Discounted Adaptive Online Learning: Towards Better Regularization

Zhiyu Zhang [1] **David Bombara** [1] **Heng Yang** [1]

Abstract

We study online learning in adversarial nonstationary environments. Since the future can be very different from the past, a critical challenge is to gracefully forget the history while new data comes in. To formalize this intuition, we revisit the discounted regret in online convex optimization, and propose an adaptive (i.e., instance optimal), FTRL-based algorithm that improves the widespread non-adaptive baseline – gradient descent with a constant learning rate. From a practical perspective, this refines the classical idea of regularization in lifelong learning: we show that designing better regularizers can be guided by the principled theory of adaptive online optimization. Complementing this result, we also consider the (Gibbs & Candes, 2021)-style online conformal prediction problem, where the goal is to sequentially predict the uncertainty sets of a black-box machine learning model. We show that the FTRL nature of our algorithm can simplify the conventional gradient-descent-based analysis, leading to instance-dependent performance guarantees.

1. Introduction

Online learning can be broadly defined as a sequential decision making problem, where each decision leverages the *learned* knowledge from previous observations. However, while *forgetting* is often thought as the opposite of *learning*, the two concepts are actually coherent due to the *distribution shifts* in practice. Think about deploying a drone in the wild: a common subtask is to learn its time-varying dynamics model on the fly, but when doing that, we have to exclude the obsolete data that possibly contradicts the current environment. This leads to a natural challenge for the resulting online learning problem: how do we handle the possible shortage of data after forgetting?

One potential solution is to inject suitable *inductive bias* into the algorithm, which is among the most important ideas in machine learning. Specifically, the inductive bias refers to our prior belief of the ground truth, before observing the data of interest. Regarding the drone example, physics provides general principles on the evolution of the nature, while modern foundation models can encode diverse "world knowledge" from large scale pre-training. The point is that even though forgetting reduces the amount of online data, one could still exploit such inductive bias to improve the learning performance.

Despite this natural intuition, algorithmically achieving it remains a nontrivial task. In particular, the associated online learning algorithm has two considerations to trade off (i.e., the inductive bias and the online data), and optimally balancing them requires going beyond simple heuristics. The present work studies this problem from a theoretical perspective, based on a *discounted* variant of *Online Convex Optimization* (OCO; Zinkevich 2003; Cesa-Bianchi & Lugosi 2006). Our results emphasize the importance of *regularization* in nonstationary online learning:

> On an algorithm that gradually forgets the online data, one could use regularizers to inject the inductive bias, *which the algorithm never forgets*.

Furthermore, designing better optimizers can be guided by the theory of adaptive OCO.

1.1. Contribution

This paper presents new results on two related topics: (Section 2) nonstationary online learning, and (Section 3) *Online Conformal Prediction* (OCP; Gibbs & Candes, 2021).

- First, we consider the discounted OCO problem, formally introduced in Section 2. It is known that under standard assumptions on the complexity of the problem, the *minimax optimal* discounted regret bound can be achieved by an extremely simple and widespread baseline – *Online Gradient Descent* with constant[1] learning rate (denoted

[1]Non-annealing and horizon-independent.

Future versions available at https://arxiv.org/abs/2402.02720. Link to the code: https://github.com/ComputationalRobotics/discounted-adaptive.

[1]Harvard University. Correspondence to: Zhiyu Zhang <zhiyuz@seas.harvard.edu>, David Bombara <davidbombara@g.harvard.edu>, Heng Yang <hankyang@seas.harvard.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

as constant-LR OGD). In this paper, we propose an *adaptive* algorithm based on *Follow the Regularized Leader* (FTRL; Abernethy et al., 2008), such that

- the minimax optimality of the OGD baseline is improved to *instance optimality*; and
- all assumptions beyond convexity are removed.

More concretely, this is achieved by combining an undiscounted adaptive OCO algorithm (Zhang et al., 2024) with a simple *rescaling trick* – the latter can convert *scale-free* (Orabona & Pál, 2018) upper bounds on the standard undiscounted regret to the discounted regret, which could be of independent interest.

In practice, compared to existing nonstationary OCO algorithms that minimize the *dynamic regret* or the *strongly adaptive regret*, our algorithm eschews the standard bi-level aggregation procedure (Hazan & Seshadhri, 2009; Daniely et al., 2015; Zhang et al., 2018a), thus is computationally more efficient. Compared to the "default" approach in continual / lifelong learning based on constant-LR OGD (Abel et al., 2023), our algorithm uses a non-trivial data-dependent regularizer to adaptively exploit the available inductive bias. Furthermore, the design of this regularizer follows from the principled theory of adaptive OCO, rather than heuristics.

- Next, we consider OCP, a downstream online learning task with set-membership predictions. To combat the distribution shifts commonly found in practice, recent works (Gibbs & Candes, 2021; Gibbs & Candès, 2022; Bhatnagar et al., 2023) applied nonstationary OCO algorithms (such as constant-LR OGD) to this setting. The twist is that besides appropriate regret bounds, one has to establish *coverage* guarantees as well.

  In this setting, our discounted OCO algorithm leads to strong instance-dependent guarantees, e.g., the adaptivity to the *targeted coverage level*. Notably, since our algorithm is built on the FTRL framework rather than gradient descent, the associated coverage guarantee follows directly from the *stability* of its iterates. This exemplifies the analytical strength of FTRL in OCP (over gradient descent), which, to our knowledge, has not been demonstrated in the literature.

Finally, we complement the above theoretical results with OCP experiments (Section 4).

1.2. Related work

This paper explores the connection between two separate topics in online learning: discounting and adaptivity.

Discounting Motivated by the intuition that "the recent history is more important than the distant past", the discounted regret has been studied by a series of works on non-stationary online learning (Cesa-Bianchi & Lugosi, 2006; Freund & Hsu, 2008; Chernov & Zhdanov, 2010; Kapralov & Panigrahy, 2011; Cesa-Bianchi et al., 2012; Brown & Sandholm, 2019). Most of them do not consider adaptivity, although the under-appreciated (Kapralov & Panigrahy, 2011) presents important early ideas. Recently, the discounted regret seems to lose its theoretical popularity to the dynamic regret and the strongly adaptive regret, which we survey in Appendix A. However, due to its computational efficiency, the idea of discounting is still prevalent in practice, as exemplified by the success of the ADAM optimizer (Kingma & Ba, 2014).

Concurrent to this work, Ahn et al. (2024) presented a conversion from the discounted regret to the dynamic regret. Independently, Jacobsen & Cutkosky (2024) studied the dynamic and strongly adaptive regret of discounted algorithms, in the context of *online linear regression*.

Adaptivity Focusing on stationary environments, adaptive OCO concerns going beyond the conventional worst case guarantees (Orabona, 2023a, Chapter 4 and 9). More than a decade of research effort culminates in a series of OCO algorithms that do not rely on any extra assumption beyond convexity (Cutkosky, 2019; Mhammedi & Koolen, 2020; Chen et al., 2021; Jacobsen & Cutkosky, 2022; Zhang et al., 2024; Cutkosky & Mhammedi, 2024), which this paper builds on. Different from non-adaptive algorithms like OGD, such adaptivity crucially relies on various sophisticated forms of regularization (McMahan & Orabona, 2014; Orabona & Pál, 2016; Cutkosky & Orabona, 2018; Zhang et al., 2022). This naturally resonates with the crucial role of regularization in "forgettive" continual / lifelong learning settings (De Lange et al., 2021), but to our knowledge, a quantitative connection has not been established in the literature. By studying adaptivity on the discounted regret, we aim to fill this gap.

Regarding the OCP problem (the second half of this paper), related works are discussed in Section 3.

1.3. Notation

Throughout this paper, $\|\cdot\|$ denotes the Euclidean norm. $\Pi_{\mathcal{X}}(x)$ is the Euclidean projection of x onto a closed convex set $\mathcal{X}$. The diameter of a set $\mathcal{X}$ is $\sup_{x,y\in\mathcal{X}} \|x-y\|$. For two integers $a \le b$, $[a:b]$ is the set of all integers c such that $a \le c \le b$. The brackets are removed when on the subscript, denoting a tuple with indices in $[a:b]$. If $a > b$, then the product $\prod_{i=a}^{b} \lambda_i := 1$. $\log$ means natural logarithm. 0 is a zero vector whose dimension depends on the context.

We define the *imaginary error function* as $\operatorname{erfi}(x) := \int_0^x \exp(u^2)du$; this is scaled by $\sqrt{\pi}/2$ from the conventional definition, thus can also be queried from standard software packages like SCIPY and JAX.

2. Discounted Adaptivity

Setting The first half of this paper is about discounted *Online Convex Optimization* (OCO), a two-person repeated game between a player we control and an adversarial environment. Different from the standard OCO problem (Zinkevich, 2003), there is also an *expert* that sequentially selects the discount factors for the player. Specifically, in each round we consider the following interaction protocol.

1. We (the player) make a prediction $x_t \in \mathcal{X}$ using past observations, where $\mathcal{X} \subset \mathbb{R}^d$ is closed and convex.
2. The environment picks a convex loss function $l_t : \mathcal{X} \to \mathbb{R}$, and reveal a subgradient $g_t \in \partial l_t(x_t)$ to the player.
3. The expert picks a discount factor $\lambda_{t-1} \in (0, \infty)$,[2] and reveal it to the player.
4. The environment can choose to terminate the game. If so, let T be the total number of rounds.

At the end of the game, the environment can also choose any fixed prediction $u \in \mathcal{X}$, called a *comparator*. Without knowing the environment and the expert beforehand, our (the player's) goal is to guarantee low *discounted regret*,

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) := \sum_{t=1}^{T} \left(\prod_{i=t}^{T-1} \lambda_i \right) [l_t(x_t) - l_t(u)] . \quad (1)$$

We say an algorithm is *minimax* or *non-adaptive* if given an uncertainty set $\mathcal{S}$, it upper-bounds the *worst case regret*

$$\sup_{(l_{1:T}, u) \in \mathcal{S}} \mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u).$$

This paper aims to design *adaptive* algorithms that directly bound $\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u)$ by a function of the *problem instance* (i.e., both the losses $l_{1:T}$ and the comparator u).

Discounting vs forgetting To motivate the above discounted setting, suppose $\lambda_t \equiv 1$. Then, Eq.(1) recovers the standard undiscounted regret in OCO, from which we can further upper-bound the *total loss* of the player, $\sum_{t=1}^{T} l_t(x_t)$. However, the effectiveness of this follow-up argument relies on the existence of a comparator u with low total loss $\sum_{t=1}^{T} l_t(u)$. We call such an environment "stationary".

This paper concerns the "nonstationary" environments violating the previous argument. Here, the expert provides discount factors $\lambda_{1:T}$ to the player as side information, suggesting the usefulness of each observation for future predictions. For example,

- With $\lambda_t \equiv \lambda < 1$, the weight of past losses decays quickly in Eq.(1), therefore the corresponding algorithm is motivated to "forget the past", matching the intuition developed in Section 1.
- A more extreme case is when λ_t takes value in $\{0, 1\}$. Then, the problem reduces to a restarting variant of standard OCO, where each restart is triggered by $\lambda_t = 0$.

Overall, this paper treats the discount factors $\lambda_{1:T}$ as part of the problem description, and focus on establishing tight upper bounds on Eq.(1). Choosing $\lambda_{1:T}$ online is an important issue, which we defer to future works.

Inductive bias In the typical application of lifelong learning, one would use the output x_t of an online learning algorithm as the parameter of the underlying machine learning model, therefore we consider the inductive bias as a fixed prediction $x^* \in \mathcal{X}$ known at the beginning of the game (possibly obtained from pre-training). Intuitively, simply predicting $x_t = x^*$ would work "decently well", and by further modifying it in the game, the goal is to correct its *time-varying imperfection* using the sequentially revealed online data. Consistent with the arguments from Section 1, an adaptive algorithm that optimally exploits x^* would perform better than algorithms that do not.

Following this intuition, we consider the initialization $x_1 = x^*$ in the discounted OCO game. Without loss of generality, we assume $x^* = 0$ throughout this section, since a different x^* can be implemented by simply shifting the coordinates.

2.1. Preliminary

We begin by introducing the widespread non-adaptive baseline, constant-LR OGD. To this end, notice that the main difference between the discounted regret Eq.(1) and the well-studied undiscounted regret ($\lambda_t \equiv 1$) is the *effective time horizon*. Instead of the maximum length T, in the t-th round Eq.(1) concerns an *exponentially weighted look-back window* of length

$$H_t := \sum_{i=1}^{t} \left(\prod_{j=i}^{t-1} \lambda_j^2 \right) , \quad (2)$$

which is roughly $\min[(1-\lambda^2)^{-1}, T]$ in the special case of $\lambda_t \equiv \lambda < 1$. For later use, we also define the *discounted gradient variance* V_t and the *discounted Lipschitz constant* G_t,

$$V_t := \sum_{i=1}^{t} \left(\prod_{j=i}^{t-1} \lambda_j^2 \right) \|g_i\|^2 ; \; G_t := \max_{i \in [1:t]} \left(\prod_{j=i}^{t-1} \lambda_j \right) \|g_i\| . \quad (3)$$

A classical wisdom in online learning is that the learning rates of OGD should be inversely proportional to the square root of the time horizon (Orabona, 2023a, Chapter 2). Combining it with the effective time horizon discussed above, the following result is a folklore.

[2] The one round delay is due to our regret definition: the loss function l_t is undiscounted in the t-th round.

Online Gradient Descent Consider the OGD update rule: after observing the loss gradient g_t, we pick a learning rate η_t, take a gradient step and project the update back to the domain $\mathcal{X}$, i.e.,

$$x_{t+1} = \Pi_{\mathcal{X}} \left(x_t - \eta_t g_t \right).$$

Theorem 1 (Abridged Theorem 6 and 7)**.** *If the loss functions are all G-Lipschitz, the diameter of the domain is at most D, and the discount factor $\lambda_t = \lambda \in (0, 1)$, then OGD with a constant learning rate $\eta_t = \frac{D}{G}\sqrt{1-\lambda^2}$ guarantees for all $T = \Omega(\frac{1}{1-\lambda})$,*

$$\sup_{l_{1:T}, u} \mathrm{R}_T^{\lambda}(l_{1:T}, u) = O\left(DG\sqrt{H_T}\right).$$

Conversely, fix any variance budget $V \in (0, G^2 H_T]$ and any comparator u such that $u, -u \in \mathcal{X}$. For any algorithm, there exists a loss sequence such that V_T defined in Eq.(3) satisfies $V_T = V$, and

$$\max\left[\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u), \mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, -u)\right] = \Omega\left(\|u\|\sqrt{V_T}\right).$$

Picking the domain $\mathcal{X}$ as a norm ball centered at the origin, one could see that the worst case bound of constant-LR OGD is *minimax optimal* under the Lipschitzness and bounded-domain assumptions, which forms the theoretical foundation of this common practice. Nonetheless, there is an *instance-dependent* gap that illustrates a natural direction to improve the algorithm: removing the supremum, and directly aiming for

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) = O\left(\|u\|\sqrt{V_T}\right). \tag{4}$$

Such a bound is never worse than the $O(DG\sqrt{H_T})$ bound of constant-LR OGD (under the same assumptions), while the associated algorithm can be *agnostic to both D and G*. This is the key strength of adaptivity: without imposing any artificial structural assumption, the algorithm performs as if it knows the "correct" assumption from the beginning.

2.2. Main result

To pursue Eq.(4), our main idea is a simple rescaling trick.

- In the undiscounted setting ($\lambda_t \equiv 1$), Eq.(4) has been almost achieved by several recent works (Cutkosky, 2019; Mhammedi & Koolen, 2020; Jacobsen & Cutkosky, 2022; Zhang et al., 2023) modulo necessary residual factors. For any such algorithm $\mathcal{A}$, let $\mathrm{R}_{T,\mathcal{A}}(g_{1:T}, u)$ denote its undiscounted regret with respect to linear losses $\langle g_t, \cdot \rangle$ and comparator u, i.e., Eq.(1) with $\lambda_t \equiv 1$.
- Next, consider the discounted regret with general $\lambda_{1:T}$. We take an aforementioned algorithm $\mathcal{A}$ and apply it to a sequence of *surrogate loss gradients* $\hat{g}_{1:T}$, where

$$\hat{g}_t = \left(\prod_{i=1}^{t-1} \lambda_i^{-1}\right) g_t. \tag{5}$$

If $\lambda_1, \ldots, \lambda_T < 1$, this amounts to "upweighting" recent losses, or equivalently, "forgetting" older ones. The generated prediction sequence $x_{1:T}$ satisfies a discounted regret bound that scales with $\mathrm{R}_{T,\mathcal{A}}(\hat{g}_{1:T}, u)$, the undiscounted regret of the base algorithm $\mathcal{A}$ on $\hat{g}_{1:T}$.

$$\begin{aligned}
&\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) \\
&= \left(\prod_{t=1}^{T-1} \lambda_t\right) \cdot \sum_{t=1}^{T} \left(\prod_{i=1}^{t-1} \lambda_i^{-1}\right) \left[l_t(x_t) - l_t(u)\right] \\
&\le \left(\prod_{t=1}^{T-1} \lambda_t\right) \mathrm{R}_{T,\mathcal{A}}\left(\hat{g}_{1:T}, u\right).
\end{aligned} \tag{6}$$

Within this trick, a particular challenge is that even if all the actual loss functions l_t are Lipschitz in a *time-uniform* manner ($\exists G, s.t., \max_t \|g_t\| \le G$), the surrogate loss functions $\langle \hat{g}_t, \cdot \rangle$ are not. Therefore, the base algorithm $\mathcal{A}$ cannot rely on any *a priori knowledge or estimate* of the time-uniform Lipschitz constant, similar to the *scale-free* property (Orabona & Pál, 2018) in adaptive online learning.[3] To make this concrete, we first forego the adaptivity to the comparator u and analyze an example based on the famous AdaGrad (Duchi et al., 2011). The obtained algorithm will be a building block of our main results.

