Adaptivity and Optimality: A Universal Algorithm for Online Convex Optimization

Guanghui Wang, Shiyin Lu, Lijun Zhang
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
{wanggh,lusy,zhanglj}@lamda.nju.edu.cn

Abstract

In this paper, we study adaptive online convex optimization, and aim to design a universal algorithm that achieves optimal regret bounds for multiple common types of loss functions. Existing universal methods are limited in the sense that they are optimal for only a subclass of loss functions. To address this limitation, we propose a novel online algorithm, namely Maler, which enjoys the optimal $O(\sqrt{T})$, $O(d \log T)$ and $O(\log T)$ regret bounds for general convex, exponentially concave, and strongly convex functions respectively. The essential idea is to run multiple types of learning algorithms with different learning rates in parallel, and utilize a meta-algorithm to track the best on the fly. Empirical results demonstrate the effectiveness of our method.

1 INTRODUCTION

Online convex optimization (OCO) is a well-established paradigm for modeling sequential decision making (Shalev-Shwartz et al., 2012). The protocol of OCO is as follows: in each round t, firstly a learner chooses an action $\mathbf{x}_t$ from a convex set $\mathcal{D} \subseteq \mathbb{R}^d$, at the same time, an adversary reveals a loss function $f_t : \mathcal{D} \mapsto \mathbb{R}$, and consequently the learner suffers a loss $f_t(\mathbf{x}_t)$. The goal is to minimize regret, defined as the difference between the cumulative loss of the learner and that of the best action in hindsight (Hazan et al., 2016):

$$R(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{D}} \sum_{t=1}^{T} f_t(\mathbf{x}). \tag{1}$$

There exist plenty of algorithms for OCO, based on different assumptions on the loss functions. Without any assumptions beyond convexity and Lipschitz continuity, the classic online gradient descent (OGD) with step size on the order of $O(1/\sqrt{t})$ (referred to as convex OGD) guarantees an $O(\sqrt{T})$ regret bound (Zinkevich, 2003), where T is the time horizon. While it has been proved minimax optimal for arbitrary convex functions (Abernethy et al., 2008), tighter bounds are still achievable when loss functions are known to fall into some easier categories *in advance*. Specifically, for *strongly* convex functions, OGD with step size proportional to $O(1/t)$ (referred to as strongly convex OGD) achieves an $O(\log T)$ regret bound (Hazan et al., 2007); for *exponentially concave* functions, the state-of-the-art algorithm is online Newton step (ONS) (Hazan et al., 2007), which enjoys an $O(d \log T)$ regret bound, where d is the dimensionality.

This divides OCO into subclasses, relying on users to decide which algorithm to use for their specific settings. Such requirements, not only are a burden to users, but also hinder the applications to broad domains where the types of loss functions are unknown and choosing the right algorithm beforehand is impossible. These issues motivate the innovation of *adaptive* algorithms, which aim to guarantee optimal regret bounds for arbitrary convex functions, and automatically exploit easier functions whenever possible. The seminal work of Hazan et al. (2008) propose adaptive online gradient descent (AOGD), which attains $O(\sqrt{T})$ and $O(\log T)$ regret bounds for convex and strongly convex functions respectively. However, AOGD requires the curvature information of f_t as input in each round, and fails to provide logarithmic regret bound for exponentially concave functions. Another milestone is MetaGrad (van Erven and Koolen, 2016), which only requires the gradient information, and achieves $O(\sqrt{T \log\log T})$ and $O(d \log T)$ regret bounds for convex and exponentially concave functions respectively. Although it also implies an $O(d \log T)$ regret for strongly convex functions, there still exists a large $O(d)$ gap from the $\Omega(\log T)$ lower bound (Abernethy et al., 2008).

Along this line of research, it is therefore natural to ask whether both adaptivity and optimality can be attained

simultaneously, or there is an inevitable price in regret to be paid for adaptivity, which was also posed as an open question by van Erven and Koolen (2016). In this paper, we give an affirmative answer by developing a novel online method, namely Maler, which achieves the optimal regret bounds for all aforementioned three types of loss functions. Inspired by MetaGrad, our method runs multiple expert algorithms in parallel, each with a different learning rate, and combines them with a meta-algorithm that learns the empirically best for the OCO problem in hand. However, different from MetaGrad where experts are the same type of OCO algorithms (i.e., a variant of ONS), experts in Maler consists of various types of OCO algorithms (i.e., convex OGD, ONS and strongly convex OGD). Essentially, the goal of MetaGrad is to learn only the optimal learning rate. In contrast, Maler searches for the best OCO algorithm and the optimal learning rate simultaneously. Theoretical analysis shows that, with $O(\log T)$ experts, which is on the same order as that of MetaGrad, Maler achieves $O(\sqrt{T})$, $O(d \log T)$ and $O(\log T)$ regret bounds for convex, exponentially concave and strongly convex functions respectively. Empirical results on both synthetic and real-world datasets demonstrate the advantages of our method.

Notation. Throughout the paper, we use lower case bold face letters to denote vectors, lower case letters to denote scalars, and upper case letters to denote matrices. We use $\|\cdot\|$ to denote the ℓ_2-norm. For a positive definite matrix $H \in \mathbb{R}^{d \times d}$, the weighted ℓ_2-norm is denoted by $\|\mathbf{x}\|_H^2 = \mathbf{x}^\top H \mathbf{x}$. The H-weighted projection $\Pi_{\mathcal{D}}^H(\mathbf{x})$ of $\mathbf{x}$ onto $\mathcal{D}$ is defined as $\Pi_{\mathcal{D}}^H(\mathbf{x}) = \operatorname{argmin}_{\mathbf{y} \in \mathcal{D}} \|\mathbf{y} - \mathbf{x}\|_H^2$. We denote the gradient of f_t at $\mathbf{x}_t$ as $\mathbf{g}_t$, and the best action in hindsight as $\mathbf{x}_* = \max_{\mathbf{x} \in \mathcal{D}} \sum_{t=1}^T f_t(\mathbf{x})$.

2 RELATED WORK

In the literature, there exist various of algorithms for OCO targeting on a specific type of loss functions. For general convex and strongly convex loss functions, the classic OGD with step size on the order of $O(1/\sqrt{t})$ and $O(1/t)$ achieve $O(\sqrt{T})$ and $O(\log T)$ regret bounds, respectively (Zinkevich, 2003; Hazan et al., 2007). For exponentially concave functions, online Newton step (ONS) attains a regret bound of $O(d \log T)$ (Hazan et al., 2007). The above bounds are known to be minimax optimal as matching lower bounds have been established (Abernethy et al., 2008).

