$

Combining (4) with (5) and applying the AM-GM inequality, we immediately obtain

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) = O\left(\frac{d}{\beta} \log T\right) = O\left(\frac{d}{\alpha} \log T\right).$$

From the above discussion, it becomes clear that if we can establish a second-order regret bound for any interval $[p, q] \subseteq [T]$, we are able to minimize the adaptive regret even when α is unknown.

The way that MetaGrad attains the regret bound in (4) is to run a set of experts, each of which minimizes a surrogate loss parameterized by a learning rate η

$$\ell_t^{\eta}(\mathbf{w}) = -\eta \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle + \eta^2 \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle^2 \tag{6}$$

and then combine the outputs of experts by a meta-algorithm named as Tilted Exponentially Weighted Average (TEWA). Specifically, it creates an expert E^{η} for each η in

$$\mathcal{S}(T) = \left\{ \frac{2^{-i}}{5DG} \;\middle|\; i = 0, 1, \ldots, \left\lceil \frac{1}{2} \log_2 T \right\rceil \right\} \tag{7}$$

and thus maintains $1 + \lceil \frac{1}{2} \log_2 T \rceil = O(\log T)$ experts during the learning process. By simultaneously considering multiple learning rates, MetaGrad is able to deal with the uncertainty of V_T. Since the surrogate loss $\ell_t^{\eta}(\cdot)$ is exp-concave, a variant of ONS is used as the expert-algorithm. Let $\mathbf{w}_t^{\eta}$ be the output of expert E^{η} in the t-th round. MetaGrad calculates the final output $\mathbf{w}_t$ according to TEWA:

$$\mathbf{w}_t = \frac{\sum_{\eta} \pi_t^{\eta} \eta \mathbf{w}_t^{\eta}}{\sum_{\eta} \pi_t^{\eta} \eta} \tag{8}$$

where $\pi_t^{\eta} \propto \exp(-\sum_{i=1}^{t-1} \ell_i^{\eta}(\mathbf{w}_i^{\eta}))$.

3.2.2 Our approach

In this section, we discuss how to minimize the adaptive regret by extending MetaGrad. Following the idea of sleeping experts [Freund et al., 1997], the most straightforward way is to create $1 + \lceil \frac{1}{2} \log_2(q - p + 1) \rceil$ experts for each interval $[p, q] \in [T]$, and combine them with a meta-algorithm that supports sleeping experts. However, this simple approach is inefficient because the total number of experts is on the order of $O(T^2 \log T)$. To control the number of experts, we make use of the geometric covering (GC) intervals [Daniely et al., 2015] defined as

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

where

$$\mathcal{I}_k = \left\{ [i \cdot 2^k, (i+1) \cdot 2^k - 1] : i \in \mathbb{N} \right\}, \; k \in \mathbb{N} \cup \{0\}.$$

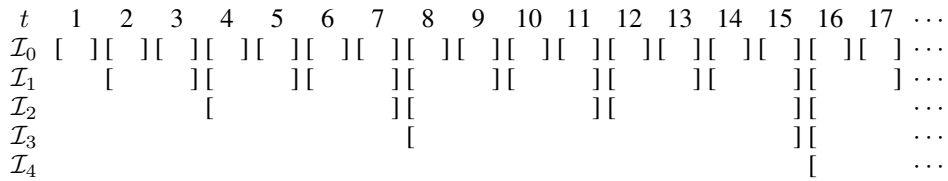


Figure 1: Geometric covering (GC) intervals of Daniely et al. [2015].

Algorithm 1 A Parameter-free and Adaptive algorithm for Exp-concave functions (PAE)