Gradient adaptive OGD Proposed for the undiscounted setting, AdaGrad (Duchi et al., 2011) represents OGD with *gradient-dependent learning rates*. Using it as the base algorithm $\mathcal{A}$ leads to the following RMSProp-like prediction rule and its discounted regret bound.

Theorem 2. *If the diameter of $\mathcal{X}$ is at most D, then OGD with learning rate $\eta_t = DV_t^{-1/2}$ guarantees for all $T \in \mathbb{N}_+$ and loss sequence $l_{1:T}$,*

$$\sup_u \mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) \le \frac{3}{2} D\sqrt{V_T}.$$

As one would hope for, the bound strictly improves constant-LR OGD while matching the lower bound on the $\sqrt{V_T}$ dependence. It is tempting to seek an even better learning rate η_t that improves the remaining D to $\|u\|$, but such a direction leads to a dead end (Remark B.1).

[3] A scale-free algorithm generates the same predictions $x_{1:T}$ if all the gradients $g_{1:T}$ are scaled by an arbitrary $c > 0$. However, we need a bit more, since an algorithm can be scale-free even if it requires an estimate of the time-uniform Lipschitz constant at the beginning (Mhammedi & Koolen, 2020; Jacobsen & Cutkosky, 2022).

Algorithm 1 1D magnitude learner on $[0, \infty)$.

Require: Hyperparameter $\varepsilon > 0$ (default $\varepsilon = 1$).

Initialize parameters $v_1 = 0$, $s_1 = 0$, $h_1 = 0$.

for $t = 1, 2, \ldots$ **do**

If $h_t = 0$, define the unprojected prediction $\tilde{x}_t = 0$. Otherwise, with $\operatorname{erfi}(x) := \int_0^x \exp(u^2) du$, (which can be queried from SCIPY and JAX),

$$\tilde{x}_t = \varepsilon \cdot \operatorname{erfi}\left(\frac{s_t}{2\sqrt{v_t + 2h_t s_t + 16h_t^2}}\right) - \frac{\varepsilon h_t}{\sqrt{v_t + 2h_t s_t + 16h_t^2}} \exp\left[\frac{s_t^2}{4(v_t + 2h_t s_t + 16h_t^2)}\right].$$

Predict $x_t = \Pi_{[0,\infty)}(\tilde{x}_t)$, the projection of $\tilde{x}_t$ to the domain $[0, \infty)$.

Receive the 1D loss gradient $g_t \in \mathbb{R}$ and the discount factor $\lambda_{t-1} \in (0, \infty)$.

Clip g_t by defining $g_{t,\text{clip}} = \Pi_{[-\lambda_{t-1}h_t, \lambda_{t-1}h_t]}(g_t)$, and update $h_{t+1} = \max(\lambda_{t-1}h_t, |g_t|)$.

If $g_{t,\text{clip}}\tilde{x}_t < g_{t,\text{clip}}x_t$, define a surrogate loss gradient $\tilde{g}_{t,\text{clip}} = 0$. Otherwise, $\tilde{g}_{t,\text{clip}} = g_{t,\text{clip}}$.

Update $v_{t+1} = \lambda_{t-1}^2 v_t + \tilde{g}_{t,\text{clip}}^2$, $s_{t+1} = \lambda_{t-1}s_t - \tilde{g}_{t,\text{clip}}$.

end for

To solve this problem, we will resort to the *Follow the Regularized Leader* (FTRL) framework (Orabona, 2023a, Chapter 7) instead of OGD. The key intuition (Fang et al., 2022; Jacobsen & Cutkosky, 2022) is that, FTRL is stronger since it *memorizes the initialization*, whereas OGD without extra regularization does not. This is particularly important in the discounted setting: FTRL gradually forgets the past online data but not the inductive bias x^*, whereas OGD forgets everything altogether.

Simultaneous adaptivity To achieve simultaneous adaptivity to both $l_{1:T}$ and u, things can get subtle. As mentioned earlier, there are a number of choices for the base algorithm $\mathcal{A}$. We will adopt the algorithm from (Zhang et al., 2024), surveyed in Appendix B.2, which offers an important benefit (i.e., no explicit T-dependence, Remark B.4). Without loss of generality,[4] assume the domain $\mathcal{X} = \mathbb{R}^d$.

Overall, our discounted algorithm employs the *polar-decomposition technique* from (Cutkosky & Orabona, 2018): using polar coordinates, predicting $x_t \in \mathbb{R}^d$ (to "chase" the optimal comparator u) can be decomposed into two independent tasks, learning the *good direction* $u/\|u\|$ and the *good magnitude* $\|u\|$. The direction is learned by the RMSPROP-like algorithm from Theorem 2, while the magnitude is learned by a discounted variant of the erfi-*potential learner* (Zhang et al., 2024, Algorithm 1) that operates on the non-negative real line $[0, \infty)$. This magnitude learner itself will be important for the OCP application, therefore we present its pseudocode as Algorithm 1.

Algorithm 1 has the following intuition. At its center is a special instance of FTRL building on the adaptive OCO theory, which generates the prediction x_t using the *discounted gradient variance* v_t, the *discounted gradient sum* s_t, and the *discounted Lipschitz constant* h_t. Complementing this core component, two additional ideas are applied to fix certain technical problems: (i) the *unconstrained-to-constrained reduction* from (Cutkosky & Orabona, 2018; Cutkosky, 2020), and (ii) the *hint-and-clipping technique* from (Cutkosky, 2019). The readers are referred to Appendix B.2 for a detailed explanation. The following theorem is proved in Appendix B.3.

Theorem 3. *Given any hyperparameter $\varepsilon > 0$, Algorithm 1 guarantees for all time horizon $T \in \mathbb{N}_+$, loss sequence $l_{1:T}$, comparator $u \in [0, \infty)$ and stability window length $\tau \in [1 : T]$,*

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) \le \varepsilon\sqrt{V_T + 2G_T S + 16G_T^2} + u(S + G_T) + \left(\max_{t\in[T-\tau+1:T]} x_t\right) G_T + \left(\prod_{t=T-\tau}^{T-1} \lambda_t\right)\left(\max_{t\in[1:T-\tau]} x_t\right) G_{T-\tau},$$

where

$$S = 8G_T\left(1 + \sqrt{\log(2u\varepsilon^{-1} + 1)}\right)^2 + 2\sqrt{V_T + 16G_T^2}\left(1 + \sqrt{\log(2u\varepsilon^{-1} + 1)}\right).$$

This result might seem a bit intimidating, so let us take a few steps to interpret it. In particular, we want to justify the appropriate asymptotic regime to consider ($V_T \gg G_T^2$ and $u \gg \varepsilon$), such that our main result on $\mathbb{R}^d$ (Theorem 4) can use the big-Oh notation to improve clarity.

- First, one would typically expect $V_T \gg G_T^2$, since when $\lambda_t = \lambda \in (0, 1)$ and $|g_t| = G$ for all t, we have $G_T = G$ and $V_T = H_T G^2 \approx (1 - \lambda^2)^{-1} G^2$. As long as the discount factor λ_t is close enough to 1, the condition $V_T \gg G_T^2$ is likely to hold for general/practical gradient sequences as well.
- Second, the hyperparameter ε serves as a prior guess of the comparator u. If the guess is correct ($\varepsilon = u$), then by assuming $V_T \gg G_T^2$ and $\max_{t\in[1:T]} x_t = O(u)$ (roughly, the predictions are *stable*), the regret bound becomes

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) = O\left(u\sqrt{V_T}\right), \tag{7}$$

[4] Due to (Cutkosky, 2020, Theorem 2), given any unconstrained algorithm that operates on $\mathcal{X} = \mathbb{R}^d$, we can impose any closed and convex constraint without changing its regret bound.

exactly matching the lower bound. Realistically such an "oracle tuning" is illegal, since ε is selected at the beginning of the game, while reasonable comparators u are hidden before all the loss functions are revealed. This is where our algorithm shines: as long as ε is moderately small, i.e., $O(u)$, we would have the $O(uS)$ term dominating the regret bound, which only suffers a multiplicative *logarithmic penalty* relative to the impossible oracle-optimal rate Eq.(7). In comparison, it is well-known that the regret bound of constant-LR OGD with learning rate η depends *polynomially* on η and η^{-1} (cf., the proof of Theorem 6), which means our algorithm is provably more robust to suboptimal hyperparameter tuning.

- Third, we explain the use of τ. In nonstationary environments, the range of x_t can vary significantly over time, therefore a time-uniform characterization of its stability ($\max_{t\in[1:T]} x_t$, Remark B.2) could be overly conservative. We use a *stability window* of arbitrary length τ to divide the time horizon into two parts: the earlier part is forgotten rapidly (due to the $\prod_{t=T-\tau}^{T-1}\lambda_t$ multiplier), so only the later part really matters.

Example 1. *Suppose again that $\lambda_t = \lambda \in (0,1)$. If $\lambda \approx 1$, the "forgetting" multiplier can be approximated by*

$$\prod_{t=T-\tau}^{T-1} \lambda_t = \lambda^\tau = \lambda^{\frac{1}{1-\lambda}\cdot(1-\lambda)\tau} \approx e^{(\lambda-1)\tau}. \text{ (Lemma B.1)}$$

Consider $\lambda = 0.99$ for example: $\tau = 700$ ensures $\lambda^\tau \approx e^{-7} < 10^{-3}$. That is, if the past range of x_t is negligible after a 10^{-3}-attenuation, then our regret bound only depends on the "localized stability" of x_t evaluated in the recent 700 rounds, which is a small fraction in (let's say) $T \approx$ millions. More intuitively, it means "past mistakes do not matter".

Given the 1D magnitude learner and its guarantee, the extension to $\mathbb{R}^d$ is now standard (Cutkosky & Orabona, 2018). We defer the pseudocode to Appendix B.3.

Theorem 4 (Main result)**.** *Given any hyperparameter $\varepsilon > 0$, Algorithm 3 in Appendix B.3 guarantees for all $T \in \mathbb{N}_+$, loss gradients $g_{1:T}$ and comparator $u \in \mathbb{R}^d$,*

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) \le \left(\max_{t\in[1:T]} x_t\right) G_T$$
$$+O\left(\|u\|\sqrt{V_T \log(\|u\|\,\varepsilon^{-1})} \vee \|u\|\, G_T \log(\|u\|\,\varepsilon^{-1})\right),$$

where $O(\cdot)$ is in the regime of large V_T ($V_T \gg G_T$) and large $\|u\|$ ($\|u\| \gg \varepsilon$). Furthermore, for $u = 0$,

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, 0) \le O\left(\varepsilon\sqrt{V_T}\right) + \left(\max_{t\in[1:T]} x_t\right) G_T.$$

Similar to Theorem 3, the iterate stability term can be split into two parts using an arbitrary τ. The key message is that if the iterates are indeed stable ($\max_{t\in[1:T]} x_t = O(\|u\|)$) and we suppress all the logarithmic factors with $\tilde{O}(\cdot)$, then

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) \le \tilde{O}\left(\|u\|\sqrt{V_T}\right).$$

It matches the lower bound and improves all the aforementioned algorithms. Granted, there are several nuances in this statement, but they are in general necessary even in the undiscounted special case (Orabona, 2023a, Chapter 9).

Finally, we summarize a range of practical strengths. As shown earlier, the proposed algorithm is robust to hyperparameter tuning. Observations and mistakes from the distant past (which are possibly misleading for the future) are appropriately forgotten, such that the algorithm runs "consistently" over its lifetime. Compared to the typical aggregation framework in nonstationary online learning (Daniely et al., 2015; Zhang et al., 2018a), our algorithm runs faster and never explicitly restarts (although we require given discount factors to "softly" restart). In addition, compared to constant-LR OGD that also tries to minimize the discounted regret, our algorithm makes a better use of the inductive bias x^* – in the general case of $x^* \neq 0$, our discounted regret bound scales with $\|u - x^*\|$. This exemplifies the crucial role of regularization, designed from the adaptive OCO theory.

3. Online Conformal Prediction

Complementing the above result, we now consider its application in conformal prediction (Vovk et al., 2005), a framework that quantifies the uncertainty of black box ML models. Specifically, we study its online version (Gibbs & Candes, 2021; Bastani et al., 2022; Gibbs & Candès, 2022; Zaffran et al., 2022; Bhatnagar et al., 2023), where no statistical assumptions (e.g., *exchangeability*) are imposed at all. Our setting follows the nicely written (Bhatnagar et al., 2023, Section 2), and the readers are referred to (Roth, 2022; Angelopoulos & Bates, 2023) for additional background.

3.1. Preliminary

Similar to OCO, OCP is again a two-person repeated game. Let a constant $\alpha \in (0,1)$ be the *targeted miscoverage rate* fixed before the game starts. At the beginning of the t-th round, we receive a set-valued function $\mathcal{C}_t : \mathbb{R}_{\ge 0} \to 2^{\mathcal{Y}}$ mapping any *radius parameter* $r \in [0,\infty)$ to a subset $\mathcal{C}_t(r)$ of the *label space* $\mathcal{Y}$. The $\mathcal{C}_t$ function is *nested*: for any $r' > r$, we have $\mathcal{C}_t(r) \subset \mathcal{C}_t(r') \subset \mathcal{Y}$. Then,

1. We pick a radius parameter $r_t \in [0,\infty)$ and output the prediction set $\mathcal{C}_t(r_t)$.
2. The environment reveals the *optimal radius* $r_t^* \in [0,\infty)$. Intuitively, our prediction set $\mathcal{C}_t(r_t)$ is "large enough" only if $r_t > r_t^*$.
3. Our performance is evaluated by the *pinball loss*

$l_t^{(\alpha,*)}(r_t)$, where for all $r \in [0, \infty)$,

$$l_t^{(\alpha,*)}(r) := \begin{cases} \alpha(r - r_t^*), & r > r_t^*, \\ (\alpha - 1)(r - r_t^*), & \text{else.} \end{cases} \tag{8}$$

To demonstrate this framework, here is a simple 1D forecasting example. Suppose there is a base ML model that in each round makes a prediction $\hat{x}_t \in \mathbb{R}$ of the true time series $x_t^* \in \mathbb{R}$. On top of that, we "wrap" such a point prediction x_t by a *confidence set* prediction $\mathcal{C}_t(r_t) = (\hat{x}_t - r_t, \hat{x}_t + r_t)$. Ideally we want $\mathcal{C}_t(r_t)$ to cover the true series x_t^*, and this can be checked after x_t^* is revealed. That is, by defining the optimal radius $r_t^* = |x_t^* - \hat{x}_t|$, we claim success if $r_t > r_t^*$.

Quite naturally, since r_t^* is arbitrary, it is impossible to *ensure* coverage unless r_t is meaninglessly large. A reasonable objective is then asking our *empirical (marginal) coverage rate* to be approximately $1 - \alpha$, which amounts to showing

$$\left| \frac{1}{T} \sum_{t=1}^{T} \mathbf{1}\left[r_t \le r_t^* \right] - \alpha \right| = o(1), \tag{9}$$

and this is beautifully *equivalent* to characterizing the cumulative subgradients of the pinball loss,[5] $\left| \sum_{t=1}^{T} \partial l_t^{(\alpha,*)}(r_t) \right| = o(T)$. One catch is that there are trivial predictors[6] satisfying Eq.(9) (Bastani et al., 2022), so one needs an extra measure to rule them out. Such a "secondary objective" can be the regret bound on the pinball loss,

$$\sum_{t=1}^{T} l_t^{(\alpha,*)}(r_t) - \sum_{t=1}^{T} l_t^{(\alpha,*)}(u) = o(T), \tag{10}$$

which motivates using OCO algorithms to select r_t.

How does nonstationarity enter the picture? Since the proposal of OCP in (Gibbs & Candes, 2021), the main emphasis is on problems with *distribution shifts*, which traditional conformal prediction methods based on exchangeability and data splitting have trouble dealing with. For example, the popular ACI algorithm (Gibbs & Candes, 2021) essentially uses constant-LR OGD – as we have shown, this is inconsistent with minimizing the standard regret Eq.(10), and the "right" OCO performance metric that justifies it could be a nonstationary one (e.g., the discounted regret). In a similar spirit, (Gibbs & Candès, 2022; Bhatnagar et al., 2023) apply *dynamic* and *strongly adaptive* OCO algorithms, effectively analyzing the subinterval variants of Eq.(9) and Eq.(10).

Another key ingredient of OCP is assuming the optimal radius $\max_t r_t^* \le D$ for some $D > 0$, which is often reasonable in practice and important for the coverage guarantee. As opposed to prior works (Bastani et al., 2022; Bhatnagar et al., 2023) that require *knowing* D at the beginning to initialize properly, we seek an adaptive algorithm agnostic to this oracle knowledge.