To simultaneously deal with multiple types of loss functions, Hazan et al. (2008) propose adaptive online gradient descent (AOGD), which is later extended to proximal settings by Do et al. (2009). Both algorithms can achieve $O(\sqrt{T})$ and $O(\log T)$ regret bounds for convex and strongly convex loss functions respectively. Moreover, they have shown superiority over non-adaptive methods in the experiments (Do et al., 2009). However, in each round t these algorithms have to be fed with a parameter which depends on the curvature information of $f_t(\cdot)$ at $\mathbf{x}_t$, and cannot achieve the logarithmic regret bound for exponentially concave cases. To address these limitations, van Erven and Koolen (2016) propose the multiple eta gradient (MetaGrad), whose basic idea is to run a bunch of variant of ONS algorithms with different learning rates simultaneously, and employ a meta-algorithm to learn the best adaptively based on the empirical performances. They show that the regret of MetaGrad for arbitrary convex functions can be simultaneously bounded by a worst-case bound of $O(\sqrt{T \log \log T})$, and a data-dependant bound of $O(\sqrt{V_T d \log T} + d \log T)$, where $V_T = \sum_{t=1}^T ((\mathbf{x}_* - \mathbf{x}_t)^\top \mathbf{g}_t))^2$. In particular, for strongly convex and exponentially concave functions, the data-dependant bound reduces to $O(d \log T)$.

The above works as well as this paper focus on adapting to different types of loss functions. A related but parallel direction is adapting to structures in *data*, such as low-rank and sparsity. This line of research includes Adagrad (Duchi et al., 2011), RMSprop (Tieleman and Hinton, 2012), and Adam (Reddi et al., 2018), to name a few. The main idea here is to utilize the gradients observed over time to dynamically adjust the learning rate or the update direction of gradient descent, and their regret bounds depend on the cumulation of gradients. For general convex functions, the bounds attain $O(\sqrt{T})$ in the worst-case, and become tighter when the gradients are sparse.

Another different direction considers adapting to *changing environments*, where some more stringent criteria are established to measure the performance of algorithms, such as dynamic regret (Zinkevich, 2003; Hall and Willett, 2013; Zhang et al., 2017, 2018a), which compares the cumulative loss of the learner against any sequence of comparators, and adaptive regret (Hazan and Seshadhri, 2007; Daniely et al., 2015; Jun et al., 2017; Wang et al., 2018; Zhang et al., 2018b, 2019), which is defined as the maximum regret over any contiguous time interval. In this paper we mainly focus on the minimization of regret, and it an interesting question to explore whether our method can be extended to adaptive and dynamic regrets.

3 MALER

In this section, we first state assumptions made in this paper, then provide our motivations, and finally present the proposed algorithm as well as its theoretical guarantees.

3.1 ASSUMPTIONS AND DEFINITIONS

Following previous studies, we introduce some standard assumptions (van Erven and Koolen, 2016) and definitions (Boyd and Vandenberghe, 2004).

Assumption 1. *The gradients of all loss functions are bounded by G, i.e., $\forall t > 0, \max_{\mathbf{x}\in\mathcal{D}} \|\nabla f_t(\mathbf{x})\| \leq G$.*

Assumption 2. *The diameter of the action set is bounded by D, i.e., $\max_{\mathbf{x}_1,\mathbf{x}_2\in\mathcal{D}} \|\mathbf{x}_1 - \mathbf{x}_2\| \leq D$.*

Definition 1. *A function $f : \mathcal{D} \mapsto \mathbb{R}$ is convex if*

$$f(\mathbf{x}_1) - f(\mathbf{x}_2) + \nabla f(\mathbf{x}_2)^\top(\mathbf{x}_1 - \mathbf{x}_2), \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{D}. \tag{2}$$

Definition 2. *A function $f : \mathcal{D} \mapsto \mathbb{R}$ is λ-strongly convex if $\forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{D}$,*

$$f(\mathbf{x}_1) - f(\mathbf{x}_2) + \nabla f(\mathbf{x}_2)^\top(\mathbf{x}_1 - \mathbf{x}_2) + \frac{\lambda}{2}\|\mathbf{x}_1 - \mathbf{x}_2\|^2.$$

Definition 3. *A function $f : \mathcal{D} \mapsto \mathbb{R}$ is α-exponentially concave (abbreviated to α-exp-concave) if $\exp(-\alpha f(\mathbf{x}))$ is concave.*

3.2 MOTIVATION

Our algorithm is inspired by MetaGrad. To help understanding, we first give a brief introduction to the intuition behind this algorithm. Specifically, MetaGrad introduces the following surrogate loss function, parameterized by $\eta \in (0, \frac{1}{5DG}]$:

$$\ell_t(\mathbf{x}) = -\eta(\mathbf{x}_t - \mathbf{x})^\top \mathbf{g}_t + \eta^2(\mathbf{x} - \mathbf{x}_t)^\top \mathbf{g}_t\mathbf{g}_t^\top(\mathbf{x} - \mathbf{x}_t). \tag{3}$$

The first advantage of the above definition is that ℓ_t is 1-exp-concave. Therefore, we can apply ONS on ℓ_t and obtain the following regret bound with respect to ℓ_t:

$$\sum_{t=1}^{T} \ell_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{D}} \sum_{t=1}^{T} \ell_t(\mathbf{x}) \leq O(d \log T). \tag{4}$$

The second advantage is that the regret with respect to the original loss function f_t can be upper bounded in terms of the regret with respect to the defined surrogate loss function ℓ_t:

$$R(T) \leq \frac{\sum_{t=1}^{T} \ell_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{D}} \sum_{t=1}^{T} \ell_t(\mathbf{x})}{\eta} + \eta V_T^{\diamond} \tag{5}$$

where $V_T^{\diamond} = \sum_{t=1}^{T}((\mathbf{x}_t - \mathbf{x}_*)^\top \mathbf{g}_t)^2$. Both advantages jointly (i.e., combining (4) and (5)) lead to a regret bound of $O((d \log T)/\eta + \eta V_T^{\diamond})$. Therefore, had we known the value of $V_T^{\diamond}$ in advance, we could set η as $\min\{\sqrt{d \log T / V_T^{\diamond}}, \frac{1}{5DG}\}$ and obtain a regret bound of order $O(\sqrt{dV_T^{\diamond} \log T} + d \log T)$. However, this is impossible since $V_T^{\diamond}$ depends on the whole learning process. To sidestep this obstacle, MetaGrad maintains multiple ONS in parallel each of which targets minimizing the regret with respect to the surrogate loss ℓ_t with a different η, and employs a meta-algorithm to track the ONS with the best η. Theoretical analysis shows that MetaGrad achieves the desired $O(\sqrt{dV_T^{\diamond} \log T} + d \log T)$ bound.

