Optimal Comparator Adaptive Online Learning with Switching Cost

Zhiyu Zhang
Boston University
zhiyuz@bu.edu

Ashok Cutkosky
Boston University
ashok@cutkosky.com

Ioannis Ch. Paschalidis
Boston University
yannisp@bu.edu

Abstract

Practical online learning tasks are often naturally defined on unconstrained domains, where optimal algorithms for general convex losses are characterized by the notion of *comparator adaptivity*. In this paper, we design such algorithms in the presence of switching cost – the latter penalizes the typical optimism in adaptive algorithms, leading to a delicate design trade-off. Based on a novel *dual space scaling* strategy discovered by a continuous-time analysis, we propose a simple algorithm that improves the existing comparator adaptive regret bound [ZCP22a] to the optimal rate. The obtained benefits are further extended to the expert setting, and the practicality of the proposed algorithm is demonstrated through a sequential investment task.[1]

1 Introduction

Online learning [CBL06, Haz16, Ora19] is a powerful framework for modeling sequential decision making tasks, such as neural network training, financial investment and robotic control. In each round, an agent picks a prediction x_t in a convex domain $\mathcal{X}$, receives a convex and Lipschitz loss function l_t that depends on $x_1, \dots, x_t$, and suffers the loss $l_t(x_t)$. The goal is to ensure that in any environment, the cumulative loss of the agent is never much worse than that of any fixed decision $u \in \mathcal{X}$. That is, one aims to upper-bound the regret $\sum_{t=1}^{T} [l_t(x_t) - l_t(u)]$, for all time horizon $T \in \mathbb{N}_+$, comparator $u \in \mathcal{X}$ and loss sequence $l_1, \dots, l_T$.

If there exists a best fixed decision $u^* = \max_x \sum_{t=1}^{T} l_t(x)$ in hindsight, then the regret with respect to any $u \in \mathcal{X}$ is dominated by the regret with respect to u^*. In the context of training machine learning models, u^* corresponds to the model parameter that minimizes the training error. Hence, intuitively, the regret bound characterizes how fast the algorithm finds u^* through training.

Most classical online learning algorithms are *minimax* in nature, only tuned to optimize the worst-case regret. For example, if the domain $\mathcal{X}$ is bounded, then the maximum distance between the best fixed decision u^* and the initialization x_1 of the algorithm is the diameter D of $\mathcal{X}$. Only considering this worst case, it suffices to use *Online Gradient Descent* (OGD) [Zin03] with learning rate $\eta_t \propto D/\sqrt{t}$. The result is a $O(D\sqrt{T})$ regret bound that holds uniformly for all comparators $u \in \mathcal{X}$. Such a minimax reasoning is prevalent, but limited in two substantial ways.

1. It requires a bounded domain. Many practical problems are naturally unconstrained, making such arguments inapplicable.
2. Practical applications are usually not the worst case, which means that the minimax bound $O(D\sqrt{T})$ is typically loose. Think about the situation where we have a prior guess of u^*, from either domain knowledge or pre-training. Using this prior as the initial prediction x_1, we should

[1] Future versions available at https://arxiv.org/abs/2205.06846.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

expect a provable gain over arbitrarily initializing the algorithm – specifically, had we known the *correct* distance $\|u^* - x_1\|$, we could have picked $\eta_t \propto \|u^* - x_1\| / \sqrt{t}$ in OGD, resulting in $O(\|u^* - x_1\| \sqrt{T})$ regret. Achieving this goal *without the prior knowledge of* $\|u^* - x_1\|$ is of both theoretical and practical importance.

Recent studies of *comparator adaptive* online learning[2] [LS15, OP16, CO18] aim to address these issues. The domain does not need to be bounded, and the regret bound is $\tilde{O}(d(u, x_1)\sqrt{T})$, where $d(\cdot, \cdot)$ is some suitable distance measure. Intuitively, these algorithms are both *optimistic* and *robust*: given prior information on u^*, we can pick x_1 such that $d(u^*, x_1)$, and consequently the regret bound, are both low. Meanwhile, even when our initialization x_1 is wrong (i.e., $d(u^*, x_1)$ is large), the regret bound is still *almost as good* (up to logarithmic factors) as that of the minimax algorithm with the best tuning in hindsight. Such properties have shown benefits in diverse applications, e.g., [OT17, JO19, vdH19].