Our goal Overall, we aim to show that without knowing D, applying Algorithm 1 leads to *discounted adaptive* versions of the marginal coverage bound Eq.(9) and the regret bound Eq.(10). This offers advantages over the ACI-like approach that tackles similar sliding window objectives using constant-LR OGD. Along the way, we demonstrate how the structure of OCP allows controlling the iterate stability of Algorithm 1 (or generally, FTRL algorithms), which then makes the proof of coverage fairly easy. Due to the limited space, experiments are presented in Appendix D.

3.2. Main result

Beyond pinball loss From now on, define $\mathcal{A}_{CP}$ as the OCP algorithm that uses Algorithm 1 to select r_t (see Appendix C.1 for pseudocode). We make a major generalization: instead of using subgradients of the pinball loss Eq.(8) to update Algorithm 1, we use subgradients $g_t^* \in \partial f_t^*(r_t)$, where $f_t^*(r)$ is any convex function minimized at r_t^*, and right at $r_t = r_t^*$ we have $g_t^* \le 0$ without loss of generality. This includes the pinball loss $l_t^{(\alpha,*)}(r)$ as a special case. Notably, $f_t^*(r)$ does not need to be globally Lipschitz, which unleashes the full power of our base algorithm. Put together, $\mathcal{A}_{CP}$ takes a confidence hyperparameter ε and a sequence of discount factors $\lambda_{1:T}$ determined by our objectives. We assume $\max_t r_t^* \le D$, but D is unknown by $\mathcal{A}_{CP}$.

Strength of FTRL The key to our result is Lemma 3.1 connecting the prediction to the *discounted coverage metric*

$$S_t^* := -\sum_{i=1}^{t} \left(\prod_{j=i}^{t-1} \lambda_j \right) g_i^*. \tag{11}$$

If $\lambda_t = 1$ for all t and we use the pinball loss to define $g_{1:T}^*$, then $|S_T^*| / T$ recovers Eq.(9). The point is that if $\lambda_t = \lambda < 1$, then just like the intuition throughout this paper, Eq.(11) is essentially a sliding window coverage metric. Associated algorithms would gradually forget the past, which intuitively counters the distribution shifts.

Lemma 3.1 (Abridged Lemma C.2)**.** *$\mathcal{A}_{CP}$ guarantees for all $t \in \mathbb{N}_+$,*

$$|S_t^*| \le O\left(\sqrt{V_t^* \log(r_{t+1}\varepsilon^{-1})} \vee G_t^* \log(r_{t+1}\varepsilon^{-1}) \right),$$

where V_t^* and G_t^* are defined on the OCP loss gradients using Eq.(3), and $O(\cdot)$ is in the regime of $r_{t+1} \gg \varepsilon$.

The proof of this lemma is a bit involved, but the high level idea is very simple: if we use a FTRL algorithm (rather than OGD) as the OCO subroutine for OCP, then

[5] At the singular point r_t^*, define $\partial l_t^{(\alpha,*)}(r_t^*) = \alpha - 1$.

[6] Alternating between $r_t = \infty$ and $r_t = 0$ independent of data.

the radius prediction is roughly $r_{t+1} \approx \psi(S_t^*/\sqrt{V_t^*})$ for some prediction function ψ (if the algorithm is not "adaptive enough" then the denominator is $\sqrt{H_t}$ instead), which means $S_t^* \approx \sqrt{V_t^*}\psi^{-1}(r_{t+1}) = O(\sqrt{H_t})$. Dividing both sides by H_t yields the desirable coverage guarantee. In comparison, the parallel analysis using OGD can be much more complicated (e.g., the grouping argument in (Bhatnagar et al., 2023)) due to the absence of S_t^* in the explicit update rule. Such an analytical strength of FTRL seems to be overlooked in the OCP literature.

We also note that although the above lemma depends only logarithmically on r_{t+1}, without any problem structure the latter could still be large (exponential in t), which invalidates this approach. The remaining step is showing that if the underlying optimal radius r_t^* is time-uniformly bounded by D, then even without knowing D, we could replace r_{t+1} in the above lemma by $O(D)$. Intuitively it should make sense; materializing it carefully gives us the final result.

Theorem 5. *Without knowing D, $\mathcal{A}_{CP}$ guarantees that for all $T \in \mathbb{N}_+$, we have the discounted coverage bound*

$$|S_T^*| \le O\left(\sqrt{V_T^* \log(D\varepsilon^{-1})} \vee G_T^* \log(D\varepsilon^{-1})\right),$$

and the discounted regret bound from Theorem 3.

To interpret this result, we focus on the coverage bound, since the discounted regret bound has been discussed extensively in Section 2. First, consider the pinball loss and the undiscounted setting $\lambda_t = 1$, where our bound can be directly compared to prior works. For OGD, (Gibbs & Candes, 2021, Proposition 1) shows that the learning rate $\eta = D/\sqrt{T}$ (as suggested by regret minimization) achieves $|S_T^*| = O(\sqrt{T})$, while (Bhatnagar et al., 2023, Theorem 2) shows that $\eta_t = D/\sqrt{V_t}$ (i.e., AdaGrad) achieves $|S_T^*| = O(\alpha^{-2}T^{3/4}\log T)$. Although the latter is empirically strong, the theory is a bit unsatisfying as one would expect the gradient adaptive approach to be a "pure upgrade" (plus, the bound blows up as $\alpha \to 0$). Our Theorem 5 is in some sense the "right fix", as essentially, $|S_T^*| = \tilde{O}(\sqrt{V_T^*}) \le \tilde{O}(\sqrt{T})$. This is primarily due to the strength of FTRL over OGD, which we hope to demonstrate. Besides, our algorithm also improves the regret bound of these baselines while being agnostic to D.

All these algorithms can be extended to the sliding window setting (e.g., the algorithm from (Gibbs & Candes, 2021) becomes constant-LR OGD, which is the version actually applied in practice), and the above comparison still roughly holds. There is just one catch: OGD algorithms make coverage guarantees on "exact sliding windows" $[T-H+1:T]$ of length H, whereas our algorithm bounds the coverage on a slightly different "exponential window" of effective length H_T, Eq.(11). Nonetheless, all these algorithms also guarantee the *same type* of discounted regret bounds, which are still comparable like in Section 2.

Adaptivity in OCP Next, we discuss the benefits of gradient adaptivity more concretely in OCP. Suppose again that $\lambda_t = 1$ and we use the pinball loss. Then, instead of the non-adaptive bound $|S_T^*| = O(\sqrt{T})$, we have

$$\begin{aligned}|S_T^*| &= \tilde{O}\left(\sqrt{V_T^*}\right) \\ &= \tilde{O}\left(\sqrt{\alpha^2 \sum_{t=1}^T \mathbf{1}[r_t > r_t^*] + (\alpha-1)^2 \sum_{t=1}^T \mathbf{1}[r_t \le r_t^*]}\right).\end{aligned}$$

Since asymptotically the miscoverage rate is α, we have $\sum_{t=1}^T \mathbf{1}[r_t \le r_t^*] \approx \alpha T$, which means the bound is roughly $\tilde{O}\left(\sqrt{\alpha(1-\alpha)T}\right)$. That is, the gradient adaptivity in OCO translates to the *target rate adaptivity* in OCP.

In terms of the sample complexity, it is then straightforward to see that the OGD baseline requires at least ε^{-2} rounds to guarantee an empirical marginal coverage rate within $[\alpha-\varepsilon, \alpha+\varepsilon]$, while our algorithm requires at least $\alpha\varepsilon^2$ rounds. Very concretely, in the typical setting of $\alpha = 0.05$ (i.e., we want 95% confidence sets), our algorithm only requires $\frac{1}{20}$ as many samples compared to the OGD baseline. Besides the coverage bound, such an α-dependent saving applies to the regret bound as well.

4. Experiment

Finally, we demonstrate the practicality of our algorithm in OCP experiments. Our setup closely builds on (Bhatnagar et al., 2023). Except our own algorithms, we adopt the implementation of the baselines and the evaluation procedure from there. Details are deferred to Appendix D.

Setup We consider image classification in a sequential setting, where each image is subject to a corruption of time-varying strength. Given a base machine learning model that generates the $\mathcal{C}_t$ function (by scoring all the possible labels), the goal of OCP is to select the radius parameter r_t, which yields a set of predicted labels. Ideally, we want such a set to contain the true label, while being as small as possible. The targeted miscoverage rate α is selected as 0.1.

We test three versions of our algorithm: MAGL-D is our Algorithm 1 with $\varepsilon = 1$ and $\lambda_t = 0.999$; MAGL is its undiscounted version ($\lambda_t = 1$); and MAGDIS is a much simplified variant of Algorithm 1 that basically sets $h_t = 0$. We are aware that a possible complaint towards our approach is that the algorithm is too complicated, but as we will show, this simplified version presented as Algorithm 5 also demonstrates strong empirical performance despite losing the performance guarantee.

The baselines we test are summarized in Table 1, following

Method	*Avg. Coverage*	*Avg. Width*	LCE_{100}	*Runtime*
Simple OGD	**0.899**	125.0	0.11	$\mathbf{1.00 \pm 0.03}$
SF-OGD	**0.899**	124.5	0.09	$\underline{1.05 \pm 0.03}$
SAOCP	0.883	$\underline{119.2}$	0.10	11.07 ± 0.19
SplitConformal	0.843	129.5	0.47	1.57 ± 0.04
NExConformal	0.892	123.0	0.14	2.22 ± 0.02
FACI	0.889	123.4	0.12	2.98 ± 0.07
FACI-S	0.892	124.7	0.11	2.17 ± 0.07
Alg. 1: MAGL-D	0.884	**118.5**	$\underline{0.08}$	1.06 ± 0.03
Alg. 2: MAGL	$\underline{0.894}$	122.1	0.09	1.05 ± 0.02
Alg. 5: MAGDIS	0.888	122.1	**0.07**	1.02 ± 0.04

Table 1. Performance of different methods. Baselines are colored in black, while our algorithms are colored in blue. The best performer in each metric is **bolded** and the second-best is underlined. The runtime, plus/minus its standard deviation, is normalized by the mean of Simple OGD's runtime; averages are take over ten trials per algorithm.

the implementation of (Bhatnagar et al., 2023). In particular, SF-OGD (Bhatnagar et al., 2023) is equivalent to ADAGRAD with *oracle tuning*: by definition it requires knowing the maximum possible radius D to set the learning rate, and in practice, D is estimated from an offline dataset (which is a form of oracle tuning from the theoretical perspective). To test the effect of such tuning, we create another baseline called "Simple OGD", which is simply SF-OGD with its estimate of D set to 1. We emphasize that despite its name, Simple OGD is still a gradient adaptive algorithm.

Four metrics are evaluated, and we define them formally in Appendix D. First, the *average coverage* measures the empirical coverage rate over the entire time horizon. Similarly, the *average width* refers to the average size of the prediction set, also over the entire time horizon. Ideally we want the average coverage to be close to $1 - \alpha = 0.9$, and if that is satisfied, lower average width is better. Different from these two, the *local coverage error* (LCE_{100}) measures the deviation of the *local* empirical coverage rate (over the "worst" sliding time window of length 100) from the target rate 0.9 – this is arguably the most important metric (due to the distribution shifts), and lower is better. Finally, we also test the runtime of all the algorithms, normalized by that of Simple OGD.

Result The results are summarized in Table 1. Among all the algorithms tested, our algorithms achieve the lowest local coverage error (LCE_{100}). In terms of metrics on the entire time horizon, our algorithms also demonstrate competitive performance compared to the baselines: although the average coverage is worse than that of Simple OGD and SF-OGD, the average width is lower. In addition, our algorithms run almost as fast as Simple OGD and SF-OGD, and importantly, they are significantly faster than SAOCP (Bhatnagar et al., 2023) which is an aggregation algorithm that minimizes the strongly adaptive regret.

By comparing the three versions of our algorithm, as one would expect, the discounted version (MAGL-D) improves the undiscounted version (MAGL). Remarkably, the much simplified MAGDIS achieves competitive performance despite the lack of performance guarantees.

5. Conclusion

This work revisits the classical notion of discounted regret using recently developed techniques in adaptive online learning. In particular, we propose a discounted "simultaneously" adaptive algorithm (with respect to both the loss sequence and the comparator), with demonstrated practical benefits in online conformal prediction. Along the way, we propose (i) a simple rescaling trick to minimize the discounted regret; and (ii) a FTRL-based analytical strategy to guarantee coverage in online conformal prediction. More broadly, we show that the adaptive OCO theory can help designing better regularization mechanisms for continual / lifelong learning, where forgetting is necessary.

Moving forward, we hope the present work can help revive the community's interest in the discounted regret. For non-stationary online learning, it is a simpler metric to study than the alternatives, while offering certain practical advantages (Appendix A). We leave the online selection of the discount factors as an important open question. In addition, there are recent works (Cutkosky et al., 2023; Ahn et al., 2024) suggesting the connection between discounting and deep learning optimization (e.g., ADAM), which could be an exciting direction for future research.

Acknowledgements

This research was supported by Harvard University.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. The algorithms in this paper can broadly help learning, prediction, and decision-making in dynamic environments with distribution shifts. The main focus is theoretical, therefore we do not foresee any negative societal impact.

References

Abel, D., Barreto, A., Van Roy, B., Precup, D., van Hasselt, H., and Singh, S. A definition of continual reinforcement learning. *arXiv preprint arXiv:2307.11046*, 2023.

Abernethy, J., Hazan, E. E., and Rakhlin, A. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Conference on Learning Theory*, pp. 263–273, 2008.

Adamskiy, D., Koolen, W. M., Chernov, A., and Vovk, V. A closer look at adaptive regret. *Journal of Machine Learning Research*, 17(1):706–726, 2016.

Ahn, K., Zhang, Z., Kook, Y., and Dai, Y. Understanding Adam optimizer via online learning of updates: Adam is FTRL in disguise. *arXiv preprint arXiv:2402.01567*, 2024.

Angelopoulos, A. N. and Bates, S. Conformal prediction: A gentle introduction. *Foundations and Trends® in Machine Learning*, 16(4):494–591, 2023.

Baby, D. and Wang, Y.-X. Optimal dynamic regret in exp-concave online learning. In *Conference on Learning Theory*, pp. 359–409. PMLR, 2021.

Baby, D. and Wang, Y.-X. Optimal dynamic regret in proper online learning with strongly convex losses and beyond. In *International Conference on Artificial Intelligence and Statistics*, pp. 1805–1845. PMLR, 2022.

Barber, R. F., Candes, E. J., Ramdas, A., and Tibshirani, R. J. Conformal prediction beyond exchangeability. *The Annals of Statistics*, 51(2):816–845, 2023.

Bastani, O., Gupta, V., Jung, C., Noarov, G., Ramalingam, R., and Roth, A. Practical adversarial multivalid conformal prediction. *Advances in Neural Information Processing Systems*, 35:29362–29373, 2022.

Bhatnagar, A., Wang, H., Xiong, C., and Bai, Y. Improved online conformal prediction via strongly adaptive online learning. In *International Conference on Machine Learning*, pp. 2337–2363. PMLR, 2023.

Brown, N. and Sandholm, T. Solving imperfect-information games via discounted regret minimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 1829–1836, 2019.

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). *Advances in Neural Information Processing Systems*, 25, 2012.

Chen, L., Luo, H., and Wei, C.-Y. Impossible tuning made possible: A new expert algorithm and its applications. In *Conference on Learning Theory*, pp. 1216–1259. PMLR, 2021.

Chernov, A. and Zhdanov, F. Prediction with expert advice under discounted loss. In *International Conference on Algorithmic Learning Theory*, pp. 255–269. Springer, 2010.

Cutkosky, A. Artificial constraints and hints for unbounded online learning. In *Conference on Learning Theory*, pp. 874–894. PMLR, 2019.

Cutkosky, A. Parameter-free, dynamic, and strongly-adaptive online learning. In *International Conference on Machine Learning*, pp. 2250–2259. PMLR, 2020.

Cutkosky, A. and Mhammedi, Z. Fully unconstrained online learning. *arXiv preprint 2405.20540*, 2024.

Cutkosky, A. and Orabona, F. Black-box reductions for parameter-free online learning in banach spaces. In *Conference On Learning Theory*, pp. 1493–1529. PMLR, 2018.

Cutkosky, A., Mehta, H., and Orabona, F. Optimal, stochastic, non-smooth, non-convex optimization through online-to-non-convex conversion. In *International Conference on Machine Learning*, pp. 6643–6670. PMLR, 2023.

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

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., and Tuytelaars, T. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

Drenska, N. and Kohn, R. V. Prediction with expert advice: A PDE perspective. *Journal of Nonlinear Science*, 30(1): 137–173, 2020.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7), 2011.

Fang, H., Harvey, N. J., Portella, V. S., and Friedlander, M. P. Online mirror descent and dual averaging: keeping pace in the dynamic case. *Journal of Machine Learning Research*, 23(1):5271–5308, 2022.

Freund, Y. and Hsu, D. A new hedging algorithm and its application to inferring latent random variables. *arXiv preprint arXiv:0806.4802*, 2008.

Gibbs, I. and Candes, E. Adaptive conformal inference under distribution shift. *Advances in Neural Information Processing Systems*, 34:1660–1672, 2021.