While the $O(\sqrt{dV_T^{\diamond} \log T} + d \log T)$ regret bound of MetaGrad can reduce to $O(d \log T)$ for exp-concave functions, it can not recover the $O(\log T)$ regret bound for strongly convex functions. To address this limitation, we design a new surrogate loss function:

$$s_t(\mathbf{x}) = -\eta(\mathbf{x}_t - \mathbf{x})^\top \mathbf{g}_t + \eta^2 G^2 \|\mathbf{x}_t - \mathbf{x}\|^2 \tag{6}$$

where $\eta \in (0, \frac{1}{5DG}]$. The main advantage of s_t over ℓ_t is its strong convexity, which allows us to adopt a strongly convex OGD that takes s_t as the objective loss function and attains an $O(\log T)$ regret with respect to s_t. On the other hand, the "upper-bound" property in (5) is preserved in the sense that the regret with respect to the original loss f_t can be upper bounded by:

$$R(T) \leq \frac{\sum_{t=1}^{T} s_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{D}} \sum_{t=1}^{T} s_t(\mathbf{x})}{\eta} + \eta V_T^{s}$$

where $V_T^{s} = \sum_{t=1}^{T} G^2 \|\mathbf{x}_t - \mathbf{x}_*\|^2$. Thus, the employed strongly convex OGD enjoys a novel data-dependant $O((\log T)/\eta + \eta V_T^{s})$ regret with respect to f_t, removing the undesirable factor of d. To optimize this bound to $O(\sqrt{V_T^{s} \log T} + \log T)$, we follow the idea of MetaGrad and run many instances of strongly convex OGD.

Finally, to obtain the optimal $O(\sqrt{T})$ regret bound for general covnex functions, we also introduce a linear surrogate loss function

$$c_t(\mathbf{x}) = -\eta^c(\mathbf{x}_t - \mathbf{x})^\top \mathbf{g}_t + (\eta^c G D)^2 \tag{7}$$

where $\eta^c = \frac{1}{2DG\sqrt{T}}$, which only depends on known quantities. It can be proved that if we run a convex OGD with $c_t(\cdot)$ as the input, its regret with respect to the original loss function $f_t(\cdot)$ can be bounded $O(\sqrt{T})$.

While the idea of incorporating new types of surrogate loss functions to enhance the adaptivity is easy to comprehend, the specific definitions of the two proposed surrogate loss functions in (6) and (7) are more involved. In fact, the proposed functions are carefully designed such that besides the aforementioned properties, they also satisfy that

$$\exp(-s_t(\mathbf{x})) - \exp(-\ell_t(\mathbf{x})) \leq 1 + \eta(\mathbf{x}_t - \mathbf{x})^\top \mathbf{g}_t$$

Algorithm 1 Meta-algorithm

Input: Learning rates $\eta^c, \eta_1, \eta_2, \ldots$, prior weights $\pi_1^c, \pi_1^{1,s}, \pi_1^{2,s}, \ldots$ and $\pi_1^{1,`}, \pi_1^{2,`}, \ldots$
for $t = 1, \ldots, T$ **do**
  Get predictions $\mathbf{x}_t^c$ from Algorithm 2, and $\mathbf{x}_t^{i,`}$, $\mathbf{x}_t^{i,s}$ from Algorithms 3 and 4 for all η
  Play $\mathbf{x}_t = \frac{\sum_i \pi_t^{i,s} \mathbf{x}_t^{i,s} + \sum_i \pi_t^{i,`} \mathbf{x}_t^{i,`}}{\sum_i \pi_t^{i,s} + \sum_i \pi_t^{i,`}}$
  Observe gradient $\mathbf{g}_t$ and send it to all experts
  Update weights:
  $\pi_{t+1}^c = \frac{\pi_t^c e^{-c_t(\mathbf{x}_t^c)}}{\Phi_t}$
  $\pi_{t+1}^{i,s} = \frac{\pi_t^{i,s} e^{-s_t(\mathbf{x}_t^{i,s})}}{\Phi_t}$ for all η
  $\pi_{t+1}^{i,`} = \frac{\pi_t^{i,`} e^{-`_t(\mathbf{x}_t^{i,`})}}{\Phi_t}$ for all η
  where

$$\Phi_t = \sum \left(\pi_t^{i,s} e^{-s_t(\mathbf{x}_t^{i,s})} + \pi_t^{i,`} e^{-`_t(\mathbf{x}_t^{i,`})} \right) + \pi_t^c e^{-c_t(\mathbf{x}_t^c)}$$

end for

Algorithm 2 Convex expert algorithm

$\mathbf{x}_1^c = 0$
for $t = 1, \ldots, T$ **do**
  Send $\mathbf{x}_t^c$ to Algorithm 1
  Receive gradient $\mathbf{g}_t$ from Algorithm 1
  Update $\mathbf{x}_{t+1}^c = \Pi_{\mathcal{D}}^{I_d} \left(\mathbf{x}_t^c - \frac{D}{G\sqrt{t}} \nabla c_t(\mathbf{x}_t^c) \right)$
  where $\nabla c_t(\mathbf{x}_t^c) = \eta^c \mathbf{g}_t$
end for

and

$$\exp(-c_t(\mathbf{x})) \geq 1 + \eta^c (\mathbf{x}_t - \mathbf{x})^\top \mathbf{g}_t$$

which are critical to keep the regret caused by the meta-algorithm under control and will be made clear in Section 4.1.

3.3 THE ALGORITHM

Our method, named **m**ultiple sub-**a**lgorithms and **le**arning **r**ates (Maler), is a two-level hierarchical structure: at the lower level, a set of experts run in parallel, each of which is configured with a different learning algorithm (Algorithm 2, 3, or 4) and learning rate. At the higher level, a meta-algorithm (Algorithm 1) is employed to track the best expert based on empirical performances of the experts.

Meta-Algorithm. Tracking the best expert is a well-studied problem, and our meta-algorithm is built upon the titled exponentially weighted average (van Erven and Koolen, 2016). The inputs of the meta-algorithm are learning rates and prior weights of the experts. In each round t, the meta-algorithm firstly receives actions from all experts (Step 3), and then combines these actions by using exponentially weighted average (Step 4). The weights of the experts are titled by their own η, so that those experts with larger learning rates will be assigned with larger weights. After observing the gradient at $\mathbf{x}_t$ (Step 5), the meta-algorithm updates the weight of each expert via an exponential weighting scheme (Step 6).