1: $\mathcal{A}_0 = \emptyset$
2: **for** $t = 1$ **to** T **do**
3: **for all** $I \in \mathcal{I}$ that starts from t **do**
4: **for all** $\eta \in \mathcal{S}(|I|)$ **do**
5: Create an expert E_I^η by running the slave algorithm of MetaGrad to minimize $\ell_t^\eta(\cdot)$ during I, and set $L_{t-1,I}^\eta = 0$
6: Add E_I^η to the set of active experts: $\mathcal{A}_t = \mathcal{A}_{t-1} \cup \{E_I^\eta\}$
7: **end for**
8: **end for**
9: Receive output $\mathbf{w}_{t,J}^\eta$ from each expert $E_J^\eta \in \mathcal{A}_t$
10: Submit $\mathbf{w}_t$ in (9)
11: Observe the loss $f_t(\cdot)$ and evaluate the gradient $\nabla f_t(\mathbf{w}_t)$
12: **for all** $E_J^\eta \in \mathcal{A}_t$ **do**
13: Update $L_{t,J}^\eta = L_{t-1,J}^\eta + \ell_t^\eta(\mathbf{w}_{t,J}^\eta)$
14: Pass the surrogate loss $\ell_t^\eta(\cdot)$ to expert E_J^η
15: **end for**
16: Remove experts whose ending times are t from $\mathcal{A}_t$
17: **end for**

A graphical illustration of GC intervals is given in Fig. 1. We observe that each $\mathcal{I}_k$ is a partition of $\mathbb{N} \setminus \{1, \cdots, 2^k - 1\}$ to consecutive intervals of length 2^k. Note that GC intervals can be generated on the fly, so we do not need to fix the horizon T.

Then, we only focus on intervals in $\mathcal{I}$. For each interval $I = [r, s] \in \mathcal{I}$, we will create $1 + \lceil \frac{1}{2} \log_2(s - r + 1) \rceil$ experts, each of which minimizes one surrogate loss in $\{\ell_t^\eta(\mathbf{w}) | \eta \in \mathcal{S}(s - r + 1)\}$ during I, where $\ell_t^\eta(\cdot)$ and $\mathcal{S}(\cdot)$ are defined in (6) and (7), respectively. These experts become active in round r and will be removed forever after round s. Since the number of intervals that contain t is $\lfloor \log_2 t \rfloor + 1$ [Daniely et al., 2015], the number of active experts is at most

$$(\lfloor \log_2 t \rfloor + 1) \left(1 + \left\lceil \frac{1}{2} \log_2 t \right\rceil\right) = O(\log^2 t).$$

So, the number of active experts is larger than that of MetaGrad by a logarithmic factor, which is the price paid in computations for the adaptivity to every interval.

Finally, we need to specify how to combine the outputs of active experts. From the construction of the surrogate loss in (6), we observe that each expert uses its own loss, which is very different from the traditional setting of prediction with expert advice in which all experts use the same loss. In the literature, the setting that each expert uses a loss function that may be different from the loss functions used by the other experts has been studied by Chernov and Vovk [2009], who name it as prediction with expert evaluators' advice. Furthermore, they have proposed a general conversion to the sleeping expert case by only considering active rounds in the calculation of the cumulative losses. Following this idea, we extend TEWA to sleeping experts.

Our Parameter-free and Adaptive algorithm for Exp-concave functions (PAE) is summarized in Algorithm 1. In the t-th round, we first create an expert E_I^η for each interval $I \in \mathcal{I}$ that starts from t and each $\eta \in \mathcal{S}(|I|)$, where $\mathcal{S}(\cdot)$ is defined in (7), and introduce a variable $L_{t-1,I}^\eta$ to record the cumulative loss of E_I^η (Step 5). The expert E_I^η is the slave algorithm of MetaGrad [van Erven and Koolen, 2016] that minimizes $\ell_t^\eta(\cdot)$ during interval I. We also maintain a set $\mathcal{A}_t$ consisting of all the active experts (Step 6). Denote the prediction of expert E_J^η at round t as $\mathbf{w}_{t,J}^\eta$. In Step 9, PAE

collects the predictions of all the active experts, and then submits the following solution in Step 10:

$$\mathbf{w}_t = \frac{1}{\sum_{E_J^\eta \in \mathcal{A}_t} \exp(-L_{t-1,J}^\eta)\eta} \sum_{E_J^\eta \in \mathcal{A}_t} \exp(-L_{t-1,J}^\eta)\eta \mathbf{w}_{t,J}^\eta. \tag{9}$$

Compared with TEWA in (8), we observe that (9) focuses on active experts and ignores inactive ones. Although the extension is inspired by Chernov and Vovk [2009], our analysis is different because the surrogate losses take a special form of (6). In Step 11, PAE observes the loss $f_t(\cdot)$ and evaluates the gradient $\nabla f_t(\mathbf{w}_t)$ to construct the surrogate loss. In Step 13, it updates the cumulative loss of each active expert, and in Step 14 passes the surrogate loss to each expert such that it can make predictions for the next round. In Step 16, PAE removes experts whose ending times are t from $\mathcal{A}_t$.