Gibbs, I. and Candès, E. Conformal inference for online prediction with arbitrary distribution shifts. *arXiv preprint arXiv:2208.08401*, 2022.

Haagerup, U. The best constants in the khintchine inequality. *Studia Mathematica*, 70(3):231–283, 1981.

Harvey, N. J., Liaw, C., Perkins, E., and Randhawa, S. Optimal anytime regret with two experts. *Mathematical Statistics and Learning*, 6(1):87–142, 2023.

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

Ivgi, M., Hinder, O., and Carmon, Y. Dog is SGD's best friend: A parameter-free dynamic step size schedule. In *International Conference on Machine Learning*, pp. 14465–14499. PMLR, 2023.

Jacobsen, A. and Cutkosky, A. Parameter-free mirror descent. In *Conference on Learning Theory*, pp. 4160–4211. PMLR, 2022.

Jacobsen, A. and Cutkosky, A. Online linear regression in dynamic environments via discounting. *arXiv preprint arXiv:2405.19175*, 2024.

Jun, K.-S., Orabona, F., Wright, S., and Willett, R. Improved strongly adaptive online learning using coin betting. In *Artificial Intelligence and Statistics*, pp. 943–951. PMLR, 2017.

Kapralov, M. and Panigrahy, R. Prediction strategies without loss. *Advances in Neural Information Processing Systems*, 24, 2011.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lu, Z. and Hazan, E. On the computational efficiency of adaptive and dynamic regret minimization. *arXiv preprint arXiv:2207.00646*, 2023.

McMahan, H. B. and Orabona, F. Unconstrained online linear learning in hilbert spaces: Minimax algorithms and normal approximations. In *Conference on Learning Theory*, pp. 1020–1039. PMLR, 2014.

Mhammedi, Z. and Koolen, W. M. Lipschitz and comparator-norm adaptivity in online learning. In *Conference on Learning Theory*, pp. 2858–2887. PMLR, 2020.

Orabona, F. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2023a.

Orabona, F. Normalized gradients for all. *arXiv preprint arXiv:2308.05621*, 2023b.

Orabona, F. and Pál, D. Coin betting and parameter-free online learning. *Advances in Neural Information Processing Systems*, 29, 2016.

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

Orabona, F. and Pál, D. Parameter-free stochastic optimization of variationally coherent functions. *arXiv preprint arXiv:2102.00236*, 2021.

Roth, A. Uncertain: Modern topics in uncertainty estimation, 2022.

Vovk, V., Gammerman, A., and Shafer, G. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

Zaffran, M., Féron, O., Goude, Y., Josse, J., and Dieuleveut, A. Adaptive conformal predictions for time series. In *International Conference on Machine Learning*, pp. 25834–25866. PMLR, 2022.

Zhang, L., Lu, S., and Zhou, Z.-H. Adaptive online learning in dynamic environments. *Advances in neural information processing systems*, 31, 2018a.

Zhang, L., Yang, T., and Zhou, Z.-H. Dynamic regret of strongly adaptive methods. In *International Conference on Machine Learning*, pp. 5882–5891. PMLR, 2018b.

Zhang, Z., Cutkosky, A., and Paschalidis, I. PDE-based optimal strategy for unconstrained online learning. In *International Conference on Machine Learning*, pp. 26085–26115. PMLR, 2022.

Zhang, Z., Cutkosky, A., and Paschalidis, I. C. Unconstrained dynamic regret via sparse coding. *arXiv preprint arXiv:2301.13349*, 2023.

Zhang, Z., Yang, H., Cutkosky, A., and Paschalidis, I. C. Improving adaptive online learning using refined discretization. In *International Conference on Algorithmic Learning Theory*, pp. 1208–1233. PMLR, 2024.

Zhao, P., Zhang, Y.-J., Zhang, L., and Zhou, Z.-H. Dynamic regret of convex and smooth functions. *Advances in Neural Information Processing Systems*, 33:12510–12520, 2020.

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

Appendix

Organization Appendix A surveys the related topic of dynamic and strongly adaptive regret. Appendix B and C contain the details of discounted online learning and the OCP application, respectively. Details of the experiments are presented in Appendix D.

A. Dynamic and Strongly Adaptive Regret

This section surveys the recent predominant approach in nonstationary online learning, namely the aggregation-type algorithms that minimize either the dynamic regret or the strongly adaptive regret. We discuss the main idea behind these algorithms, as well as the possible practical concerns.

Definition For clarity, let us assume the diameter of the domain $\mathcal{X}$ is at most D.

- Generalizing the undiscounted static regret, the *Zinkevich-style dynamic regret* (Zinkevich, 2003; Zhang et al., 2018a;b; Zhao et al., 2020; Baby & Wang, 2021; 2022; Jacobsen & Cutkosky, 2022; Lu & Hazan, 2023; Zhang et al., 2023) allows the comparator $u \in \mathcal{X}$ to be time-varying. Formally, the goal is to upper-bound

$$\mathrm{R}_T(l_{1:T}, u_{1:T}) := \sum_{t=1}^{T} l_t(x_t) - \sum_{t=1}^{T} l_t(u_t),$$

  for all comparator sequences $u_1, \dots, u_T$. If the losses are G-Lipschitz, then a typical dynamic regret bound has the form

$$\mathrm{R}_T(l_{1:T}, u_{1:T}) \le \tilde{O}\left(G\sqrt{DP_uT}\right),$$

  where $P_u := \sum_{t=1}^{T-1} \|u_t - u_{t+1}\|$ is called the *path length* of the $u_{1:T}$ sequence.

- Alternatively, the *(strongly) adaptive regret* (Hazan & Seshadhri, 2009; Daniely et al., 2015; Adamskiy et al., 2016; Jun et al., 2017; Cutkosky, 2020; Lu & Hazan, 2023) generalizes the undiscounted static regret to *subintervals* of the time horizon. Formally, for a time interval $\mathcal{I} \subset [1:T]$, we define

$$\mathrm{R}_{\mathcal{I}}(l_{\mathcal{I}}, u) := \sum_{t\in\mathcal{I}} l_t(x_t) - \sum_{t\in\mathcal{I}} l_t(u),$$

  and with G-Lipschitz losses, the typical goal is to show that simultaneously on all time intervals $\mathcal{I}$,

$$\mathrm{R}_{\mathcal{I}}(l_{\mathcal{I}}, u) \le \tilde{O}\left(DG\sqrt{|\mathcal{I}|}\right),$$

  regardless of the specific loss sequence and the comparator. Note that the name "strongly adaptive" is due to historical reasons; in general it is narrower that the recent concept of *adaptive online learning* in the literature. The latter (adopted in this work) refers to achieving instance-dependent performance guarantees.

Hidden in the above definitions is an implicit dependence on the nonstationarity of the environment. Taking the dynamic regret for example,[7] the standard follow-up reasoning is bounding the total loss of the algorithm, $\sum_{t=1}^{T} l_t(x_t)$, through the *oracle inequality*,

$$\sum_{t=1}^{T} l_t(x_t) \le \inf_{u_{1:T}} \left[\sum_{t=1}^{T} l_t(u_t) + \tilde{O}\left(G\sqrt{DP_uT}\right)\right].$$

Although the dynamic regret bound holds for all comparator sequences $u_{1:T}$, the only important ones are those with low total loss, $\sum_{t=1}^{T} l_t(u_t)$. In this way, the path length P_u of the "important comparator sequence" essentially measures the variation of the loss sequence $l_{1:T}$.

[7] For the strongly adaptive regret, a similar argument can be made using the length $|\mathcal{I}|$ of "important time intervals", where the environment is almost stationary and a time-invariant comparator $u \in \mathcal{X}$ induces low loss.

Algorithm Due to the intricate connections between these two performance metrics (Zhang et al., 2018b; Cutkosky, 2020; Baby & Wang, 2021), all known algorithms that minimize either of them share the same two-level compositional design philosophy:

1. The low level maintains a class of "base online learning algorithms" in parallel, each targeting a different "nonstationarity level" of the environment that is possibly correct.
2. The high level aggregates these base algorithms, in order to adapt to the true nonstationarity level unknown beforehand.

Within this procedure, a particularly important consideration is the *range* of targeted nonstationarity levels. This determines the amount of base algorithms maintained at the same time, thus affects the overall statistical and computational performance.

Practical concern Following the above reasoning, we argue that minimizing either the dynamic regret or the strongly adaptive regret requires targeting a wide range of nonstationarity levels, which is sometimes impractical. For example, in the dynamic regret, the path length P_u of the "important comparator sequence" can take any value from $\Theta(1)$ to $\Theta(T)$, whose ratio grows with T. From the computational perspective, the consequence is that the standard model selection approach on an exponentially spaced grid of P_u requires maintaining $O(\log t)$ base algorithms in the t-th round,[8] making the entire algorithm slower over time. Furthermore, excessively conflicting beliefs $P_u = \Theta(1)$ and $P_u = \Theta(T)$ are maintained simultaneously, with the former advocating convergence and the latter on the exact contrary. As we demonstrate shortly, this may cause "statistical failures": even in periodic environments where $P_u = \Theta(T)$ is the only "correct belief", the algorithm may still be deceived by the possibility of $P_u = \Theta(1)$ and converge to trivial time-invariant decisions.

Possible example of failure Consider a 1D OCO problem inspired by time series forecasting, where the loss functions $l_{1:T}$ are the quadratic loss with respect to a ground truth $z_{1:T}$ sequence. That is,

$$l_t(x) = (x - z_t)^2 .$$

Let the ground truth be a unit square wave with period $2H$: $z_t = (-1)^{\lfloor (t-1)/H \rfloor}$. Furthermore, we set the domain $\mathcal{X} = [-2, 2]$, such that the Lipschitz constant $G = 6$.

Consider any algorithm with an optimal P_u-dependent dynamic regret bound, or an optimal strongly adaptive regret bound for 1D OCO (with bounded domain and Lipschitz losses). Then, as a consequence, the algorithm also guarantees a sublinear *static regret bound*,

$$\sum_{t=1}^{T} l_t(x_t) \le \min_{u \in \mathcal{X}} \sum_{t=1}^{T} l_t(u) + o(T).$$

Suppose the time horizon T is a multiple of $2H$. Then,

$$\sum_{t=1}^{T} l_t(u) = \frac{T}{2} (u - 1)^2 + \frac{T}{2} (u + 1)^2 = T(u^2 + 1),$$

which means that the optimal fixed comparator is $u = 0$. It suggests that the x_t sequence generated by the algorithm converges to the trivial time-invariant prediction 0 in the long run, even though the ground truth z_t does not converge. (Technically, the algorithm can do better by "learning" the periodicity of the ground truth and "changing the direction" accordingly, e.g., always predicting a nonzero x_t of the *same sign* as z_t. However, since the algorithm is designed for adversarial environments which are not necessarily periodic, such a strategic behavior is unlikely to hold.) To be more specific, we expect the x_t sequence to track the ground truth z_t sequence, but less and less responsively as t increases (which could be called "degradation"), and eventually x_t converges to 0.

We performed preliminary numerical experiments to validate this hypothesis. The expected degradation is clearly observed on the algorithm from (Jacobsen & Cutkosky, 2022), but for the structurally simpler meta-expert algorithm from (Zhang et al., 2018a), the degradation of the x_t sequence is not significant given the scale of our preliminary experiments. We defer a thorough investigation of this problem to future works.

[8] A notable exception is (Lu & Hazan, 2023), which shows that $\log \log t$ computation per round is achievable at the expense of slightly larger regret bounds.

Takeaway Back to our main argument, we aim to show that for a nonstationary online learning algorithm, it could be impractical to simultaneously maintain a wide range of beliefs on the correct nonstationarity level (of the environment). In some sense, this is on the contrary of a common perception of the field (i.e., the adaptivity to the true nonstationarity level is always better). Our study of discounted algorithms is partially motivated by this idea: such discounted algorithms only target one nonstationarity level, which is specified by the given discount factor.

B. Detail of Section 2

Appendix B.1 contains omitted proofs from Section 2, excluding the analysis of our main algorithm. Appendix B.2 introduces the undiscounted algorithm from (Zhang et al., 2024) and its guarantees; these are applied to our reduction from Section 2.2. Appendix B.3 proves our main result, i.e., discounted regret bounds that adapt simultaneously to $l_{1:T}$ and u.

B.1. Preliminary proofs

We first prove an auxiliary lemma.

Lemma B.1. *For all $\lambda \in (0,1)$, $\lambda^{\frac{1}{1-\lambda}} \le e^{-1}$. Moreover, $\lim_{\lambda\to 1^-} \lambda^{\frac{1}{1-\lambda}} = e^{-1}$.*

Proof of Lemma B.1. For the first part of the lemma, taking $\log$ on both sides, it suffices to show

$$\frac{1}{1-\lambda}\log\lambda \le -1.$$

This holds due to $\log\lambda \le \lambda - 1$. The second part is due to $\lim_{\lambda\to 1^-}(1-\lambda)^{-1}\log\lambda = -1$. □

The following theorem characterizes non-adaptive OGD.

Theorem 6. *If the loss functions are all G-Lipschitz and the diameter of the domain $\operatorname{diam}(\mathcal{X})$ is at most D, then OGD with learning rate $\eta_t = DG^{-1}H_t^{-1/2}$ guarantees for all $T \in \mathbb{N}_+$,*

$$\sup_{l_{1:T},u} \mathrm{R}_T^{\lambda_{1:T}}(l_{1:T},u) \le \frac{3}{2}DG\sqrt{H_T}.$$

Furthermore, if the discount factor $\lambda_t = \lambda$ for some $\lambda \in (0,1)$, then OGD with a time-invariant learning rate $\eta_t = DG^{-1}\sqrt{1-\lambda^2}$ guarantees for all $T \ge \frac{1}{2}(1-\lambda)^{-1}$,

$$\sup_{l_{1:T},u} \mathrm{R}_T^{\lambda}(l_{1:T},u) \le \frac{3}{2}\frac{DG}{\sqrt{1-\lambda^2}} \le \frac{3}{2\sqrt{1-e^{-1}}}DG\sqrt{H_T}.$$

Proof of Theorem 6. We start with the first part of the theorem. The standard analysis of gradient descent centers around the *descent lemma* (Orabona, 2023a, Lemma 2.12),

$$\langle g_t, x_t - u\rangle \le \frac{1}{2\eta_t}\|x_t-u\|^2 - \frac{1}{2\eta_t}\|x_{t+1}-u\|^2 + \frac{\eta_t}{2}\|g_t\|^2.$$

Applying convexity and taking a telescopic sum,

$$\begin{aligned}
&\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T},u)\\
=\ &\sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)[l_t(x_t)-l_t(u)]\\
\le\ &\sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\langle g_t, x_t-u\rangle\\
\le\ &\sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\left(\frac{1}{2\eta_t}\|x_t-u\|^2 - \frac{1}{2\eta_t}\|x_{t+1}-u\|^2 + \frac{\eta_t}{2}\|g_t\|^2\right)
\end{aligned}$$

$$= \frac{1}{2\eta_1}\|x_1-u\|^2\left(\prod_{i=1}^{T-1}\lambda_i\right)+\frac{1}{2}\sum_{t=2}^{T}\left(\frac{1}{\eta_t}-\frac{\lambda_{t-1}}{\eta_{t-1}}\right)\left(\prod_{i=t}^{T-1}\lambda_i\right)\|x_t-u\|^2+\sum_{t=1}^{T}\frac{\eta_t}{2}\left(\prod_{i=t}^{T-1}\lambda_i\right)\|g_t\|^2. \tag{12}$$

Now, consider the second sum on the RHS. Notice that $H_t=\lambda_{t-1}^2H_{t-1}+1$,

$$\frac{1}{\eta_t}-\frac{\lambda_{t-1}}{\eta_{t-1}}=\frac{G}{D}\left(\sqrt{H_t}-\lambda_{t-1}\sqrt{H_{t-1}}\right)=\frac{G}{D}\left(\sqrt{\lambda_{t-1}^2H_{t-1}+1}-\sqrt{\lambda_{t-1}^2H_{t-1}}\right)\geq 0.$$

Therefore, we can apply the diameter condition $\|x_t-u\|\leq D$ (and the Lipschitzness $\|g_t\|\leq G$ as well), and use a telescopic sum again to obtain

$$\begin{aligned}
\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T},u) &\leq \frac{D^2}{2\eta_1}\left(\prod_{i=1}^{T-1}\lambda_i\right)+\frac{D^2}{2}\sum_{t=2}^{T}\left(\frac{1}{\eta_t}-\frac{\lambda_{t-1}}{\eta_{t-1}}\right)\left(\prod_{i=t}^{T-1}\lambda_i\right)+\frac{G^2}{2}\sum_{t=1}^{T}\eta_t\left(\prod_{i=t}^{T-1}\lambda_i\right)\\
&=\frac{D^2}{2\eta_T}+\frac{G^2}{2}\sum_{t=1}^{T}\eta_t\left(\prod_{i=t}^{T-1}\lambda_i\right)\\
&=\frac{1}{2}DG\sqrt{H_T}+\frac{1}{2}DG\sum_{t=1}^{T}H_t^{-1/2}\left(\prod_{i=t}^{T-1}\lambda_i\right).
\end{aligned}$$

To proceed, define

$$\mathrm{Sum}_T:=\sum_{t=1}^{T}H_t^{-1/2}\left(\prod_{i=t}^{T-1}\lambda_i\right).$$

We now show that $\mathrm{Sum}_T\leq 2\sqrt{H_T}$ by induction.