Algorithm 3 Exp-concave expert algorithm

Input: Learning rate η
$\mathbf{x}_1^{i,`} = 0$, $\beta = \frac{1}{2}\min\left\{\frac{1}{4G`D}, 1\right\}$, where $G` = \frac{7}{25D}$, $A_1 = \frac{1}{`^2 D^2} I_d$
for $t = 1, \ldots, T$ **do**
  Send $\mathbf{x}_t^{i,`}$ to Algorithm 1
  Receive gradient $\mathbf{g}_t$ from Algorithm 1
  Update

$$A_{t+1} = A_t + \nabla \ell_t \left(\mathbf{x}_t^{i,`}\right) \left(\nabla \ell_t \left(\mathbf{x}_t^{i,`}\right)\right)^\top$$
$$\mathbf{x}_{t+1}^{i,`} = \Pi_{\mathcal{D}}^{A_{t+1}} \left( \mathbf{x}_t^{i,`} - \frac{1}{\beta} A_{t+1}^{-1} \nabla \ell_t \left(\mathbf{x}_t^{i,`}\right) \right)$$

  where $\nabla \ell_t(\mathbf{x}_t^{i,`}) = \eta \mathbf{g}_t + 2\eta^2 \mathbf{g}_t \mathbf{g}_t^\top (\mathbf{x}_t^{i,`} - \mathbf{x}_t)$
end for

Algorithm 4 Strongly convex expert algorithm

Input: Learning rate η
$\mathbf{x}_1^{i,s} = 0$
for $t = 1, \ldots, T$ **do**
  Send $\mathbf{x}_t^{i,s}$ to Algorithm 1
  Receive gradient $\mathbf{g}_t$ from Algorithm 1
  Update

$$\mathbf{x}_{t+1}^{i,s} = \Pi_{\mathcal{D}}^{I_d} \left(\mathbf{x}_t^{i,s} - \frac{1}{2\eta^2 G^2 t} \nabla s_t(\mathbf{x}_t^{i,s}) \right)$$

  where $\nabla s_t(\mathbf{x}_t^{i,s}) = \eta \mathbf{g}_t + 2\eta^2 G^2 (\mathbf{x}_t^{i,s} - \mathbf{x}_t)$
end for

Experts. Experts are themselves *non-adaptive* algorithms, such as OGD and ONS. In each round t, each expert sends its action to the meta-algorithm, then receives a gradient vector from the meta-algorithm, and finally updates the action based on the received vector. To optimally handle general convex, exp-concave, and strongly convex functions simultaneously, we design three types of experts as follows:

Convex expert. As discussed in Section 3.2, there is no need to search for the optimal learning rate in convex cases and thus we only run one convex OGD

(Algorithm 2) on the convex surrogate loss function $c_t(\mathbf{x})$ in (7). We denote its action in round t as $\mathbf{x}_t^c$. Its prior weight π_1^c and learning rate η^c are set to be $1/3$ and $1/(2GD\sqrt{T})$, respectively.

Exp-concave experts. We keep $\lceil \frac{1}{2}\log T \rceil + 1$ exp-concave experts, each of which is a standard ONS (Algorithm 3) running on an exp-concave surrogate loss function $\ell_t(\cdot)$ in (3) with a different η. We denote its output in round t as $\mathbf{x}_t^{i,\backprime}$. For expert $i = 0, 1, 2, ..., \lceil \frac{1}{2}\log T \rceil$, its learning rate and prior weight are assigned as follows:

$$\eta_i = \frac{2^{-i}}{5DG}, \text{ and } \pi_1^{i,\backprime} = \frac{C}{3(i+1)(i+2)}$$

where $C = 1 + 1/\left(1 + \lceil \frac{1}{2}\log T \rceil\right)$ is a normalization parameter.

Strongly convex experts. We maintain $\lceil \frac{1}{2}\log T \rceil + 1$ strongly convex experts. In each round t, every expert takes a strongly convex surrogate loss $s_t(\cdot)$ in (6) (with different η) as the loss function, and adopts strongly convex OGD (Algorithm 4) to update its action, denoted as $\mathbf{x}_t^{i,S}$. For $i = 0, 1, 2, ..., \lceil \frac{1}{2}\log T \rceil$, we configure the i-th strongly convex expert as follows:

$$\eta_i = \frac{2^{-i}}{5DG}, \text{ and } \pi_1^{i,S} = \frac{C}{3(i+1)(i+2)}.$$

Computational Complexity. The computational complexity of Maler is dominated by its experts. If we ignore the projection procedure, the run time of Algorithms 2, 3 and 4 are $O(d)$, $O(d^2)$ and $O(d)$ per iteration respectively. Combining with the number of experts, the total run time of Maler is $O(d^2 \log T)$, which is of the same order as that of MetaGrad. When taking the projection into account, we note that it can be computed efficiently for many convex bodies used in practical applications such as d-dimensional balls, cubes and simplexes (Hazan et al., 2007). To put it more concrete, when the convex body is a d-dimensional ball, projections in Algorithms 2, 3, and 4 require $O(d)$, $O(d^3)$, and $O(d)$ time respectively, and consequently the total computational complexity of Maler is $O(d^3 \log T)$, which is also the same as that of MetaGrad.

3.4 THEORETICAL GUARANTEES

Theorem 1. *Suppose Assumptions 1 and 2 hold. Let* $V_T^S = G^2 \sum_{t=1}^T \|\mathbf{x}_t - \mathbf{x}_*\|^2$, *and* $V_T^{\backprime} = \sum_{t=1}^T ((\mathbf{x}_t - \mathbf{x}_*)^\top \mathbf{g}_t)^2$. *Then the regret of Maler is simultaneously bounded by*

$$R(T) \leq \left(2\ln 3 + \frac{3}{2}\right) GD\sqrt{T} = O\left(\sqrt{T}\right) \quad (8)$$

$$\begin{aligned} R(T) &\leq 3\sqrt{V_T^{\backprime}\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 10d\log T\right)} \\ &\quad + 10GD\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 10d\log T\right) \\ &= O\left(\sqrt{V_T^{\backprime} d\log T} + d\log T\right) \end{aligned} \quad (9)$$

and

$$\begin{aligned} R(T) &\leq 3\sqrt{V_T^S\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 1 + \log T\right)} \\ &\quad + 10GD\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 1 + \log T\right) \\ &= O\left(\sqrt{V_T^S \log T} + \log T\right). \end{aligned} \quad (10)$$

Remark. Theorem 1 implies that, similar to MetaGrad, Maler can be upper bounded by $O(\sqrt{V_T^{\backprime} d\log T} + d\log T)$. Hence, the conclusions of MetaGrad under some fast rates examples such as Bernstein condition (van Erven et al., 2015) still hold for Maler. Moreover, it shows that Maler also enjoys a new type of data-dependant bound $O(\sqrt{V_T^S \log T} + \log T)$, and thus may perform better than MetaGrad in some high dimensional cases such that $V_T^S \quad dV_T^{\backprime}$.