Due to limitations of space, we provide the expert-algorithm in the supplementary. The theoretical guarantee of PAE is given below.

Theorem 1 *Under Assumptions 1 and 2, for any interval $[p,q] \subseteq [T]$ and any $\mathbf{w} \in \Omega$, PAE satisfies*

$$\sum_{t=p}^{q} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle \leq 10DGa(p,q)b(p,q) + 3\sqrt{a(p,q)b(p,q)} \sqrt{\sum_{t=p}^{q} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle^2}$$

where

$$a(p,q) = 2\log_2(2q) + \frac{1}{2} + \frac{d}{2} \ln\left(1 + \frac{2}{25d}(q-p+1)\right), \tag{10}$$

$$b(p,q) = 2\lceil \log_2(q-p+2) \rceil. \tag{11}$$

Furthermore, if all the online functions are α-exp-concave, we have

$$\sum_{t=p}^{q} f_t(\mathbf{w}_t) - \sum_{t=p}^{q} f_t(\mathbf{w}) \leq \left(10DG + \frac{9}{2\beta}\right) a(p,q)b(p,q) = O\left(\frac{d \log q \log(q-p)}{\alpha}\right)$$

where $\beta = \frac{1}{2}\min\{\frac{1}{4GD}, \alpha\}$.

Remark Theorem 1 indicates that PAE enjoys a second-order regret bound for any interval, which in turn implies a small regret for exp-concave functions. Specifically, for α-exp-concave functions, PAE satisfies $\text{SA-Regret}(T,\tau) = O(\frac{d}{\alpha}\log\tau\log T)$, which matches the regret of efficient FLH [Hazan and Seshadhri, 2007]. This is a remarkable result given the fact that PAE is *agnostic* to α.

3.3 A universal algorithm for minimizing the adaptive regret

In this section, we extend PAE to support strongly convex functions and general convex functions. Inspired by Wang et al. [2020], we introduce a new surrogate loss to handle strong convexity:

$$\hat{\ell}_t^\eta(\mathbf{w}) = -\eta\langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle + \eta^2 \|\nabla f_t(\mathbf{w}_t)\|^2 \|\mathbf{w}_t - \mathbf{w}\|^2 \tag{12}$$

which is also parameterized by $\eta > 0$. Our goal is to attain another second-order type of regret bound

$$\sum_{t=p}^{q} f_t(\mathbf{w}_t) - \sum_{t=p}^{q} f_t(\mathbf{w}) \leq \sum_{t=p}^{q} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle = \widetilde{O}\left(\sqrt{\sum_{t=p}^{q} \|\mathbf{w}_t - \mathbf{w}\|^2}\right) \tag{13}$$

for any interval $[p,q] \subseteq T$. Combining (13) with Definition 1, we can establish a tight regret bound for λ-strongly convex functions over any interval without knowing the value of λ. Furthermore, we can also derive a regret bound for general convex functions from (13), and there is no need to add additional surrogate losses.

Our Universal algorithm for Minimizing the Adaptive regret (UMA) is summarized in Algorithm 2. UMA is a natural extension of PAE by incorporating the new surrogate loss $\hat{\ell}_t^\eta(\cdot)$. The overall procedure of UMA is very similar to PAE, except that the number of experts doubles and the weighting formula is modified accordingly. Specifically, in each round t, we further create an expert $\widehat{E}_I^\eta$ for each interval $I \in \mathcal{I}$ that starts from t and each $\eta \in \mathcal{S}(|I|)$. $\widehat{E}_I^\eta$ is an instance of AOGD

Algorithm 2 A Universal algorithm for Minimizing the Adaptive regret (UMA)