- When $T=1$, we have $H_1=1$ and $\mathrm{Sum}_1=H_1^{-1/2}=1$, therefore $\mathrm{Sum}_1\leq 2\sqrt{H_1}$.
- When $T>1$, starting from the induction hypothesis $\mathrm{Sum}_{T-1}\leq 2\sqrt{H_{T-1}}$,

  $$\mathrm{Sum}_T=H_T^{-1/2}+\lambda_{T-1}\mathrm{Sum}_{T-1}\leq H_T^{-1/2}+2\lambda_{T-1}\sqrt{H_{T-1}}.$$

  Applying $H_T=\lambda_{T-1}^2H_{T-1}+1$,

  $$\mathrm{Sum}_T\leq(\lambda_{T-1}^2H_{T-1}+1)^{-1/2}+2\sqrt{\lambda_{T-1}^2H_{T-1}}\leq 2\sqrt{\lambda_{T-1}^2H_{T-1}+1}=2\sqrt{H_T},$$

  where the inequality is due to the concavity of square root.

Combining everything above leads to the first part of the theorem.

As for the second part (time-invariant $\lambda<1$), we follow the same procedure until Eq.(12). The learning rate η_t is independent of t; denote it as $\eta=DG^{-1}\sqrt{1-\lambda^2}$. Then,

$$\begin{aligned}
\mathrm{R}_T^{\lambda}(l_{1:T},u) &\leq \frac{\lambda^{T-1}}{2\eta}\|x_1-u\|^2+\frac{1-\lambda}{2\eta}\sum_{t=2}^{T}\lambda^{T-t}\|x_t-u\|^2+\frac{\eta}{2}\sum_{t=1}^{T}\lambda^{T-t}\|g_t\|^2\\
&\leq \frac{D^2\lambda^{T-1}}{2\eta}+\frac{D^2(1-\lambda)}{2\eta}\sum_{t=2}^{T}\lambda^{T-t}+\frac{\eta G^2}{2}\sum_{t=1}^{T}\lambda^{T-t}\\
&=\frac{D^2\lambda^{T-1}}{2\eta}+\frac{D^2(1-\lambda)}{2\eta}\frac{1-\lambda^{T-1}}{1-\lambda}+\frac{\eta G^2}{2}\frac{1-\lambda^T}{1-\lambda}\\
&\leq \frac{DG}{2\sqrt{1-\lambda^2}}+\frac{DG\sqrt{1-\lambda^2}}{2(1-\lambda)}
\end{aligned}$$

$$\leq \frac{3}{2}\frac{DG}{\sqrt{1-\lambda^2}}. \qquad (1-\lambda^2 \leq 2(1-\lambda))$$

Following the definition of H_T, in the case of time-invariant $\lambda < 1$,

$$H_T = \frac{1-\lambda^{2T}}{1-\lambda^2}.$$

Due to Lemma B.1, if $T \geq \frac{1}{2}(1-\lambda)^{-1}$, we have $\lambda^{2T} \leq e^{-1}$, therefore

$$\frac{1}{\sqrt{1-\lambda^2}} \leq \sqrt{\frac{H_T}{1-e^{-1}}}.$$

Combining the above completes the proof. □

Accompanying the upper bounds, the following theorem characterizes an instance-dependent discounted regret lower bound.

Theorem 7. *Consider any combination of time horizon $T \in \mathbb{N}_+$, discount factors $\lambda_{1:T}$, Lipschitz constant $G > 0$, and variance budget $V \in \left(0, G^2 H_T\right]$, where H_T is defined in Eq.(2). Furthermore, consider any nonzero $u \in \mathcal{X}$ that satisfies $-u \in \mathcal{X}$. For any OCO algorithm that possibly depends on these quantities, there exists a sequence of linear losses $l_t(x) = \langle g_t, x\rangle$ such that $\|g_t\| \leq G$ for all $t \in [1:T]$,*

$$V = \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i^2\right)\|g_t\|^2,$$

and

$$\max\left[\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u), \mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, -u)\right] \geq \frac{1}{\sqrt{2}}\|u\|\sqrt{V}.$$

Proof of Theorem 7. The proof is a mild generalization of the undiscounted argument, e.g., (Orabona, 2023a, Theorem 5.1). We provide it here for completeness.

We first define a random sequence of loss gradients. Let $\varepsilon_1, \ldots, \varepsilon_T$ be a sequence of iid Rademacher random variables: ε_t equals ± 1 with probability $\frac{1}{2}$ each. Also, let

$$L = \sqrt{\frac{V}{\sum_{t=1}^{T}\prod_{i=t}^{T-1}\lambda_i^2}}.$$

Then, we define the loss gradient sequence $\tilde{g}_{1:T}$ as $\tilde{g}_t = L\varepsilon_t\frac{u}{\|u\|}$. Notice that $L \leq G$.

Now consider the regret with respect to $\tilde{l}_{1:T}$, the random linear losses induced by the random gradient sequence $\tilde{g}_{1:T}$.

$$\mathrm{R}_T^{\lambda_{1:T}}\left(\tilde{l}_{1:T}, u\right) = \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\langle\tilde{g}_t, x_t - u\rangle = \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\frac{L}{\|u\|}\langle u, x_t\rangle\varepsilon_t - \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)L\|u\|\varepsilon_t.$$

$$\mathrm{R}_T^{\lambda_{1:T}}\left(\tilde{l}_{1:T}, -u\right) = \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\langle\tilde{g}_t, x_t + u\rangle = \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\frac{L}{\|u\|}\langle u, x_t\rangle\varepsilon_t + \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)L\|u\|\varepsilon_t.$$

Therefore,

$$\begin{aligned}
&\mathbb{E}\left[\max\left[\mathrm{R}_T^{\lambda_{1:T}}(\tilde{l}_{1:T}, u), \mathrm{R}_T^{\lambda_{1:T}}(\tilde{l}_{1:T}, -u)\right]\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\frac{L}{\|u\|}\langle u, x_t\rangle\varepsilon_t\right] + \mathbb{E}\left[\max\left[-\sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)L\|u\|\varepsilon_t, \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)L\|u\|\varepsilon_t\right]\right] \\
&= \sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\frac{L}{\|u\|}\mathbb{E}\left[\langle u, x_t\rangle\varepsilon_t\right] + L\|u\|\,\mathbb{E}\left[\left|\sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\varepsilon_t\right|\right]
\end{aligned}$$

$$
\begin{aligned}
&= L\|u\|\,\mathbb{E}\left[\left|\sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i\right)\varepsilon_t\right|\right] \\
&\geq \frac{1}{\sqrt{2}}L\|u\|\sqrt{\sum_{t=1}^{T}\left(\prod_{i=t}^{T-1}\lambda_i^2\right)} && \text{(Khintchine inequality (Haagerup, 1981))} \\
&= \frac{1}{\sqrt{2}}\|u\|\sqrt{V}.
\end{aligned}
$$

Finally, note that the above lower-bounds the expectation. There exists a loss gradient sequence $g_{1:T}$ with $g_t \in \{Lu/\|u\|, -Lu/\|u\|\}$, such that $\max\left[\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T},u), \mathrm{R}_T^{\lambda_{1:T}}(l_{1:T},-u)\right] \geq \frac{1}{\sqrt{2}}\|u\|\sqrt{V}$. □

The next theorem, presented in Section 2.2, analyzes gradient adaptive OGD.

Theorem 2. *If the diameter of $\mathcal{X}$ is at most D, then OGD with learning rate $\eta_t = DV_t^{-1/2}$ guarantees for all $T \in \mathbb{N}_+$ and loss sequence $l_{1:T}$,*

$$\sup_u \mathrm{R}_T^{\lambda_{1:T}}(l_{1:T},u) \leq \frac{3}{2}D\sqrt{V_T}.$$

Proof of Theorem 2. As shown in (Orabona, 2023a, Chapter 4.2), applying OGD with the undiscounted gradient-dependent learning rate

$$\eta_t = \frac{D}{\sqrt{\sum_{i=1}^{t}\|\hat{g}_i\|^2}} \tag{13}$$

to surrogate linear losses $\langle \hat{g}_t, \cdot\rangle$ guarantees the undiscounted regret bound

$$\sum_{t=1}^{T}\langle \hat{g}_t, x_t - u\rangle \leq \frac{3}{2}D\sqrt{\sum_{t=1}^{T}\|\hat{g}_t\|^2}.$$

For the discounted setting, we follow the rescaling trick from Section 2.2. First, consider the effective prediction rule when $\hat{g}_t$ is defined according to Eq.(5).

$$
\begin{aligned}
x_{t+1} &= \Pi_{\mathcal{X}}\left(x_t - \frac{D}{\sqrt{\sum_{i=1}^{t}\|\hat{g}_i\|^2}}\hat{g}_t\right) \\
&= \Pi_{\mathcal{X}}\left[x_t - \frac{D}{\sqrt{\sum_{i=1}^{t}\left(\prod_{j=1}^{i-1}\lambda_j^{-2}\right)\|g_i\|^2}}\left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)g_t\right] \\
&= \Pi_{\mathcal{X}}\left[x_t - \frac{D}{\sqrt{\sum_{i=1}^{t}\left(\prod_{j=i}^{t-1}\lambda_j^{2}\right)\|g_i\|^2}}g_t\right],
\end{aligned}
$$

which is exactly the prediction rule this theorem proposes. As for the regret bound, we have

$$\sum_{t=1}^{T}\langle \hat{g}_t, x_t - u\rangle \leq \frac{3}{2}D\sqrt{\sum_{t=1}^{T}\|\hat{g}_t\|^2} = \frac{3}{2}D\sqrt{\sum_{t=1}^{T}\left(\prod_{i=1}^{t-1}\lambda_i^{-2}\right)\|g_t\|^2}.$$

Plugging it into Eq.(6) and using the definition of V_T from Eq.(3) complete the proof. □

Remark B.1 (Failure of OGD). *As mentioned in Section 2.2, one might suspect that OGD with an even better learning rate than Theorem 2 (i.e., possibly using the oracle knowledge of $\|u\|$) can further improve the regret bound to $O(\|u\|\sqrt{V_T})$, matching the lower bound (Theorem 7). We now explain where this intuition could come from, and why it is misleading.*

Consider the undiscounted setting ($\lambda_t = 1$). For any scaling factor α, it is well-known that OGD with learning rate $\eta = \alpha G^{-1} T^{-1/2}$ achieves the undiscounted regret bound $O\left((\alpha + \|u\|^2 \alpha^{-1}) G\sqrt{T}\right)$ (Orabona, 2023a, Chapter 2.1), which becomes $O(\|u\| G\sqrt{T})$ if the oracle tuning $\alpha = \|u\|$ is allowed. This might suggest that the remaining suboptimality of Theorem 2 can be closed by a similar oracle tuning, i.e., setting

$$\eta_t = \frac{\|u\|}{\sqrt{V_t}}. \tag{14}$$

However, such an analogy misses a key point: the aforementioned nice property of "α-scaled OGD" only holds with time-invariant learning rates, and therefore, it can be found that OGD with the time-varying learning rate Eq.(14) does not yield the $O(\|u\| G\sqrt{V_T})$ bound we aim for, let alone the impossibility of oracle tuning. Broadly speaking, it is known (but sometimes overlooked) that OGD has certain fundamental incompatibility with time-varying learning rates, but this can be fixed by an extra regularization centered at the origin (Orabona & Pál, 2018; Fang et al., 2022; Jacobsen & Cutkosky, 2022). Such a procedure encodes the initialization into the algorithm, which essentially makes it similar to FTRL.

B.2. Algorithm from (Zhang et al., 2024)

In this subsection we introduce the algorithm from (Zhang et al., 2024) and its guarantee. Proposed for the undiscounted setting ($\lambda_t = 1$), it achieves a scale-free regret bound that simultaneously adapts to both the loss sequence and the comparator.

Following the polar-decomposition technique from (Cutkosky & Orabona, 2018), the main component of this algorithm is a subroutine (a standalone one dimensional OCO algorithm) that operates on the positive real line $[0, \infty)$. We present the pseudocode of this subroutine as Algorithm 2. Technically it is a combination of (Zhang et al., 2024, Algorithm 1 + part of Algorithm 2) and (Cutkosky, 2019, Algorithm 2). Except the choice of the Φ function which uses an unusual continuous-time analysis, everything else is somewhat standard in the community. Here is an overview of the idea.

- The core component is the *potential function* $\Phi_{h,\varepsilon}(v, s)$ that takes two input arguments, the *observed gradient variance* v and the *observed gradient sum* s. For now let us briefly suppress the dependence on auxiliary parameters h and ε. At the beginning of the t-th round, with the observations of past gradients $g_{1:t-1}$, we "ideally" (there are two twists explained later) would like to define its summaries (sometimes called *sufficient statistics*)

$$v_t = \sum_{i=1}^{t-1} g_i^2, \quad s_t = -\sum_{i=1}^{t-1} g_i,$$

  and predict $x_t = \partial_2 \Phi(v_t, s_t)$, i.e., the partial derivative of the potential function Φ with respect to its second argument, evaluated at the pair (v_t, s_t). This is the standard procedure of the "potential method" (Cesa-Bianchi & Lugosi, 2006; McMahan & Orabona, 2014; Mhammedi & Koolen, 2020) in online learning, which is associated to a well-established analytical strategy (McMahan & Orabona, 2014). Equivalently, one may also interpret this procedure as *Follow the Regularized Leader* (FTRL) (Orabona, 2023a, Chapter 7.3) with linearized losses: the potential function Φ is essentially the convex conjugate of a FTRL regularizer (Zhang et al., 2022, Section 3.1).

  A bit more on the design of Φ: the intuition is that h is typically much smaller than v and s, so if we set $h = 0$ (rigorously this is not allowed since the prediction would be a bit too aggressive; but just for our intuitive discussion this is fine), then the potential function is morally

$$\Phi(v, s) \approx \varepsilon\sqrt{v}\left(2\int_0^{\frac{s}{2\sqrt{v}}} \operatorname{erfi}(u)du - 1\right),$$

  which is associated to the prediction function

$$\partial_2 \Phi(v, s) \approx \varepsilon \cdot \operatorname{erfi}\left(\frac{s}{2\sqrt{v}}\right) = \varepsilon \int_0^{s/\sqrt{4v}} \exp(u^2)du.$$

Algorithm 2 ((Zhang et al., 2024, Algorithm 1 and 2) + (Cutkosky, 2019, Algorithm 2)) Undiscounted 1D magnitude learner on $[0, \infty)$.

Require: Hyperparameter $\varepsilon > 0$ (default $\varepsilon = 1$).

1: Initialize parameters $v_1 = 0$, $s_1 = 0$, $h_1 = 0$. Define the following (h, ε)-parameterized *potential function* $\Phi_{h,\varepsilon}(v, s)$ with two input arguments v and s.

$$\Phi_{h,\varepsilon}(v,s) := \varepsilon\sqrt{v+2hs+16h^2}\left(2\int_0^{\frac{s}{2\sqrt{v+2hs+16h^2}}}\operatorname{erfi}(u)du - 1\right).$$

2: **for** $t = 1, 2, \ldots$ **do**

3: If $h_t = 0$, define an unprojected prediction $\tilde{x}_t = 0$; otherwise, define

$$\begin{aligned}\tilde{x}_t &= \partial_2\Phi_{h_t,\varepsilon}(v_t, s_t)\\ &= \varepsilon\cdot\operatorname{erfi}\left(\frac{s_t}{2\sqrt{v_t+2h_ts_t+16h_t^2}}\right) - \frac{\varepsilon h_t}{\sqrt{v_t+2h_ts_t+16h_t^2}}\exp\left[\frac{s_t^2}{4(v_t+2h_ts_t+16h_t^2)}\right].\end{aligned}$$

4: Predict $x_t = \Pi_{[0,\infty)}(\tilde{x}_t)$, the projection of $\tilde{x}_t$ to the domain $[0, \infty)$.

5: Receive the loss gradient $g_t \in \mathbb{R}$.

6: Clip the gradient by defining $g_{t,\text{clip}} = \Pi_{[-h_t,h_t]}(g_t)$, and let $h_{t+1} = \max(h_t, |g_t|)$.

7: Define a surrogate loss gradient $\tilde{g}_{t,\text{clip}} \in \mathbb{R}$, where

$$\tilde{g}_{t,\text{clip}} = \begin{cases} g_{t,\text{clip}}, & g_{t,\text{clip}}\tilde{x}_t \geq g_{t,\text{clip}}x_t,\\ 0, & \text{else.}\end{cases}$$

8: Update $v_{t+1} = v_t + \tilde{g}_{t,\text{clip}}^2$, $s_{t+1} = s_t - \tilde{g}_{t,\text{clip}}$.