Next, based on Theorem 1, we derive the following regret bounds for strongly convex and exp-concave loss functions, respectively.

Corollary 2. *Suppose Assumptions 1 and 2 hold. For* λ*-strongly convex functions, the regret of Maler is upper bounded by*

$$\begin{aligned} R(T) &\leq \left(10GD + \frac{9G^2}{2\lambda}\right)\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right)\right. \\ &\quad \left. + 1 + \log T\right) = O\left(\frac{1}{\lambda}\log T\right). \end{aligned}$$

For α*-exp-concave functions, let* $\beta = \frac{1}{2}\min\left\{\alpha, \frac{1}{4GD}\right\}$, *and Maler enjoys the following regret bound*

$$\begin{aligned} R(T) &\leq \left(10GD + \frac{9}{2\beta}\right)\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right)\right. \\ &\quad \left. + 10d\log T\right) = O\left(\frac{1}{\alpha}d\log T\right). \end{aligned}$$

Remark. Theorem 1 and Corollary 2 indicate that our proposed algorithm achieves the minimax optimal $O(\sqrt{T})$, $O(d \log T)$ and $O(\log T)$ regret bounds for convex, exponentially concave and strongly convex functions respectively. In contrast, the regret bounds of MetaGrad for the three types of loss functions are $O(\sqrt{T \log\log T})$, $O(d \log T)$ and $O(d \log T)$ respectively, which are sub-optimal for convex and strongly convex functions.

4 REGRET ANALYSIS

The regret of Maler can be generally decomposed into two components, i.e., the regret of the meta-algorithm (meta regret) and the regrets of expert algorithms (expert regret). We firstly upper bound the two parts separately, and then analyse their composition to prove Theorem 1.

4.1 META REGRET

We define meta regret as the difference between the cumulative surrogate losses of the actions of the meta-algorithm (i.e., $\mathbf{x}_t$s) and that of the actions from a specific expert, which measures the learning ability of the meta-algorithm. For meta regret, we introduce the following lemma.

Lemma 1. *For every grid point η, we have*

$$\sum_{t=1}^{T} s_t(\mathbf{x}_t) - \sum_{t=1}^{T} s_t(\mathbf{x}_t^{i,s}) \leq 2 \ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) \tag{11}$$

$$\sum_{t=1}^{T} \ell_t(\mathbf{x}_t) - \sum_{t=1}^{T} \ell_t(\mathbf{x}_t^{i,\ell}) \leq 2 \ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) \tag{12}$$

and

$$\sum_{t=1}^{T} c_t(\mathbf{x}_t) - \sum_{t=1}^{T} c_t(\mathbf{x}_t^c) \leq \ln 3. \tag{13}$$

Proof. We firstly introduce three inequalities. For every grid point η,

$$\begin{aligned} e^{-s_t(\mathbf{x}_t^{i,s})} &\overset{(6)}{\leq} e^{(\mathbf{x}_t - \mathbf{x}_t^{i,s})^\top \mathbf{g}_t - \frac{1}{2}G^2\|\mathbf{x}_t - \mathbf{x}_t^{i,s}\|^2} \\ &\leq e^{(\mathbf{x}_t - \mathbf{x}_t^{i,s})^\top \mathbf{g}_t - \left((\mathbf{x}_t - \mathbf{x}_t^{i,s})^\top \mathbf{g}_t\right)^2} \\ &\leq 1 + \eta(\mathbf{x}_t - \mathbf{x}_t^{i,s})^\top \mathbf{g}_t \end{aligned} \tag{14}$$

where the first inequality follows from Cauchy-Schwarz inequality, and the second inequality is due to $e^{x - x^2} \leq 1 + x$ for any $x \leq \frac{2}{3}$ (van Erven and Koolen, 2016). Applying similar arguments, we have

$$\begin{aligned} e^{-\ell_t(\mathbf{x}_t^{i,\ell})} &\overset{(3)}{\leq} e^{(\mathbf{x}_t - \mathbf{x}_t^{i,\ell})^\top \mathbf{g}_t - \left((\mathbf{x}_t - \mathbf{x}_t^{i,\ell})^\top \mathbf{g}_t\right)^2} \\ &\leq 1 + \eta(\mathbf{x}_t - \mathbf{x}_t^{i,\ell})^\top \mathbf{g}_t \end{aligned} \tag{15}$$

and

$$\begin{aligned} e^{-c_t(\mathbf{x}_t^c)} &\overset{(7)}{\leq} e^{\eta^c(\mathbf{x}_t - \mathbf{x}_t^c)^\top \mathbf{g}_t - (\eta^c GD)^2} \\ &\leq e^{\eta^c(\mathbf{x}_t - \mathbf{x}_t^c)^\top \mathbf{g}_t - \left(\eta^c(\mathbf{x}_t - \mathbf{x}_t^c)^\top \mathbf{g}_t\right)^2} \\ &\leq 1 + \eta^c(\mathbf{x}_t - \mathbf{x}_t^c)^\top \mathbf{g}_t. \end{aligned} \tag{16}$$

Note that by definition of η^c we have $\eta^c(\mathbf{x}_t - \mathbf{x}_t^c)^\top \mathbf{g}_t > \frac{1}{2}$.