In this paper, we extend the design of these algorithms to a classical setting with switching costs. Here the agent is penalized not only by its loss, but also by how fast it changes its predictions. Practically, switching costs are useful whenever the smooth operation of a system is favored, such as in network routing, control of electrical grid, portfolio management with transaction costs, etc. Recently they also naturally show up in online decision problems with *long term effects*, such as *nonstochastic control* [ABH+19]. With a given weight $\lambda \geq 0$ and a norm $\|\cdot\|$, our goal is to guarantee a comparator adaptive bound for the *augmented regret*

$$\sum_{t=1}^{T} [l_t(x_t) - l_t(u)] + \lambda \sum_{t=1}^{T-1} \|x_t - x_{t+1}\| .$$

While gradient descent can incorporate switching costs by simply scaling its learning rate, extending comparator adaptive algorithms is a lot harder. Just like other adaptive algorithms [DHS11, DGSS15], the key idea of comparator adaptivity is to quickly respond to the incoming information and hedge aggressively. Switching costs, on the other hand, encourage the agent to stay still. Therefore, achieving our goal requires a delicate balance between these two opposite considerations.

Similar trade-offs between adaptivity and switching costs have led to intriguing results in the past. For example, Gofer [Gof14] showed that the gradient variance adaptivity well-studied in the switching-free setting is impossible with normed switching costs, thus establishing a clear separation caused by the latter. Daniely and Mansour [DM19] showed that a common analytical technique for switching costs is incompatible to the so-called *"strong adaptivity"* (i.e., a form of adaptivity w.r.t. nonstationary comparators). Regarding comparator adaptivity, our prior work [ZCP22a] proposed the first comparator adaptive algorithm with switching costs, but the obtained regret bound does not achieve the optimality criterion of the switching-free setting. The present paper closes this gap.

1.1 Contribution

We develop comparator adaptive algorithms for two classical settings: (i) *Online Linear Optimization* (OLO) with switching cost; (ii) *Learning with Expert Advice* (LEA) with switching cost.

1. For one-dimensional unconstrained OLO with switching costs, assuming loss gradients $|g_t| \leq 1$ and initial prediction[3] $x_1 = 0$, we propose an algorithm that guarantees
$$\sum_{t=1}^{T} g_t(x_t - u) + \lambda \sum_{t=1}^{T-1} |x_t - x_{t+1}| \leq C\sqrt{\lambda T} + |u|\, O\left(\sqrt{\lambda T \log(C^{-1}\,|u|)}\right),$$
where $C > 0$ is any hyperparameter chosen by the user (Section 2). Our bound achieves several forms of optimality with respect to λ, $|u|$ and T, improving the prior work [ZCP22a]. Extensions to bounded domains and general dimensional domains are presented, which include some new technical components (Appendix B).
2. Converting the above result from OLO to LEA, we demonstrate how classical conversion techniques [LS15, OP16] are *designed* to have large switching costs, and then propose a fix with a clear geometric interpretation. This leads to the first comparator adaptive algorithm for LEA with switching costs (Section 3).

[2] Also called *parameter-free* online learning due to historical reasons.

[3] For general x_1, we can replace $|u|$ in the regret bound by $|u - x_1|$.

Technically, our improvement over [ZCP22a] relies on a novel *dual space scaling* strategy. This is actually not guessed, but *systematically discovered* by a continuous-time analysis (Section 2.4), whose procedure follows another prior work of ours [ZCP22b]. In the continuous-time limit, it becomes evident what kinds of algorithmic structures from the switching-free setting are transferable to the setting with switching costs. Indeed, revealing generalizable knowledge is a key benefit of the continuous-time analysis, which was not demonstrated in [ZCP22b]. As an added bonus, both our OLO algorithm and its analysis are considerably simpler than those from [ZCP22a].