1: $\mathcal{A}_0 = \widehat{\mathcal{A}}_0 = \emptyset$
2: **for** $t = 1$ **to** T **do**
3: **for all** $I \in \mathcal{I}$ that starts from t **do**
4: **for all** $\eta \in \mathcal{S}(|I|)$ **do**
5: Create an expert E_I^η by running the slave algorithm of MetaGrad to minimize $\ell_t^\eta(\cdot)$ during I, and set $L_{t-1,I}^\eta = 0$
6: Add E_I^η to the set of active experts: $\mathcal{A}_t = \mathcal{A}_{t-1} \cup \{E_I^\eta\}$
7: Create an expert $\widehat{E}_I^\eta$ by running an instance of AOGD to minimize $\hat{\ell}_t^\eta(\cdot)$ during I, and set $\widehat{L}_{t-1,I}^\eta = 0$
8: Add $\widehat{E}_I^\eta$ to the set of active experts: $\widehat{\mathcal{A}}_t = \widehat{\mathcal{A}}_{t-1} \cup \{\widehat{E}_I^\eta\}$
9: **end for**
10: **end for**
11: Receive output $\mathbf{w}_{t,J}^\eta$ from each expert $E_J^\eta \in \mathcal{A}_t$ and $\widehat{\mathbf{w}}_{t,J}^\eta$ from each expert $\widehat{E}_J^\eta \in \widehat{\mathcal{A}}_t$
12: Submit $\mathbf{w}_t$ in (14)
13: Observe the loss $f_t(\cdot)$ and evaluate the gradient $\nabla f_t(\mathbf{w}_t)$
14: **for all** $E_J^\eta \in \mathcal{A}_t$ **do**
15: Update $L_{t,J}^\eta = L_{t-1,J}^\eta + \ell_t^\eta(\mathbf{w}_{t,J}^\eta)$
16: Pass the surrogate loss $\ell_t^\eta(\cdot)$ to expert E_J^η
17: **end for**
18: **for all** $\widehat{E}_J^\eta \in \widehat{\mathcal{A}}_t$ **do**
19: Update $\widehat{L}_{t,J}^\eta = \widehat{L}_{t-1,J}^\eta + \hat{\ell}_t^\eta(\mathbf{w}_{t,J}^\eta)$
20: Pass the surrogate loss $\hat{\ell}_t^\eta(\cdot)$ to expert $\widehat{E}_J^\eta$
21: **end for**
22: Remove experts whose ending times are t from $\mathcal{A}_t$ and $\widehat{\mathcal{A}}_t$
23: **end for**

[Bartlett et al., 2008] that is able to minimize $\hat{\ell}_t^\eta$ during interval I. We use $\widehat{L}_{t-1,I}^\eta$ to represent the cumulative loss of $\widehat{E}_I^\eta$ till round $t-1$, and $\widehat{\mathcal{A}}_t$ to store all the active $\widehat{E}_I^\eta$'s. Denote the prediction of expert $\widehat{E}_J^\eta$ at round t as $\widehat{\mathbf{w}}_{t,J}^\eta$. In Step 11, UMA receives predictions from experts in $\mathcal{A}_t$ and $\widehat{\mathcal{A}}_t$, and submits the following solution in Step 12:

$$\mathbf{w}_t = \frac{\sum_{E_J^\eta \in \mathcal{A}_t} \exp(-L_{t-1,J}^\eta)\eta \mathbf{w}_{t,J}^\eta + \sum_{\widehat{E}_J^\eta \in \widehat{\mathcal{A}}_t} \exp(-\widehat{L}_{t-1,J}^\eta)\eta \widehat{\mathbf{w}}_{t,J}^\eta}{\sum_{E_J^\eta \in \mathcal{A}_t} \exp(-L_{t-1,J}^\eta)\eta + \sum_{\widehat{E}_J^\eta \in \widehat{\mathcal{A}}_t} \exp(-\widehat{L}_{t-1,J}^\eta)\eta} \tag{14}$$

which is an extension of (9) to accommodate more experts.

Due to limitations of space, we provide the expert-algorithm, i.e., AOGD, in the supplementary. Our analysis shows that UMA inherits the theoretical guarantee of PAE, and meanwhile is able to minimize the adaptive regret of general convex and strongly convex functions.