9: **end for**

The use of the erfi function might seem obscure, but essentially it is rooted in a recent trend (Drenska & Kohn, 2020; Zhang et al., 2022; Harvey et al., 2023) connecting online learning to stochastic calculus: we scale the OCO game towards its continuous-time limit and solve the obtained *Backward Heat Equation*. Prior to this trend, the predominant idea was using (McMahan & Orabona, 2014; Orabona & Pál, 2016; Mhammedi & Koolen, 2020)

$$\Phi(v,s) \approx \frac{\varepsilon}{\sqrt{v}}\exp\left(\frac{s^2}{\text{constant}\cdot v}\right), \tag{15}$$

$$\partial_2\Phi(v,s) \approx \varepsilon\frac{\text{constant}\cdot s}{v^{3/2}}\exp\left(\frac{s^2}{\text{constant}\cdot v}\right).$$

It can be shown that the erfi prediction rule is quantitatively stronger, and quite importantly, it makes the hyperparameter ε "unitless" (Zhang et al., 2024, Section 5). In contrast, in the more classical potential function Eq.(15), ε carries the unit of "gradient squared", which means the algorithm requires a guess of the *time-uniform Lipschitz constant* $\max_t \|g_t\|$ at the *beginning of the game*. Due to the discussion in Section 2.2 (right after introducing the rescaling trick), this suffers from certain suboptimality (Remark B.4) when applied with rescaling.

- Although most of the heavy lifting is handled by the choice of Φ, a remaining issue is that with $h \neq 0$, the prediction $x_t = \partial_2\Phi(v_t, s_t)$ is not necessarily positive, which violates the domain constraint $[0, \infty)$. To fix this issue, we adopt the technique from (Cutkosky & Orabona, 2018; Cutkosky, 2020) which has become a standard tool for the community: predict $x_t = \Pi_{[0,\infty)}(\tilde{x}_t)$ which is the projection of $\tilde{x}_t = \partial_2\Phi(v_t, s_t)$ to the domain $[0, \infty)$, and update the sufficient statistics (v_{t+1}, s_{t+1}) using a *surrogate loss gradient* $\tilde{g}_{t,\text{clip}}$ (Line 7 of Algorithm 2) instead of $g_{t,\text{clip}}$ (the clipping will be explained later; for now we might regard $g_{t,\text{clip}} = g_t$, the actual gradient).

  The intuition behind Line 7 is that, if the unprojected prediction $\tilde{x}_t = \partial_2\Phi(v_t, s_t)$ is already negative and the actual loss gradient $g_{t,\text{clip}}$ encourages it to be "more negative", then we set $\tilde{g}_{t,\text{clip}} = 0$ in the update of the (v_{t+1}, s_{t+1}) pair to

avoid this undesirable behavior (unprojected prediction drifting away from the domain). In other situations, it is fine to use $g_{t,\mathrm{clip}}$ directly in the update of (v_{t+1}, s_{t+1}), therefore we simply set $\tilde{g}_{t,\mathrm{clip}} = g_{t,\mathrm{clip}}$.

Rigorously, (Cutkosky, 2020, Theorem 2) shows that for all comparator $u \in [0, \infty)$, $\langle g_{t,\mathrm{clip}}, x_t - u\rangle \le \langle \tilde{g}_{t,\mathrm{clip}}, \tilde{x}_t - u\rangle$. That is, as long as the unprojected prediction sequence $\tilde{x}_{1:T}$ guarantees a good regret bound (in an improper manner, i.e., violating the domain constraint), then the projected prediction sequence $x_{1:T}$ also guarantees a good regret bound, but properly.

- Another twist is related to *updating* the auxiliary parameter h, which was previously ignored. Due to the typical limitation of FTRL algorithms, we have to guess the range of g_t (i.e., a time-varying Lipschitz constant G_t such that $\|g_t\| \le G_t$) *right before* making the prediction x_t, and h_t serves as this guess. (Cutkosky, 2019) suggests using the range of past loss gradients $h_t = \max_{i\in[1:t-1]} |g_i|$ to guess that $|g_t| \le h_t$. Surely this could be wrong: in that case ($|g_t| > h_t$), we clip g_t to $[-h_t, h_t]$ before sending it to the (v_{t+1}, s_{t+1}) update.

  We also clarify a possible confusion related to "guessing the Lipschitzness". We have argued that if an algorithm requires guessing the time-uniform Lipschitz constant $\max_t \|g_t\|$ at the beginning of the game, then it does not serve as a good base algorithm $\mathcal{A}$ in our rescaling trick. The use of h_t is different: it is updated online as a guess of $\|g_t\|$, which is fine (to apply with rescaling).

In terms of the concrete guarantee, the following theorem characterizes the undiscounted regret bound of Algorithm 2. The proof is a straightforward corollary of (Zhang et al., 2024, Lemma B.2) and (Cutkosky, 2020, Theorem 2), therefore omitted.

Theorem 8 ((Cutkosky, 2020; Zhang et al., 2024))**.** *Algorithm 2 guarantees for all time horizon $T \in \mathbb{N}_+$, loss gradients $g_{1:T}$ and comparator $u \in [0, \infty)$,*

$$\sum_{t=1}^{T} g_t(x_t - u) \le \varepsilon\sqrt{\sum_{t=1}^{T} g_t^2 + 2GS + 16G^2} + uS + \sum_{t=1}^{T} |g_t - g_{t,\mathrm{clip}}|\,|x_t - u|\,,$$

where $G = \max_{t\in[1:T]} |g_t|$ and

$$S = 8G\left(1 + \sqrt{\log(2u\varepsilon^{-1} + 1)}\right)^2 + 2\sqrt{\sum_{t=1}^{T} g_t^2 + 16G^2}\left(1 + \sqrt{\log(2u\varepsilon^{-1} + 1)}\right).$$

Remark B.2 (Iterate stability)**.** *The above bound has an iterate stability term $\sum_{t=1}^{T} |g_t - g_{t,\mathrm{clip}}|\,|x_t - u|$, which can be further upper-bounded by $(u + \max_{t\in[1:T]} x_t)G_T$. Similar characterizations of stability have appeared broadly in stochastic optimization before (Orabona & Pál, 2021; Ivgi et al., 2023; Orabona, 2023b). For the general adversarial setting we consider, (Cutkosky, 2019) suggests using* artificial constraints *to trade $\max_t x_t$ for terms that only depend on $g_{1:T}$ and u. We do not take this route because (i) it makes our discounted algorithm more complicated but does not seem to improve the performance; and (ii) even without artificial constraints, the prediction magnitude is indeed controlled in our main application (online conformal prediction), and possibly in the downstream stochastic setting as well (following (Orabona, 2023b)).*

Given the above undiscounted 1D subroutine, we can extend it to $\mathbb{R}^d$ following the standard polar-decomposition technique (Cutkosky & Orabona, 2018). Overall, the algorithm becomes the $\lambda_t = 1$ special case of Algorithm 3 (our main algorithm presented in Section B.3). This is as expected, since the discounted setting is a strict generalization. The following theorem is essentially (Zhang et al., 2024, Theorem 2 + the discussion after that).

Theorem 9 ((Zhang et al., 2024))**.** *With $\lambda_t = 1$ for all t, Algorithm 3 from Section B.3 guarantees for all $T \in \mathbb{N}_+$, loss gradients $g_{1:T}$ and comparator $u \in \mathbb{R}^d$,*

$$\sum_{t=1}^{T} \langle g_t, x_t - u\rangle \le O\left(\|u\|\sqrt{V_T \log(\|u\|\,\varepsilon^{-1})} \vee \|u\|\,G_T \log(\|u\|\,\varepsilon^{-1})\right) + \sum_{t=1}^{T} \|g_t - g_{t,\mathrm{clip}}\|\,\|x_t - u\|\,,$$

where V_T and G_T are defined in Eq.(3), and $O(\cdot)$ is in the regime of large V_T ($V_T \gg G_T$) and large $\|u\|$ ($\|u\| \gg \varepsilon$). Furthermore, if the comparator $u = 0$, we have

$$\sum_{t=1}^{T} \langle g_t, x_t\rangle \le O\left(\varepsilon\sqrt{V_T}\right) + \sum_{t=1}^{T} \|g_t - g_{t,\mathrm{clip}}\|\,\|x_t\|\,.$$

B.3. Analysis of the main algorithm

This subsection presents our main discounted algorithm and its regret bound. We start from its 1D magnitude learner.

Theorem 3. *Given any hyperparameter $\varepsilon > 0$, Algorithm 1 guarantees for all time horizon $T \in \mathbb{N}_+$, loss sequence $l_{1:T}$, comparator $u \in [0, \infty)$ and stability window length $\tau \in [1:T]$,*

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T}, u) \leq \varepsilon\sqrt{V_T + 2G_T S + 16G_T^2} + u\left(S + G_T\right) + \left(\max_{t\in[T-\tau+1:T]} x_t\right) G_T + \left(\prod_{t=T-\tau}^{T-1} \lambda_t\right)\left(\max_{t\in[1:T-\tau]} x_t\right) G_{T-\tau},$$

where

$$S = 8G_T\left(1 + \sqrt{\log(2u\varepsilon^{-1}+1)}\right)^2 + 2\sqrt{V_T + 16G_T^2}\left(1 + \sqrt{\log(2u\varepsilon^{-1}+1)}\right).$$

Proof of Theorem 3. The proof mostly follows from carefully checking the equivalence of the following two algorithms: (i) Algorithm 1 (the discounted algorithm) on a sequence of loss gradients $g_{1:T}$, and (ii) Algorithm 2 (the undiscounted algorithm) on the sequence of scaled surrogate gradients, $\left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right) g_t; \forall t \in [1:T]$. Since the quantities in Algorithm 1 and 2 follow the same notation, we separate them by adding a superscript D on quantities in Algorithm 1, and ND in their Algorithm 2 counterparts.

We show this by induction: suppose for some $t \in [1:T]$, we have

$$s_t^D = \left(\prod_{i=1}^{t-2}\lambda_i\right) s_t^{ND}, \quad v_t^D = \left(\prod_{i=1}^{t-2}\lambda_i^2\right) v_t^{ND}, \quad h_t^D = \left(\prod_{i=1}^{t-2}\lambda_i\right) h_t^{ND}.$$

Such an induction hypothesis holds for $t = 1$. Then, from the prediction rules (and the fact that all the discount factors are strictly positive), the unprojected predictions $\tilde{x}_t^D = \tilde{x}_t^{ND}$, and the projected predictions $x_t^D = x_t^{ND}$.

Now consider the gradient clipping. Since $g_t^{ND} = \left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right) g_t^D$,

$$\begin{aligned}
g_{t,\mathrm{clip}}^{ND} &= \Pi_{[-h_t^{ND}, h_t^{ND}]}\left(g_t^{ND}\right) \\
&= c^{-1}\Pi_{[-ch_t^{ND}, ch_t^{ND}]}\left(cg_t^{ND}\right) && (\forall c > 0) \\
&= \left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)\Pi_{[-\lambda_{t-1}h_t^D, \lambda_{t-1}h_t^D]}\left(g_t^D\right) && (c = \textstyle\prod_{i=1}^{t-1}\lambda_i) \\
&= \left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right) g_{t,\mathrm{clip}}^D.
\end{aligned}$$

Then, due to $x_t^D = x_t^{ND}$ and $\tilde{x}_t^D = \tilde{x}_t^{ND}$, we have $\tilde{g}_{t,\mathrm{clip}}^{ND} = \left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)\tilde{g}_{t,\mathrm{clip}}^D$.

Finally, consider the updates of s_{t+1}^D, v_{t+1}^D and h_{t+1}^D, as well as their Algorithm 2 counterparts.

$$s_{t+1}^D = \lambda_{t-1}s_t^D - \tilde{g}_{t,\mathrm{clip}}^D = \left(\prod_{i=1}^{t-1}\lambda_i\right) s_t^{ND} - \left(\prod_{i=1}^{t-1}\lambda_i\right)\tilde{g}_{t,\mathrm{clip}}^{ND} = \left(\prod_{i=1}^{t-1}\lambda_i\right)\left(s_t^{ND} - \tilde{g}_{t,\mathrm{clip}}^{ND}\right) = \left(\prod_{i=1}^{t-1}\lambda_i\right) s_{t+1}^{ND}.$$

Similarly,

$$v_{t+1}^D = \left(\prod_{i=1}^{t-1}\lambda_i^2\right) v_{t+1}^{ND},$$

$$h_{t+1}^D = \max\left(\lambda_{t-1}h_t^D, \left|g_t^D\right|\right) = \max\left[\left(\prod_{i=1}^{t-1}\lambda_i\right) h_t^{ND}, \left(\prod_{i=1}^{t-1}\lambda_i\right)\left|g_t^{ND}\right|\right] = \left(\prod_{i=1}^{t-1}\lambda_i\right) h_{t+1}^{ND}.$$

That is, the induction hypothesis holds for $t+1$, and therefore we have shown the equivalence of the considered two algorithms.

As for the regret bound of Algorithm 1, combining the reduction Eq.(6) and the regret bound of Algorithm 2 (Theorem 8 from Appendix B.2) immediately gives us

$$\mathrm{R}_T^{\lambda_{1:T}}(l_{1:T},u) \le \varepsilon\sqrt{V_T+2G_TS+16G_T^2}+uS+\left(\prod_{t=1}^{T-1}\lambda_t\right)\sum_{t=1}^{T}\left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)|g_t-g_{t,\mathrm{clip}}|\,|x_t-u|\,,$$

where

$$S=8G_T\left(1+\sqrt{\log(2u\varepsilon^{-1}+1)}\right)^2+2\sqrt{V_T+16G_T^2}\left(1+\sqrt{\log(2u\varepsilon^{-1}+1)}\right).$$

The remaining clipping error term can be bounded similarly as (Cutkosky, 2019, Theorem 2). For any $\tau\in[1:T]$,

$$\begin{aligned}
\sum_{t=1}^{T-\tau}\left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)|g_t-g_{t,\mathrm{clip}}|\,|x_t-u| &\le \max_{t\in[1:T-\tau]}|x_t-u|\sum_{t=1}^{T-\tau}\left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)|g_t-g_{t,\mathrm{clip}}|\\
&= \max_{t\in[1:T-\tau]}|x_t-u|\sum_{t=1}^{T-\tau}\left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)(h_{t+1}-\lambda_{t-1}h_t) \qquad \text{(from Algorithm 1)}\\
&= \max_{t\in[1:T-\tau]}|x_t-u|\sum_{t=1}^{T-\tau}\left[\left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)h_{t+1}-\left(\prod_{i=1}^{t-2}\lambda_i^{-1}\right)h_t\right]\\
&= \max_{t\in[1:T-\tau]}|x_t-u|\left[\left(\prod_{i=1}^{T-\tau-1}\lambda_i^{-1}\right)h_{T-\tau+1}-h_1\right]\\
&\le \left(\max_{t\in[1:T-\tau]}x_t+u\right)\left(\prod_{i=1}^{T-\tau-1}\lambda_i^{-1}\right)h_{T-\tau+1}.
\end{aligned}$$

Similarly,

$$\sum_{t=T-\tau+1}^{T}\left(\prod_{i=1}^{t-1}\lambda_i^{-1}\right)|g_t-g_{t,\mathrm{clip}}|\,|x_t-u| \le \left(\max_{t\in[T-\tau+1:T]}x_t+u\right)\left[\left(\prod_{i=1}^{T-1}\lambda_i^{-1}\right)h_{T+1}-\left(\prod_{i=1}^{T-\tau-1}\lambda_i^{-1}\right)h_{T-\tau+1}\right].$$

Finally, multiplying $\prod_{t=1}^{T-1}\lambda_t$ and using $G_T=h_{T+1}$ and $G_{T-\tau}=h_{T-\tau+1}$ complete the proof. □

As for the extension to $\mathbb{R}^d$ following (Cutkosky & Orabona, 2018), we present the pseudocode as Algorithm 3. There is a small twist: when applying Algorithm 1 as the 1D subroutine, in its Line 6 we set $g_{t,\mathrm{clip}}=g_t$, and h_{t+1} is given by the meta-algorithm. That is, the gradient clipping is handled on the high level (Algorithm 3) rather than the low level (Algorithm 1).

Algorithm 3 Discounted adaptivity on $\mathbb{R}^d$.

1: Define $\mathcal{A}_{1d}$ as a minor variant of Algorithm 1 (with hyperparameter ε), where its Line 6 is replaced by: "Set $g_{t,\mathrm{clip}}=g_t$, and receive a hint h_{t+1}."
2: Define $\mathcal{A}_{\mathsf{B}}$ as the algorithm from Theorem 2, on the d-dimensional unit L_2 norm ball (with $D=2$).
3: Initialize $h_1=0$.
4: **for** $t=1,2,\dots$ **do**
5: Query $\mathcal{A}_{1d}$ for its prediction $y_t\in\mathbb{R}$.
6: Query $\mathcal{A}_{\mathsf{B}}$ for its prediction $w_t\in\mathbb{R}^d$; $\|w_t\|\le 1$.
7: Predict $x_t=w_ty_t$, receive the loss gradient $g_t\in\mathbb{R}^d$ and the discount factor $\lambda_t\in(0,\infty)$.
8: Update $h_{t+1}=\max\left(\lambda_{t-1}h_t,\|g_t\|\right)$, and define $g_{t,\mathrm{clip}}=g_t\lambda_{t-1}h_t/h_{t+1}$ (if $h_{t+1}=0$ then $g_{t,\mathrm{clip}}=0$).
9: Send $\langle g_{t,\mathrm{clip}},w_t\rangle$ and $g_{t,\mathrm{clip}}$ as the surrogate loss gradients to $\mathcal{A}_{1d}$ and $\mathcal{A}_{\mathsf{B}}$, respectively.
10: Send the discount factor λ_t to $\mathcal{A}_{1d}$ and $\mathcal{A}_{\mathsf{B}}$.
11: Send the hint h_{t+1} to $\mathcal{A}_{1d}$.
12: **end for**

Algorithm 3 induces our main theorem (Theorem 4). The proof combines the undiscounted regret bound (Theorem 9) and our rescaling trick. It is almost the same as the above proof of Theorem 3, therefore omitted.