Now we are ready to prove Lemma 1. Define potential function

$$\Phi_T = \sum \left(\pi_1^{i,s} e^{-\sum_{t=1}^{T} s_t(\mathbf{x}_t^{i,s})} + \pi_1^{i,\ell} e^{-\sum_{t=1}^{T} \ell_t(\mathbf{x}_t^{i,\ell})}\right) + \pi_1^c e^{-\sum_{t=1}^{T} c_t(\mathbf{x}_t^c)}. \tag{17}$$

We have

$$\begin{aligned} &\Phi_{T+1} - \Phi_T \\ &= \sum \pi_1^{i,s} e^{-\sum_{t=1}^{T} s_t(\mathbf{x}_t^{i,s})} \left(e^{-s_{T+1}(\mathbf{x}_{T+1}^{i,s})} - 1\right) \\ &\quad + \sum \pi_1^{i,\ell} e^{-\sum_{t=1}^{T} \ell_t(\mathbf{x}_t^{i,\ell})} \left(e^{-\ell_{T+1}(\mathbf{x}_{T+1}^{i,\ell})} - 1\right) \\ &\quad + \pi_1^c e^{-\sum_{t=1}^{T} c_t(\mathbf{x}_t^c)} \left(e^{-c_t(\mathbf{x}_{T+1}^c)} - 1\right) \\ &\leq \sum \pi_1^{i,s} e^{-\sum_{t=1}^{T} s_t(\mathbf{x}_t^{i,s})} \eta(\mathbf{x}_{T+1} - \mathbf{x}_{T+1}^{i,s})^\top \mathbf{g}_t \\ &\quad + \sum \pi_1^{i,\ell} e^{-\sum_{t=1}^{T} \ell_t(\mathbf{x}_t^{i,\ell})} \eta(\mathbf{x}_{T+1} - \mathbf{x}_{T+1}^{i,\ell})^\top \mathbf{g}_t \\ &\quad + \pi_1^c e^{-\sum_{t=1}^{T} c_t(\mathbf{x}_t^c)} \eta^c(\mathbf{x}_{T+1} - \mathbf{x}_{T+1}^c)^\top \mathbf{g}_t \\ &= (a_T \mathbf{x}_{T+1} - \mathbf{b}_T)^\top \mathbf{g}_t \end{aligned} \tag{18}$$

where the inequality is due to (14), (15), and (16),

$$\begin{aligned} a_T &= \sum \pi_1^{i,s} e^{-\sum_{t=1}^{T} s_t(\mathbf{x}_t^{i,s})} \eta + \pi_1^c e^{-\sum_{t=1}^{T} c_t(\mathbf{x}_t^c)} \eta^c \\ &\quad + \sum \pi_1^{i,\ell} e^{-\sum_{t=1}^{T} \ell_t(\mathbf{x}_t^{i,\ell})} \eta \\ \mathbf{b}_T &= \sum \pi_1^{i,\ell} e^{-\sum_{t=1}^{T} \ell_t(\mathbf{x}_t^{i,\ell})} \eta \mathbf{x}_{T+1}^{i,\ell} \\ &\quad + \pi_1^c e^{-\sum_{t=1}^{T} c_t(\mathbf{x}_t^c)} \eta^c \mathbf{x}_{T+1}^c \\ &\quad + \sum \pi_1^{i,s} e^{-\sum_{t=1}^{T} s_t(\mathbf{x}_t^{i,s})} \eta \mathbf{x}_{T+1}^{i,s} \end{aligned}$$

On the other hand, by the update rule of $\mathbf{x}_t$, we have

$$\begin{aligned} \mathbf{x}_{T+1} &= \frac{\sum (\pi_{T+1}^{i,s} \eta \mathbf{x}_{T+1}^{i,s} + \pi_{T+1}^{i,\ell} \eta \mathbf{x}_{T+1}^{i,\ell}) + \pi_{T+1}^c \eta^c \mathbf{x}_{T+1}^c}{\sum (\pi_{T+1}^{i,s} \eta + \pi_{T+1}^{i,\ell} \eta) + \pi_{T+1}^c \eta^c} \\ &= \frac{\mathbf{b}_T}{a_T} \end{aligned} \tag{19}$$

where the second equality comes from Step 6 of Algorithm 1, and note that π^c_{t+1}, π_{t+1} and π^s_{t+1} share the same denominator. Plugging (19) into (18), we get

$$T+1 \quad T \quad 0$$

which implies that

$$1 = \quad 0 \quad 1 \quad T. \tag{20}$$

Note that all terms in the the definition of T (17) are positive. Combining with (20), it indicates that these terms are less than 1. Thus,

$$0 \quad \ln\left(\pi_1 e^{-\sum_{t=1}^{T} s_t(x_t)}\right) = \sum_{t=1}^{T} s_t(x_t) + \ln\frac{1}{\pi_1}$$

$$0 \quad \ln\left(\pi_1 e^{-\sum_{t=1}^{T} \ell_t(x_t)}\right) = \sum_{t=1}^{T} \ell_t(x_t) + \ln\frac{1}{\pi_1}$$

and

$$0 \quad \ln\left(\pi^c_1 e^{-\sum_{t=1}^{T} c_t(x^c_t)}\right) = \sum_{t=1}^{T} c_t(x^c_t) + \ln\frac{1}{\pi^c_1}.$$

We finish the proof by noticing that for every grid point η,

$$\ln\frac{1}{\pi_1} \quad \ln\left(3\left(\left\lceil\frac{1}{2}\log T\right\rceil + 1\right)\left(\left\lceil\frac{1}{2}\log T\right\rceil + 2\right)\right) \quad 2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right)$$

$$\ln\frac{1}{\pi_1} \quad \ln\left(3\left(\left\lceil\frac{1}{2}\log T\right\rceil + 1\right)\left(\left\lceil\frac{1}{2}\log T\right\rceil + 2\right)\right) \quad 2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right)$$

and $\ln\frac{1}{\pi^c_1} = \ln 3$. □

4.2 EXPERT REGRET

For the regret of each expert, we have the following lemma. The proof is postponed to the appendix.

Lemma 2. *For every grid point η and any* u *2 D, we have*

$$\sum_{t=1}^{T} s_t(x_t) \quad \sum_{t=1}^{T} s_t(u) \quad 1 + \log T \tag{21}$$

$$\sum_{t=1}^{T} \ell_t(x_t) \quad \sum_{t=1}^{T} \ell_t(u) \quad 10d\log T \tag{22}$$

and

$$\sum_{t=1}^{T} c_t(x^c_t) \quad \sum_{t=1}^{T} c_t(u) \quad \frac{3}{4}. \tag{23}$$

4.3 PROOF OF THEOREM 1

In the following, we combine the regret analysis of the meta and expert algorithms to prove Theorem 1.

Proof. To get the $O(\sqrt{T})$ bound of (8), we upper bound the regret by using the properties of c_t as follows.

$$\begin{aligned}
&R(T)\\
&\overset{(1)}{=} \sum_{t=1}^{T} f_t(x_t) \quad \sum_{t=1}^{T} f_t(x_*)\\
&\overset{(2)}{\le} \sum_{t=1}^{T} g_t^\top(x_t \quad x_*)\\
&\overset{(7)}{\le} \frac{\sum_{t=1}^{T} \quad c_t(\mathbf{x}_*) + \sum_{t=1}^{T}(\eta^c GD)^2}{\eta^c}\\
&= \frac{\sum_{t=1}^{T}(c_t(x_t) \quad c_t(x^c_t)) + \sum_{t=1}^{T}(c_t(x^c_t) \quad c_t(\mathbf{x}_*))}{\eta^c}\\
&\quad \left(\ln 3 + \frac{3}{4}\right) 2GD\sqrt{T}
\end{aligned}$$

where the last inequality follows from (13) and (23).