Theorem 2 *Under Assumptions 1 and 2, for any interval $[p, q] \subseteq [T]$ and any $\mathbf{w} \in \Omega$, UMA enjoys the theoretical guarantee of PAE in Theorem 1. Besides, it also satisfies*

$$\sum_{t=p}^{q} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle \leq 10DG\hat{a}(p,q)b(p,q) + 3G\sqrt{\hat{a}(p,q)b(p,q)}\sqrt{\sum_{t=p}^{q} \|\mathbf{w}_t - \mathbf{w}\|^2}, \tag{15}$$

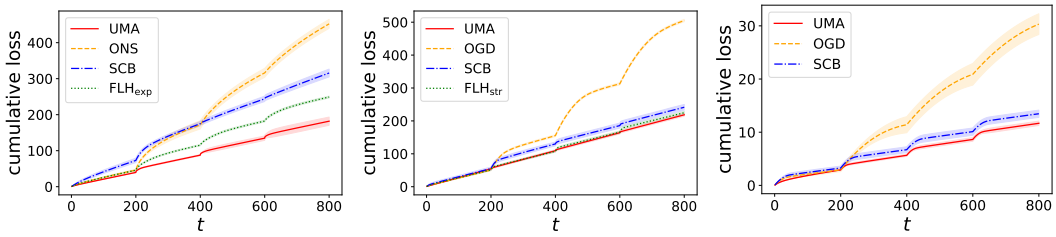
$$\sum_{t=p}^{q} \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle \leq 10DG\hat{a}(p,q)b(p,q) + 21DG\sqrt{\hat{a}(p,q)(q-p+1)} \tag{16}$$

where $b(\cdot,\cdot)$ is given in (11), and

$$\hat{a}(p,q) = 1 + 2\log_2(2q) + \log(q-p+2). \tag{17}$$

Furthermore, if all the online functions are λ-strongly convex, we have

$$\sum_{t=p}^{q} f_t(\mathbf{w}_t) - \sum_{t=p}^{q} f_t(\mathbf{w}) \leq \left(10DG + \frac{9G^2}{2\lambda}\right)\hat{a}(p,q)b(p,q) = O\left(\frac{\log q \log(q-p)}{\lambda}\right).$$



(a) Exp-concave functions (b) Strongly convex functions (c) Convex Functions

Figure 2: Cumulative losses of different methods versus the number of iterations.

Remark First, (15) shows that UMA is equipped with another second-order regret bound for any interval, leading to a small regret for strongly convex functions. Specifically, for λ-strongly convex functions, UMA achieves $\text{SA-Regret}(T, \tau) = O(\frac{1}{\lambda} \log \tau \log T)$, which matches the regret of the efficient algorithm of Zhang et al. [2018]. Second, (16) manifests that UMA attains an $O(\sqrt{\tau \log T})$ strongly adaptive regret for general convex functions, which again matches the state-of-the-art result of Jun et al. [2017a] exactly. Finally, because of the dual adaptivity, UMA can handle the tough case that the type of functions switches or the parameter of functions changes.

4 Experiments

We present empirical results to evaluate the proposed UMA. Due to the limitation of space, we only provide partial results, and the full details can be found in the supplementary.

First, we focus on exp-concave functions, and perform online classification on the ijcnn1 dataset [Chang and Lin, 2011, Prokhorov, 2001], where the logistic loss is used. In each round, we sample a mini-batch of examples to construct the loss, and flip the labels per 200 rounds to simulate changing environments. We compare UMA with FLH for exp-concave functions (abbr. FLH$_{\text{exp}}$) [Hazan and Seshadhri, 2007], ONS [Hazan et al., 2007], and SCB [Jun et al., 2017a]. Notice that (i) both FLH$_{\text{exp}}$ and ONS need to know the modulus of exp-concavity beforehand; and (ii) SCB can be applied because exp-concave functions are also convex. Second, we consider strongly convex functions, and follow the above setting but choose the regularized hinge loss. We compare UMA with FLH for strongly convex functions (abbr. FLH$_{\text{str}}$) [Zhang et al., 2018], OGD [Shalev-Shwartz et al., 2007], and SCB. Similarly, both FLH$_{\text{str}}$ and OGD require the modulus of strong convexity as their input. Finally, we investigate convex functions, and implement online linear regression on synthetic data, where the absolute loss is chosen. We generate the data in a way such that the underlying parameter changes every 200 rounds, and compare UMA with SCB, and OGD [Zinkevich, 2003].