Remark B.3 (Importance of forgetting)**.** *Different from the undiscounted algorithm in (Zhang et al., 2024), the above discounted generalization rapidly forgets the past observations (which could be misleading for the future). Such an idea might even be useful in stochastic convex optimization as well, in particular regarding functions with spatially inhomogeneous Lipschitzness. An example is the quadratic function, which is both strongly convex and smooth. Globally it is not Lipschitz, but near its minimizer the Lipschitz constant is small. When optimizing such a function, if an optimization algorithm internally estimates the Lipschitz constant from its historical observations, then this estimate could be overly conservative as the iterates move closer to the minimizer. An intuitive solution is to gradually forget the past, just like the idea of discounted online learning algorithms. Rigorously characterizing this intuition could be an exciting direction, which we defer to future works.*

Remark B.4 (Benefit of (Zhang et al., 2024))**.** *We used (Zhang et al., 2024) as the base algorithm in our scaling trick, but there are other options (Mhammedi & Koolen, 2020; Jacobsen & Cutkosky, 2022). The problem of such alternatives is that, they still need an estimate of the time-uniform Lipschitz constant at the beginning of the game, due to certain unit inconsistency (discussed in Appendix B.2, e.g., Eq.(15)). In the undiscounted setting, they guarantee*

$$\sum_{t=1}^{T} \langle g_t, x_t - u \rangle \le O\left(\|u\| \sqrt{V_T \log(\|u\| T)} \vee \|u\| G_T \log(\|u\| T) \right) + \sum_{t=1}^{T} \|g_t - g_{t,\mathrm{clip}}\| \, \|x_t - u\| ,$$

as opposed to Theorem 9 in this paper. After the scaling trick, the T dependence in this bound will be transferred to the obtained discounted regret bound, such that the latter also depends on the end time T. In other words, such a discounted regret bound is not "lifetime consistent".

C. Detail of Section 3

This section presents omitted details of our OCP application. First, we present the pseudocode of $\mathcal{A}_{CP}$ in Appendix C.1, with the notations (relevant quantities are with the superscipt $*$) from the OCP setting. All the concrete proofs are presented in Appendix C.2.

C.1. Pseudocode of the OCP algorithm

The pseudocode of $\mathcal{A}_{CP}$ is Algorithm 4. This is equivalent to directly applying Algorithm 1, our main 1D OCO algorithm, to the setting of Section 3.

In particular, the problem structure of OCP allows removing the surrogate loss construction (Line 7 of Algorithm 1), which makes the algorithm slightly simpler. To see this, notice that the surrogate loss $\tilde{g}_{t,\mathrm{clip}}$ there is only needed (i.e., does not equal $g_{t,\mathrm{clip}}$) when the unprojected prediction $\tilde{x}_t < 0$ and the projected prediction $x_t = 0$. In the OCP notation, this means that our radius prediction $r_t = 0 \le r_t^*$, therefore the subgradient g_t^* evaluated at r_t satisfies $g_t^* \le 0$. Back to the notation of Algorithm 1, we have $g_t \le 0$, therefore after clipping $g_{t,\mathrm{clip}} \le 0$. Putting things together,

$$g_{t,\mathrm{clip}} \left(\tilde{x}_t - x_t \right) \ge 0.$$

That is, the condition in Algorithm 1 that triggers $\tilde{g}_{t,\mathrm{clip}} = 0$ is impossible, therefore $\tilde{g}_{t,\mathrm{clip}}$ always equals $g_{t,\mathrm{clip}}$.

C.2. Omitted proofs

Moving to the analysis, we first prove the key lemma connecting the prediction magnitude of $\mathcal{A}_{CP}$ to the coverage metric S_t^*, Eq.(11). This is divided into two steps.

- First (Lemma C.1), we approximate the prediction rule of $\mathcal{A}_{CP}$, such that r_{t+1} characterizes the discounted sum of *clipped gradients*

$$S_{t,\mathrm{clip}}^* = -\sum_{i=1}^{t} \left(\prod_{j=i}^{t-1} \lambda_j \right) g_{i,\mathrm{clip}}^*, \tag{16}$$

  which is an internal quantity of Algorithm 4.

- The second step (Lemma C.2) is connecting $S_{t,\mathrm{clip}}^*$ to the unclipped version S_t^*, Eq.(11). This is the version presented in the main paper.

Algorithm 4 The proposed OCP algorithm $\mathcal{A}_{CP}$.

Require: Hyperparameter $\varepsilon > 0$ (default $\varepsilon = 1$).

1: Initialize $S^*_{0,\text{clip}} = 0$, $V^*_{0,\text{clip}} = 0$, $G^*_0 = 0$.

2: **for** $t = 1, 2, \ldots$ **do**

3: If $G^*_{t-1} = 0$, define the unprojected prediction $\tilde{r}_t = 0$. Otherwise,

$$\tilde{r}_t = \varepsilon \cdot \operatorname{erfi}\left(\frac{S^*_{t-1,\text{clip}}}{2\sqrt{V^*_{t-1,\text{clip}} + 2G^*_{t-1}S^*_{t-1,\text{clip}} + 16\left(G^*_{t-1}\right)^2}}\right) - \frac{\varepsilon G^*_{t-1}}{\sqrt{V^*_{t-1,\text{clip}} + 2G^*_{t-1}S^*_{t-1,\text{clip}} + 16\left(G^*_{t-1}\right)^2}} \exp\left[\frac{\left(S^*_{t-1,\text{clip}}\right)^2}{4\left(V^*_{t-1,\text{clip}} + 2G^*_{t-1}S^*_{t-1,\text{clip}} + 16\left(G^*_{t-1}\right)^2\right)}\right].$$

4: Predict $r_t = \Pi_{[0,\infty)}\left(\tilde{r}_t\right)$.

5: Receive the OCP loss gradient $g^*_t \in \mathbb{R}$ and the discount factor $\lambda_{t-1} \in (0, \infty)$.

6: Clip g^*_t by defining

$$g^*_{t,\text{clip}} = \Pi_{[-\lambda_{t-1}G^*_{t-1}, \lambda_{t-1}G^*_{t-1}]}\left(g^*_t\right).$$

7: Compute running statistics of past observations,

$$S^*_{t,\text{clip}} = -\sum_{i=1}^{t}\left(\prod_{j=i}^{t-1}\lambda_j\right)g^*_{i,\text{clip}}, \quad V^*_{t,\text{clip}} = \sum_{i=1}^{t}\left(\prod_{j=i}^{t-1}\lambda_j^2\right)\left\|g^*_{i,\text{clip}}\right\|^2, \quad G^*_t = \max_{i\in[1:t]}\left(\prod_{j=i}^{t-1}\lambda_j\right)\left\|g^*_i\right\|.$$

8: **end for**

Lemma C.1. *Consider $S^*_{t,\text{clip}}$ from Eq.(16). $\mathcal{A}_{CP}$ (Algorithm 4) guarantees for all t,*

$$\left|S^*_{t,\text{clip}}\right| \le 2\sqrt{V^*_{t,\text{clip}}}\left(1 + \sqrt{\log\left(1 + 2r_{t+1}\varepsilon^{-1}\right)}\right) + 13G^*_t\left(1 + \sqrt{\log\left(1 + 2r_{t+1}\varepsilon^{-1}\right)}\right)^2,$$

where $V^*_{t,\text{clip}}$ and G^*_t are internal quantities of Algorithm 4.

Proof of Lemma C.1. Throughout this proof we will consider the internal variables of Algorithm 4. The superscript $*$ are always removed to simplify the notation. Assume without loss of generality that internally in Algorithm 4, $G_t \neq 0$ for the considered t. Otherwise, all the gradients $g_1, \ldots, g_t$ are zero, which makes the statement of the lemma obvious.

To upper-bound $S_{t,\text{clip}}$, the analysis is somewhat similar to the control of Fenchel conjugate in (Zhang et al., 2024, Lemma B.1), although the latter serves a different purpose. Consider the input argument of the erfi function in the definition of r_{t+1}. There are two cases.

- **Case 1:** $S_{t,\text{clip}} < 2\sqrt{V_{t,\text{clip}} + 2G_tS_{t,\text{clip}} + 16G_t^2}$**.**

  With straightforward algebra,

$$S^2_{t,\text{clip}} - 8G_tS_{t,\text{clip}} - 4\left(V_{t,\text{clip}} + 16G_t^2\right) < 0,$$

$$S_{t,\text{clip}} \le 4G_t + \sqrt{16G_t^2 + 4\left(V_{t,\text{clip}} + 16G_t^2\right)} \le 2\sqrt{V_{t,\text{clip}}} + 13G_t.$$

- **Case 2:** $S_{t,\text{clip}} \ge 2\sqrt{V_{t,\text{clip}} + 2G_tS_{t,\text{clip}} + 16G_t^2}$**.**

  Since $G_t \neq 0$, Algorithm 4 predicts $r_{t+1} = \Pi_{[0,\infty)}(\tilde{r}_{t+1})$, where

$$\tilde{r}_{t+1} = \varepsilon \cdot \operatorname{erfi}\left(\frac{S_{t,\text{clip}}}{2\sqrt{V_{t,\text{clip}} + 2G_tS_{t,\text{clip}} + 16G_t^2}}\right)$$

$$- \frac{\varepsilon G_t}{\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}} \exp\left[\frac{S_{t,\text{clip}}^2}{4(V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2)}\right].$$

Due to a lower estimate of the erfi function (Zhang et al., 2024, Lemma A.3), for all $x \geq 1$, $\text{erfi}(x) \geq \exp(x^2)/2x$. Then,

$$\text{erfi}\left(\frac{S_{t,\text{clip}}}{2\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}}\right) \geq \frac{\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}}{S_{t,\text{clip}}} \exp\left[\frac{S_{t,\text{clip}}^2}{4(V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2)}\right],$$

and simple algebra characterizes the multiplier on the RHS,

$$\frac{\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}}{S_{t,\text{clip}}} - \frac{2G_t}{\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}} = \frac{V_{t,\text{clip}} + 16G_t^2}{S_{t,\text{clip}}\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}} \geq 0.$$

Therefore,

$$\begin{aligned}
r_{t+1} &\geq \tilde{r}_{t+1} \\
&\geq \frac{\varepsilon}{2}\text{erfi}\left(\frac{S_{t,\text{clip}}}{2\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}}\right) + \varepsilon \exp\left[\frac{S_{t,\text{clip}}^2}{4(V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2)}\right] \\
&\quad \times \left(\frac{\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}}{2S_{t,\text{clip}}} - \frac{G_t}{\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}}\right) \\
&\geq \frac{\varepsilon}{2}\text{erfi}\left(\frac{S_{t,\text{clip}}}{2\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}}\right).
\end{aligned}$$

Due to another estimate of erfi^{-1} (Zhang et al., 2024, Lemma A.4), for all $x \geq 0$ we have $\text{erfi}^{-1}(x) \leq 1 + \sqrt{\log(x+1)}$. Then,

$$\begin{aligned}
S_{t,\text{clip}} &\leq 2\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2} \cdot \text{erfi}^{-1}\left(2r_{t+1}\varepsilon^{-1}\right) \\
&\leq 2\sqrt{V_{t,\text{clip}} + 2G_t S_{t,\text{clip}} + 16G_t^2}\left(1 + \sqrt{\log\left(1 + 2r_{t+1}\varepsilon^{-1}\right)}\right).
\end{aligned}$$

Similar to the algebra of Case 1,

$$S_{t,\text{clip}} \leq 2\sqrt{V_{t,\text{clip}}}\left(1 + \sqrt{\log\left(1 + 2r_{t+1}\varepsilon^{-1}\right)}\right) + 13G_t\left(1 + \sqrt{\log\left(1 + 2r_{t+1}\varepsilon^{-1}\right)}\right)^2.$$

Combining the two cases, we have

$$S_{t,\text{clip}} \leq 2\sqrt{V_{t,\text{clip}}}\left(1 + \sqrt{\log\left(1 + 2r_{t+1}\varepsilon^{-1}\right)}\right) + 13G_t\left(1 + \sqrt{\log\left(1 + 2r_{t+1}\varepsilon^{-1}\right)}\right)^2.$$

That is, $S_{t,\text{clip}}$ is now bounded from the above by $\tilde{O}(\sqrt{V_{t,\text{clip}}})$.

On the other side, we now consider bounding $S_{t,\text{clip}}$ from below. This is a classical induction argument similar to (Zhang et al., 2024, Lemma 4.1). Consider any time index $i \in [1:t]$,

- If $S_{i-1,\text{clip}} \leq 0$, then according to the prediction rule of Algorithm 4, we have $r_i = 0$ regardless of the value of G_{i-1}. Then, due to the structure of OCP, we have $g_i \leq 0$, thus $g_{i,\text{clip}} \leq 0$ and $S_{i,\text{clip}} = \lambda_{i-1}S_{i-1,\text{clip}} - g_{i,\text{clip}} \geq \lambda_{i-1}S_{i-1,\text{clip}}$.
- If $S_{i-1,\text{clip}} > 0$, then $S_{i,\text{clip}} \geq \lambda_{i-1}S_{i-1,\text{clip}} - |g_{i,\text{clip}}| \geq -G_i$.

Combining these two and using an induction from $S_{0,\text{clip}} = 0$, we have $S_{t,\text{clip}} \geq -G_t$.

Summarizing the above completes the proof. □

The following lemma is the full version of Lemma 3.1 presented in Section 3; we further use $V^*_{t,\text{clip}} \le V^*_t$ there.

Lemma C.2. *$\mathcal{A}_{CP}$ guarantees for all t,*

$$|S^*_t| \le 2\sqrt{V^*_{t,\text{clip}}}\left(1+\sqrt{\log\left(1+2r_{t+1}\varepsilon^{-1}\right)}\right) + 14G^*_t\left(1+\sqrt{\log\left(1+2r_{t+1}\varepsilon^{-1}\right)}\right)^2,$$

where $V^*_{t,\text{clip}}$ and G^*_t are internal quantities of Algorithm 4.

Proof of Lemma C.2. Given Lemma C.1, the remaining task is connecting $|S^*_{t,\text{clip}}|$ to $|S^*_t|$. To this end,

$$|S^*_t| \le \left|S^*_{t,\text{clip}}\right| + \sum_{i=1}^{t}\left(\prod_{j=i}^{t-1}\lambda_j\right)\left|g^*_i - g^*_{i,\text{clip}}\right|,$$

and the sum on the RHS is at most G^*_t following the proof of Theorem 3. □

The next lemma exploits the bounded domain assumption.

Lemma C.3. *If $\max_t r^*_t \le D$, then without knowing D, $\mathcal{A}_{CP}$ guarantees for all t,*

$$|S^*_t| \le 2\sqrt{V^*_{t,\text{clip}}}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right) + 15G^*_t\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right)^2,$$

where $V^*_{t,\text{clip}}$ and G^*_t are internal quantities of Algorithm 4.

Proof of Lemma C.3. Consider $S^*_{t,\text{clip}}$ defined in Eq.(16). We now use induction to show that

$$\left|S^*_{t,\text{clip}}\right| \le 2\sqrt{V^*_{t,\text{clip}}}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right) + 14G^*_t\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right)^2,$$

and after that, we complete the proof using $|S^*_t| \le \left|S^*_{t,\text{clip}}\right| + G^*_t$, just like Lemma C.2.

Concretely, note that such a statement on $\left|S^*_{t,\text{clip}}\right|$ trivially holds for $t=1$. Then, suppose it holds for any t. In the $t+1$-th round, there are two cases.