Next, to achieve the regret of (10), we upper bound $R(T)$ by making use of the properties of s_t. For every grid point η, we have

$$\begin{aligned}
&R(T)\\
&\overset{(1)}{=} \sum_{t=1}^{T} f_t(x_t) \quad \sum_{t=1}^{T} f_t(x_*)\\
&\overset{(2)}{\le} \sum_{t=1}^{T} g_t^\top(x_t \quad x_*)\\
&\overset{(6)}{\le} \frac{\sum_{t=1}^{T} \quad s_t(\mathbf{x}_*) + \eta^2 G^2 k x_* \quad x_t k^2}{\eta}\\
&= \frac{\sum_{t=1}^{T}(s_t(x_t) \quad s_t(x_t)) + \sum_{t=1}^{T}(s_t(x_t) \quad s_t(\mathbf{x}_*))}{\eta}\\
&\quad + \sum_{t=1}^{T} \eta G^2 k x_t \quad x_* k^2\\
&\quad \frac{2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 1 + \log T}{\eta}\\
&\quad + \sum_{t=1}^{T} \eta G^2 k x_t \quad x_* k^2\\
&= \eta V_T^s + \frac{2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 1 + \log T}{\eta}
\end{aligned} \tag{24}$$

where $V_T^s = \sum_{t=1}^{T} G^2 k x_t \quad x_* k^2$, and the inequality

comes from (11) and (21). Define

$$A = 2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 1 + \log T \quad 1.$$

The optimal η to minimize the right hand side of (24) is

$$\eta = \sqrt{\frac{A}{V_T^s}} \quad \frac{1}{5GD}\sqrt{\frac{1}{T}}. \tag{25}$$

If $\eta \quad \frac{1}{5GD}$, then by construction there exists a grid point η such that $\eta \in [\frac{1}{2}, \eta]$, and thus

$$R(T) \quad \eta V_T^s + \frac{A}{\eta} \quad 2\eta V_T^s + \frac{A}{\eta} = 3\sqrt{V_T^s A}.$$

On the other hand, if $\eta > \frac{1}{5GD}$, then by (25) we get

$$V_T^s \quad 25G^2D^2A.$$

Thus for $\eta_1 = \frac{1}{5GD}$, we have

$$R(T) \quad 10GDA.$$

Overall, we obtain

$$R(T) \quad 3\sqrt{V_T^s A} + 10GDA.$$

Finally, we upper bound the regret by using the properties of the exp-concave surrogate loss functions. For every grid point η, we have

$$\begin{aligned}
&R(T)\\
&\overset{(2)}{=} \sum_{t=1}^{T} \mathsf{g}_t^\top(\mathsf{x}_t \quad \mathsf{x}_*)\\
&\overset{(3)}{\leq} \frac{\sum_{t=1}^{T} \quad \ell_t(\mathbf{x}_*) + \eta^2\left(\mathsf{g}_t^\top(\mathsf{x}_t \quad \mathsf{x}_*)\right)^2}{\eta}\\
&= \frac{\sum_{t=1}^{T}\left(\ell_t(\mathsf{x}_t) \quad \ell_t(\mathsf{x}_t')\right)}{\eta} + \eta\sum_{t=1}^{T}\left(\mathsf{g}_t^\top(\mathsf{x} \quad \mathsf{x}_*)\right)^2\\
&\quad + \frac{\sum_{t=1}^{T}\left(\ell_t(\mathsf{x}_t') \quad \ell_t(\mathbf{x}_*)\right)}{\eta}\\
&\quad \frac{2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 10d\log T}{\eta}\\
&\quad + \eta\sum_{t=1}^{T}\left(\mathsf{g}_t^\top(\mathsf{x}_* \quad \mathsf{x}_t)\right)^2\\
&= \eta V_T' + \frac{2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 10d\log T}{\eta}
\end{aligned}$$

where the last inequality comes from (12) and (22). Define

$$B = 2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 10d\log T.$$

By similar arguments, we get

$$R(T) \quad 3\sqrt{V_T' B} + 10GDB.$$

□

4.4 PROOF OF COROLLARY 2

Proof. For α-exp-concave functions, we have

$$\begin{aligned}
&R(T)\\
&\sum_{t=1}^{T}\mathsf{g}_t^\top(\mathsf{x}_t \quad \mathsf{x}_*) \quad \frac{\beta}{2}V_T'\\
&3\sqrt{V_T'\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 10d\log T\right)}\\
&+ 10GD\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 10d\log T\right)\\
&\quad \frac{\beta}{2}V_T'\\
&\frac{3\gamma}{2}V_T' + \left(10GD + \frac{3}{2\gamma}\right)\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right)\right.\\
&\left. + 10d\log T\right) \quad \frac{\beta}{2}V_T'
\end{aligned}$$

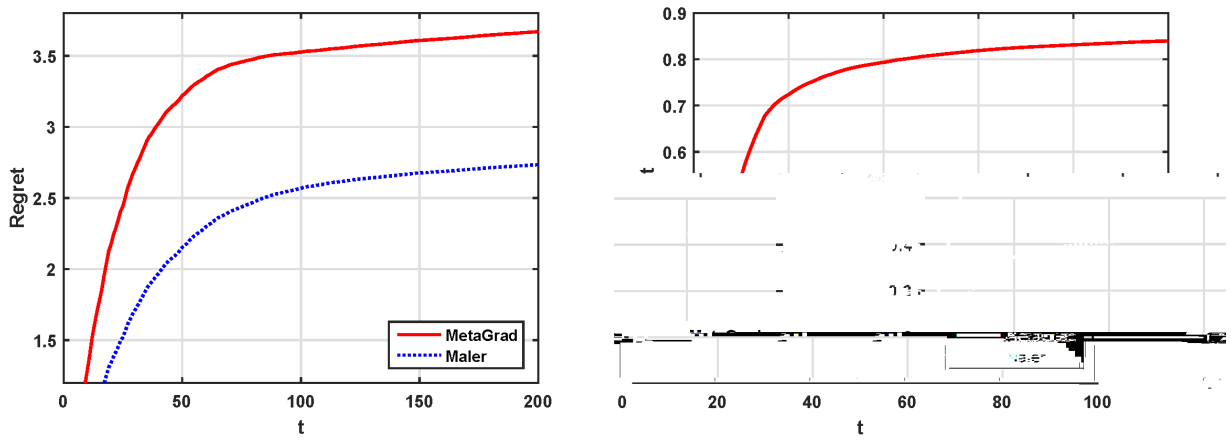
where the last inequality is based on $\sqrt{xy} \quad \frac{\gamma}{2}x + \frac{y}{2\gamma}$ for all $x, y, \gamma > 0$, The result follows from $\gamma = \frac{1}{3}$.
For λ-strongly convex functions, we have