All the experiments are done on a personal laptop. We report the cumulative losses of different methods in Fig. 2. As can be seen, UMA performs nearly the best in all cases. Specifically, UMA is better than FLH$_{\text{exp}}$ in Fig. 2(a) and very close to FLH$_{\text{str}}$ in Fig. 2(b), which indicates that UMA can estimate the moduli of exp-concavity and strong convexity automatically. We also observe that the cumulative losses of ONS and OGD, which are designed to minimize the static regret, increase rapidly over time, especially when the optimal model changes.

5 Conclusion and future work

In this paper, we develop a universal algorithm that is able to minimize the adaptive regret of general convex, exp-concave and strongly convex functions simultaneously. For each type of functions, our theoretical guarantee matches the performance of existing algorithms specifically designed for this type of function under apriori knowledge of parameters.

Recent studies [Jun et al., 2017b, Zhang et al., 2019] have demonstrated that smoothness can be exploited to improve the adaptive regret, in analogy to the way that smoothness helps tighten the static regret [Srebro et al., 2010]. In the future, we will investigate whether our universal algorithm for minimizing the adaptive regret can be extended to support smoothness.

References

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

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

P. L. Bartlett, E. Hazan, and A. Rakhlin. Adaptive online gradient descent. In *Advances in Neural Information Processing Systems 20*, pages 65–72, 2008.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

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

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011.

A. Chernov and V. Vovk. Prediction with expert evaluators' advice. In *Proceedings of the 20th International Conference on Algorithmic Learning Theory*, pages 8–22, 2009.

A. Cutkosky. Parameter-free, dynamic, and strongly-adaptive online learning. In *Proceedings of the 37th International Conference on Machine Learning*, pages 2250–2259, 2020.

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

C. Do, Q. Le, and C.-S. Foo. Proximal regularization for online and batch learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 257–264, 2009.

Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 334–343, 1997.

P. Gaillard, G. Stoltz, and T. van Erven. A second-order bound with excess losses. In *Proceedings of the 27th Conference on Learning Theory*, pages 176–196, 2014.

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

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

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

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

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

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

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

K.-S. Jun, F. Orabona, S. Wright, and R. Willett. Online learning for changing environments using coin betting. *Electronic Journal of Statistics*, 11(2):5282–5310, 2017b.

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

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

Z. Mhammedi, W. M. Koolen, and T. Van Erven. Lipschitz adaptivity with multiple learning rates in online learning. In *Proceedings of the 32nd Conference on Learning Theory*, pages 2490–2511, 2019.

D. Prokhorov. IJCNN 2001 neural network competition. Technical report, Ford Research Laboratory, 2001.

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

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: primal estimated sub-gradient solver for SVM. In *Proceedings of the 24th International Conference on Machine Learning*, pages 807–814, 2007.

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

T. van Erven and W. M. Koolen. MetaGrad: Multiple learning rates in online learning. In *Advances in Neural Information Processing Systems 29*, pages 3666–3674, 2016.

G. Wang, S. Lu, and L. Zhang. Adaptivity and optimality: A universal algorithm for online convex optimization. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*, 2019.

G. Wang, S. Lu, Y. Hu, and L. Zhang. Adapting to smoothness: A more universal algorithm for online convex optimization. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 6162–6169, 2020.

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

L. Zhang, T.-Y. Liu, and Z.-H. Zhou. Adaptive regret of convex and smooth functions. In *Proceedings of the 36th International Conference on Machine Learning*, pages 7414–7423, 2019.

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

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] On Page 6, we have pointed out that the number of active experts of our algorithm is larger than that of MetaGrad by a logarithmic factor, which is the main limitation of this work.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] This work is mainly theoretical.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] All the proofs can be found in the supplementary.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We include them in the supplementary.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We specify all the details in the supplementary.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We have reported error bars in the figures.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] All the experiments are performed on a personal laptop.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] We use the ijcnn1 data set from LIBSVM data, and cite the creators in Section 4.

(b) Did you mention the license of the assets? [No]

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] In the supplementary, we include new code implemented by us.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] The ijcnn1 data set is publicly available.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] But we think it is safe to use the ijcnn1 data set since it is so popular.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]