- **Case 1:** $r_{t+1} \le D$**.**

  Due to Lemma C.1, we have

$$\begin{aligned}\left|S^*_{t,\text{clip}}\right| &\le 2\sqrt{V^*_{t,\text{clip}}}\left(1+\sqrt{\log\left(1+2r_{t+1}\varepsilon^{-1}\right)}\right) + 13G^*_t\left(1+\sqrt{\log\left(1+2r_{t+1}\varepsilon^{-1}\right)}\right)^2\\ &\le 2\sqrt{V^*_{t,\text{clip}}}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right) + 13G^*_t\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right)^2.\end{aligned}$$

$$\begin{aligned}\left|S^*_{t+1,\text{clip}}\right| &\le \lambda_t\left|S^*_{t,\text{clip}}\right| + \left|g^*_{t+1,\text{clip}}\right|\\ &= 2\sqrt{\lambda_t^2 V^*_{t,\text{clip}}}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right) + 13\lambda_t G^*_t\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right)^2 + \left|g^*_{t+1,\text{clip}}\right|\\ &\le 2\sqrt{V^*_{t+1,\text{clip}}}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right) + 14G^*_{t+1}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right)^2.\end{aligned}$$

- **Case 2:** $r_{t+1} > D$**.**

  In this case, $r_{t+1} > r^*_{t+1}$ so the OCP gradient $g^*_{t+1} \ge 0$, and the clipped gradient $g^*_{t+1,\text{clip}} \ge 0$. Meanwhile, in order to have $r_{t+1} > D \ge 0$, we must have $S^*_{t,\text{clip}} \ge 0$ from the prediction rule. Then, due to the same signs of $g^*_{t+1,\text{clip}}$ and $S^*_{t,\text{clip}}$, we have

$$\left|S^*_{t+1,\text{clip}}\right| = \left|\lambda_t S^*_{t,\text{clip}} - g^*_{t+1,\text{clip}}\right|$$

$$
\begin{aligned}
&\leq \lambda_t \left|S^*_{t,\text{clip}}\right| \vee \left|g^*_{t+1,\text{clip}}\right| \\
&\leq 2\sqrt{\lambda_t^2 V^*_{t,\text{clip}}}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right) + 14\lambda_t G^*_t\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right)^2 \\
&\leq 2\sqrt{V^*_{t+1,\text{clip}}}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right) + 14G^*_{t+1}\left(1+\sqrt{\log\left(1+2D\varepsilon^{-1}\right)}\right)^2.
\end{aligned}
$$

Combining the two cases, the induction statement holds in the $t+1$-th round. Finally we use $|S^*_t| \leq \left|S^*_{t,\text{clip}}\right| + G^*_t$. □

With everything above, Theorem 5 is a simple corollary.

Theorem 5. *Without knowing D, $\mathcal{A}_{CP}$ guarantees that for all $T \in \mathbb{N}_+$, we have the discounted coverage bound*

$$
|S^*_T| \leq O\left(\sqrt{V^*_T \log(D\varepsilon^{-1})} \vee G^*_T \log(D\varepsilon^{-1})\right),
$$

and the discounted regret bound from Theorem 3.

Proof of Theorem 5. The regret bound trivially applies. The coverage bound follows from Lemma C.3 and the fact that $V^*_{t,\text{clip}} \leq V^*_t$. Then we consider the asymptotic regime of $D \gg \varepsilon$. □

D. Detail of Experiment

This section presents details of our experiment. Our setup builds on the great work of (Bhatnagar et al., 2023).

Setup We test our OCP algorithm (Algorithm 4, which is based on OCO Algorithm 1 and 2) in the context of classifying altered images that arrive in a sequential manner. Given a parameterized prediction set $\mathcal{C}_t(\cdot)$ dependent on the label provided by a base ML model, at each time step our algorithms predict the radius r_t that corresponds to the uncertainty of the base model's prediction, resulting in the prediction set $\mathcal{C}_t(r_t)$. We adopt the procedure, code, and base model from (Bhatnagar et al., 2023). Given a sequence of images, we expect that if images are increasingly "corrupted" by blur, noise, and other factors, the prediction set size must increase to account for the deviation from the base model's training distribution. We hypothesize that the rate of such a distribution shift also affects the OCP algorithms' performance, thus we test the cases where the corruption levels shift suddenly versus gradually.

Our algorithms are specifically designed to not use knowledge of the maximum magnitude D of the optimal radius r^*_t (i.e., the maximum uncertainty level). In some prediction scenarios, it is conceivable that D is impossible to know a priori and thus cannot be used as a hyperparameter. In contrast, certain algorithms from the literature use an empirical estimate of D from an offline dataset, which amounts to "oracle tuning". These include the Strongly Adaptive Online Conformal Prediction (SAOCP) and Scale-Free Online Gradient Descent (SF-OGD) proposed by (Bhatnagar et al., 2023). In our study, we create a modified version of SF-OGD called "Simple OGD", that does not use such an oracle tuning. The only hyperparameter that we set is the learning rate, which we set to 1 for Simple OGD. Note that despite the name, Simple OGD is also gradient adaptive, which differs from the ACI algorithm from (Gibbs & Candes, 2021).

Baselines We perform OCP for ten different algorithms:

1. MAGL-D: We test our Algorithm 1 with $\varepsilon = 1$ and discount factor $\lambda_t = 0.999$, which we name MAGL-D (**Mag**nitude Learner with **L**ipschitz Constant Estimate and **D**iscounting).
2. MAGL: We test Algorithm 2, the undiscounted algorithm that uses the running estimate of the Lipschitz constant, h_t, with $\varepsilon = 1$. We name this algorithm MAGL (**Mag**nitude Learner with **L**ipschitz constant estimate).
3. MAGDIS: We also test Algorithm 5 with $\varepsilon = 1$ and $\lambda_t = 0.999$, which is a simplified version of Algorithm 1 that essentially sets $h_t = 0$, does not clip g_t, and initializes $v_t > 0$. We name this algorithm MAGDIS (**Mag**nitude Learner with **Dis**counting).

Using the implementation from (Bhatnagar et al., 2023), we also obtain results for SAOCP, SF-OGD, Simple OGD, Standard Split Conformal Prediction (SCP) (Vovk et al., 2005), Non-Exchangeable SCP (NExCP) (Barber et al., 2023), Fully-Adaptive Conformal Inference (FACI) (Gibbs & Candès, 2022), and FACI-S (Bhatnagar et al., 2023).

Algorithm 5 Modified 1D magnitude learner on $[0, \infty)$.

Require: Hyperparameter $\varepsilon > 0$.
1: Initialize parameters $v_1 > 0$, $s_1 = 0$.
2: **for** $t = 1, 2, \ldots$ **do**
3: Define the unprojected prediction $\tilde{x}_t$,

$$\tilde{x}_t = \varepsilon \cdot \operatorname{erfi}\left(\frac{s_t}{2\sqrt{v_t}}\right).$$

4: Predict $x_t = \Pi_{[0,\infty)}(\tilde{x}_t)$, the projection of $\tilde{x}_t$ to the domain $[0, \infty)$.
5: Receive the 1D loss gradient $g_t \in \mathbb{R}$ and the discount factor $\lambda_{t-1} \in (0, \infty)$.
6: If $g_t \tilde{x}_t < g_t x_t$, define a surrogate loss gradient $\tilde{g}_t = 0$. Otherwise, $\tilde{g}_t = g_t$.
7: Update $v_{t+1} = \lambda_{t-1}^2 v_t + \tilde{g}_t^2$, $s_{t+1} = \lambda_{t-1} s_t - \tilde{g}_t$.
8: **end for**

Metrics We choose a targeted coverage rate of 90% for all experiments, which means $\alpha = 0.1$. To quantify the algorithms' performance, we follow the definitions from (Bhatnagar et al., 2023) to evaluate the coverage (local and average), prediction width (local and average), worst-case local coverage error (LCE_k), and runtime. They are defined as follows.

- First, let Y_t be the true label of the t-th image, and for brevity, let $\widehat{C}_t \leftarrow \mathcal{C}_t(r_t)$ be the t-th prediction set over the labels. Then, err_t is the indicator function of miscoverage at time t:

$$\mathrm{err}_t := \mathbf{1}\left[Y_t \notin \widehat{C}_t\right],$$

  where $\mathrm{err}_t = 1$ if the prediction set $\widehat{C}_t$ does not include the true label Y_t.

- **Local Coverage** Over any sliding window, k, the local coverage is defined as:

$$\text{Local Coverage}(t) := \frac{1}{k}\sum_{i=t}^{t+k-1}(1 - \mathrm{err}_i).$$

  For all trials, we used an interval length of $k = 100$.

- **Average Coverage** The average coverage is similarly defined, but averaged over the total time steps T. For all experiments, $T = 6011$.

$$\text{Avg. Coverage} := \frac{1}{T}\sum_{i=0}^{T}(1 - \mathrm{err}_i).$$

- **Local Width** The local width is the cardinality of $\widehat{C}_t$, averaged over the length k time window:

$$\text{Local Width}(t) := \frac{1}{k}\sum_{i=t}^{t+k-1}\left|\widehat{C}_t\right|.$$

  It is compared to the "best fixed" local width defined as follows. If we let $C_t^* \leftarrow \mathcal{C}_t(r_t^*)$ denote the optimal prediction set had we known the optimal radius r_t^* beforehand, then the *best fixed local width* $\text{Local Width}^*(t)$ is defined as the $1 - \alpha$ quantile of $\{|C_i^*|\,; i \in [t : t + k - 1]\}$.

- **Average Width** Similarly, the average width is defined as:

$$\text{Avg.Width} := \frac{1}{T}\sum_{t=1}^{T}\left|\widehat{C}_t\right|.$$

- **Local Coverage Error (LCE)** The LCE over the sliding window of length k is defined as:

$$\mathrm{LCE}_k := \max_{\tau,\tau+k-1\subseteq[1,T]} \left| \alpha - \frac{1}{k} \sum_{t=\tau}^{\tau+k-1} \mathrm{err}_t \right|.$$

  Essentially, it means the largest deviation of the empirical miscoverage rate (evaluated over sliding time windows of length k) from the targeted miscoverage rate α.

Main results Our main results are shown in Figure 1 and Table 1 (Section 4). The purpose of Figure 1 is to demonstrate the dependence of the local coverage and the local width on (1) sudden and (2) gradual distribution shifts (i.e., the time-varying corruption level).[9] The purpose of Table 1 is to summarize the performance of our algorithms and compare them to the baselines; the results there correspond to the case of sudden distribution shift.

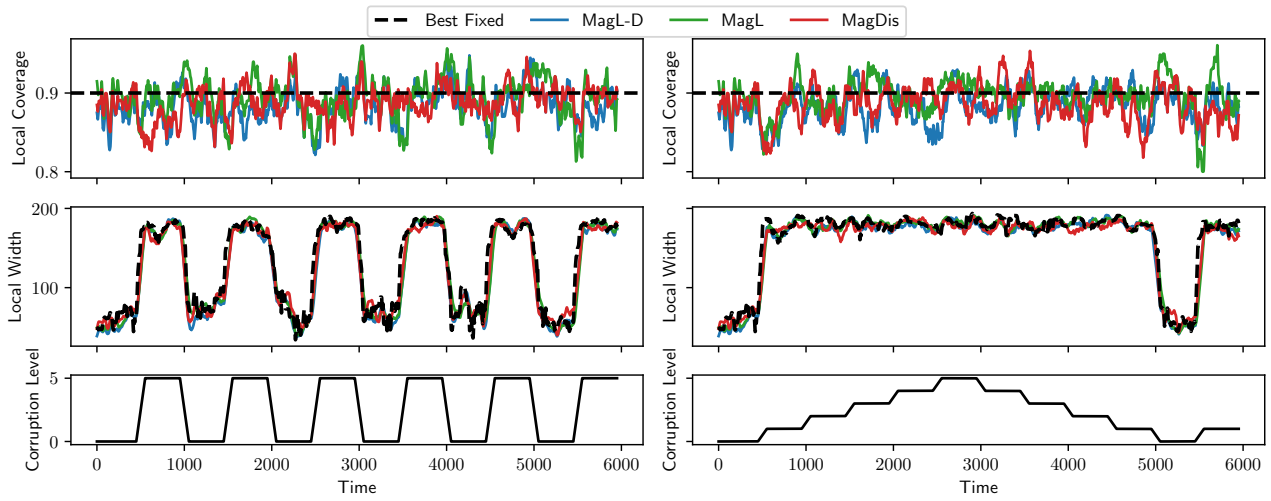

Figure 1. The local coverage (first row), local width (second row), and corresponding corruption level (third row) of our algorithms. Results are obtained using corrupted versions of TinyImageNet, with time-varying corruption level (distribution shift). (Left) Results for *sudden* changes in corruption level. (Right) Results for *gradual* changes in corruption level. The distribution shifts every 500 steps. Moving averages are plotted with a window size of 100 time steps ($k = 100$).

Figure 1 justifies the validity of our algorithms. Specifically, the local coverage of our algorithms fluctuate around the target coverage of 0.9 for both sudden and gradual distribution shifts. Similarly, the local width approximately replicates the best fixed local width, $\mathrm{Local\,Width}^*(t)$, as shown in Figure 1 (middle row).

Hyperparameter sensitivity We also test the algorithms' sensitivities to the offline estimate of D, which we denote as D_{est}. The baselines SAOCP and SF-OGD require this parameter to initialize, while Simple OGD and all three of our algorithms do not. To this end, we rerun the experiments above and change the ratio of D_{est} to the true value D. The following settings are tested:

$$\frac{D_{\mathrm{est}}}{D} = \left\{10^{-3},\quad 10^{-2},\quad 10^{-1},\quad 10^{0},\quad 10^{1},\quad 10^{2},\quad 10^{3}\right\}.$$

We plot the influence of D_{est} on the average coverage and the average width in Figure 2. For these experiments, we use the case when image corruptions are sudden. Coverage being much larger or much less than 0.9 is not a desirable behavior; given satisfactory coverage, lower width is desirable.

As D_{est}/D increases, both the average coverage and average width increase for SAOCP and SF-OGD. In the opposite direction, the average coverage and average width also decrease as D_{est}/D decrease for SAOCP and SF-OGD. When $D_{\mathrm{est}} = D$, Simple OGD and SF-OGD have approximately equivalent performance. This is not to say that using an offline

[9] For better visibility, a 1D Gaussian filter is applied to the local width and the local coverage plots, same as in (Bhatnagar et al., 2023).

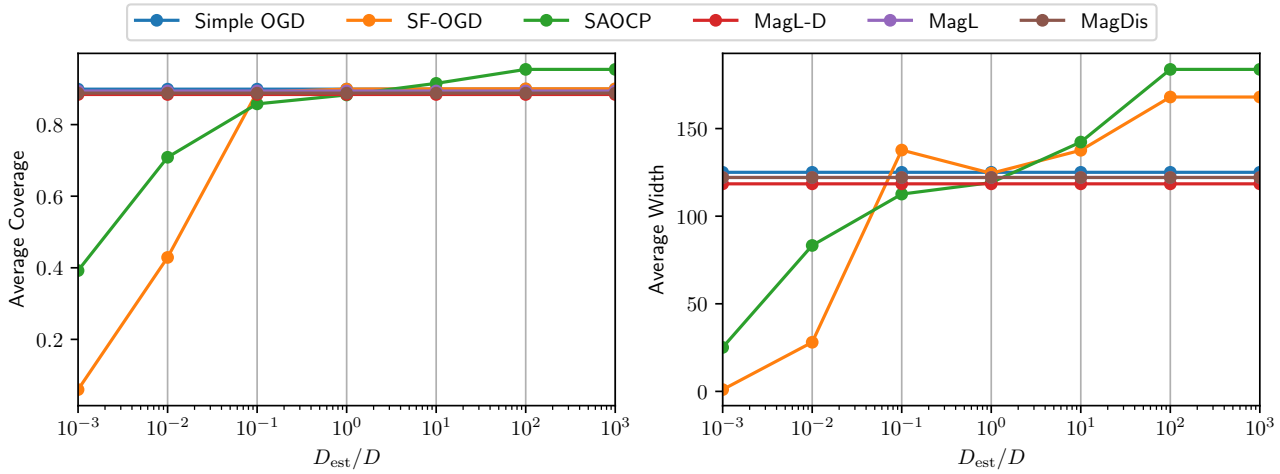

Figure 2. The average coverage (first row) and average width (second row) as a function of the estimated maximum radius, D_{est} relative to the true radius D. The performance of SF-OGD and SAOCP (Bhatnagar et al., 2023) are sensitive to D_{est}/D. Averages are taken over the entire time horizon, where the total time steps $T = 6011$.

estimate of D_{est}/D does not improve performance. The catch is that, for the particular experimental dataset, setting the learning rate to 1 in Simple OGD coincidentally just happens to be a well-picked learning rate, given the true value of the maximum radius magnitude D. In contrast, our magnitude learners remain fixed under variations in D_{est}/D, since they do not need D_{est} to initialize. The baselines SCP, NExCP, FACI, and FACI-S are not presented in Figure 2 as to better highlight the performance of SAOCP, OGD, and our magnitude learners. Benchmarks on these additional baselines can be found in (Bhatnagar et al., 2023).

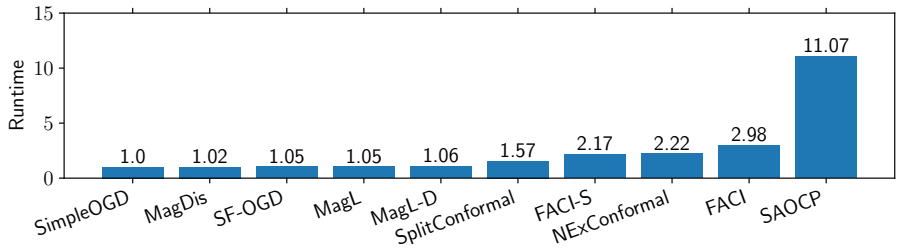

Figure 3. The runtime per time step of each algorithm, normalized to the runtime of *Simple OGD*. The runtime of SAOCP is longest due to it being a meta-algorithm that initializes SF-OGD on each time step.

Runtime We also measure the time for each algorithm to complete a single prediction. Runtime results are provided in Figure 3. The results are normalized relative to the runtime of Simple OGD. Note the runtime standard deviations in Table 1. Accounting for standard deviations, the runtime differences between Simple OGD, SF-OGD, MAGL, and MAGDIS are negligible. The runtime of SAOCP is longest due to it being a meta-algorithm that initializes SF-OGD on each time step, c.f., Appendix A.