$$\begin{aligned}
&R(T)\\
&\sum_{t=1}^{T}\mathsf{g}_t^\top(\mathsf{x}_t \quad \mathsf{x}_*) \quad \frac{\lambda}{2}\|\mathsf{x}_t \quad \mathsf{x}_*\|^2\\
&3\sqrt{V_T^s\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 1 + \log T\right)}\\
&+ 10GD\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right) + 1 + \log T\right)\\
&\quad \frac{\lambda}{2G^2}V_T^s\\
&\frac{3\gamma V_T^s}{2} + \left(10GD + \frac{3}{2\gamma}\right)\left(2\ln\left(\sqrt{3}\left(\frac{1}{2}\log_2 T + 3\right)\right)\right.\\
&\left. + 1 + \log T\right) \quad \frac{\lambda}{2}V_T^s
\end{aligned}$$

where the last inequality is based on $\sqrt{xy} \quad \frac{\gamma}{2}x + \frac{y}{2\gamma}$ for all $x, y, \gamma > 0$, and the results follows from $\gamma = \frac{1}{3G^2}$. □

5 EXPERIMENTS

In this section, we present empirical results on different online learning tasks to evaluate the proposed algorithm. We choose MetaGrad as the baseline algorithm.



(a) Online regression

(b) Online classification

Fig. 1: Emprecial results of Maler and MetaGrad for online regression and classification

5.1 ONLINE REGRESSION

We consider mini-batch least mean square regression with ℓ_2-regularizer, which is a classic problem belonging to online strongly convex optimization. In each round t, a small batch of training examples $f(\mathbf{x}_{t;1}, y_{t;1}), \ldots, (\mathbf{x}_{t;n}, y_{t;n})g$ arrives, and at the same time, the learner makes a prediction of the unknown parameter $\mathbf{w}_*$, denoted as $\mathbf{w}_t$, and suffers a loss, defined as

$$f_t(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{w}^\top \mathbf{x}_{t;i} \quad y_{t;i}\right)^2 + \lambda k\mathbf{w}k^2. \qquad (26)$$

We conduct the experiment on a symmetric data set, which is constructed as follows. We sample $\mathbf{w}_*$ and feature vectors $\mathbf{x}_{t;i}$ uniformly at random from the d-ball of diameter 1 and 10 respectively, and generate $y_{t;i}$ according to a linear model: $y_{t;i} = \mathbf{w}_*^\top \mathbf{x}_{t;i} + \eta_t$, where the noise is drawn from a normal distribution. We set batch size $n = 200$, $\lambda = 0.001$, $d = 50$, and $T = 200$. The regret v.s. time horizon is shown in Fig. 1(a). It can be seen that Maler achieves faster convergence rate than MetaGrad.

5.2 ONLINE CLASSIFICATION

Next, we consider online classification by using logistic regression. In each round t, we receive a batch of training examples $f(\mathbf{x}_{t;1}, y_{t;1}), \ldots, (\mathbf{x}_{t;n}, y_{t;n})g$, and choose a linear classifier $\mathbf{w}_t$. After that, we suffer a logistic loss

$$f_t(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\log(1 + \exp(\quad y_{t;i}\mathbf{w}_t^\top \mathbf{x}_{t;i})) \qquad (27)$$

which is exp-concave. We conduct the experiments on a classic real-world data set a9a (Chang and Lin, 2011). We scale all feature vectors to the unit ball, and restrict the decision set D to be a ball of radius 0.5 and centered at the origin, so that Assumptions 1 and 2 are satisfied. We set batch size $n = 200$, and $T = 100$. The regret v.s. time horizon is shown in Figure 1(b). It can be seen that Maler performs better than MetaGrad. Although the worst-case regret bounds of Maler and MetaGrad for exp-concave loss are on the same order, the experimental results are not surprising since Maler enjoys a tighter data-dependant regret bound than that of MetaGrad.

6 CONCLUSION AND FURORE WORK

In this paper, we propose a universal algorithm for online convex optimization, which achieves the optimal $O(\sqrt{T})$, $O(d \log T)$ and $O(\log T)$ regret bounds for general convex, exp-concave and strongly convex functions respectively, and enjoys a new type of data-dependent bound. The main idea is to consider different types of learning algorithms and learning rates at the same time. Experiments on online regression and online classification problems demonstrate the effectiveness of our method. In the future, we will investigate whether our proposed algorithm can extend to achieve border adaptivity in various directions, for example, adapting to changing environments (Hazan and Seshadhri, 2007; Jun et al., 2017) and/or adapting to data structures (Reddi et al., 2018; Wang et al., 2019).

Acknowledgement

This work was partially supported by NSFC-NRF Joint Research Project (61861146001), YESS (2017QNRC001), and Zhejiang Provincial Key Laboratory of Service Robot.

References

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

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

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

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

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

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

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

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

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

Hazan, E., Rakhlin, A., and Bartlett, P. L. (2008). Adaptive online gradient descent. In *Advances in Neural Information Processing Systems 21*, pages 65–72.

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

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

Reddi, S. J., Kale, S., and Kumar, S. (2018). On the convergence of adam and beyond. In *Proceedings of 6th International Conference on Learning Representations*.

Shalev-Shwartz, S. et al. (2012). Online learning and online convex optimization. *Foundations and Trends R in Machine Learning*, 4(2):107–194.

Tieleman, T. and Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, pages 26–31.

van Erven, T., Grünwald, P. D., Mehta, N. A., Reid, M. D., and Williamson, R. C. (2015). Fast rates in statistical and online learning. *Journal of Machine Learning Research*, 16:1793–1861.

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

Wang, G., Lu, S., Tu, W., and Zhang, L. (2019). Sadam: A variant of adam for strongly convex functions. *ArXiv preprint arXiv:1905.02957*.

Wang, G., Zhao, D., and Zhang, L. (2018). Minimizing adaptive regret with one gradient per iteration. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 2762–2768.

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

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

Zhang, L., Yang, T., Jin, R., and Zhou, Z.-H. (2018b). Dynamic regret of strongly adaptive methods. In *Proceedings of the 35th International Conference on Machine Learning*, pages 5877–5886.

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

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