Concluding these theoretical results, our OLO algorithm is applied to a portfolio management task with transaction costs (Appendix D). Numerical results support its superiority over the existing approach [ZCP22a].

1.2 Related work

Online learning basics Throughout this paper we will only consider linear losses. The generality of our setting is preserved, since convex losses can be reduced to linear losses through the relation $\sum_{t=1}^{T}[l_t(x_t)-l_t(u)]\le\sum_{t=1}^{T}\langle\nabla l_t(x_t),x_t-u\rangle$ [Haz16, Ora19]. Online learning with linear losses is called *Online Linear Optimization* (OLO). As its important special case, *Learning with Expert Advice* (LEA) considers OLO on a probability simplex, but aims at a different form of regret bound due to its different geometry.

Classical minimax approaches in online learning include *Online Mirror Descent* (OMD) and *Follow the Regularized Leader* (FTRL), with *Online Gradient Descent* (OGD) being their most well-known special case. We write "gradient descent" as the minimax baseline for the ease of exposition. Moreover, both OMD and FTRL have elegant duality interpretations [Ora19, Section 6.4.1 and 7.3], involving simultaneous updates on the primal space (the domain $\mathcal{X}$) and the dual space (the space of gradients). We will exploit this duality in our analysis.

Comparator adaptive online learning Also known as *parameter-freeness*, comparator adaptive online learning aims at matching the performance of the optimally-tuned gradient descent in hindsight, without knowing the correct tuning parameter. The associated regret bound can appear in different forms, depending on the specific learning setting.

1. For LEA, a comparator adaptive bound has the form $O\left(\sqrt{T\cdot\mathrm{KL}(u||\pi)}\right)$, where u and π are distributions on the expert space representing the comparator and a user-chosen prior. Such an idea was initiated in [CFH09], and the analysis was improved and extended by a series of works [CV10, LS15, KVE15, CLW21, NBC+21, PLH22]. Notably, a comparator adaptive LEA algorithm naturally induces a bound on the *ε-quantile regret* – the regret with respect to the ε-quantile best expert. The latter is particularly meaningful when the number of experts is large. Lower bounds were considered in [NBC+21].

   We will present a nonasymptotic improvement of the $\sqrt{\mathrm{KL}}$ divergence in this paper. Frameworks that generalize root KL to *f-divergences* have been studied in [Alq21, NBC+21], but to our knowledge, no existing algorithm guarantees a better divergence term than root KL, even without switching costs.
2. For OLO, typical comparator adaptive bounds are $C+\|u\|\,O\left(\sqrt{T\log(C^{-1}\|u\|_*T)}\right)$ or $C\sqrt{T}+\|u\|\,O\left(\sqrt{T\log(C^{-1}\|u\|_*)}\right)$, where a prior x_1 can be incorporated by letting $u\leftarrow u-x_1$. These two bounds are both *Pareto-optimal* (see [ZCP22b] for a detailed explanation), as they represent different trade-offs on the *loss* (the regret at $u=x_1$) and the *asymptotic regret* (when $\|u-x_1\|$ is large). Existing works [MO14, CO18, FRS18, MK20, JC22] were mostly independent of the LEA setting, but unified views were presented in [FRS15, OP16]. Lower bounds were studied in [SM12, Ora13, ZCP22b].

Switching cost Motivated by numerous applications, switching costs in online decision making have been studied from many different angles. For example, beside online learning, the online algorithm community has investigated settings like *smoothed online optimization* [CGW18, GLSW19, LQL20] and *convex body chasing* [BLLS19, Sel20], where the loss function l_t is observed *before* the agent picks the prediction x_t. There, the switching cost is the key consideration that prevents the trivial strategy $x_t\in\arg\min_x l_t(x)$. As for online learning, an additional complication is that x_t (e.g., the investment portfolio) should be selected without knowing l_t (e.g., tomorrow's stock price).

Even within online learning, there are several ways to model switching costs. In cases like network routing, every switch means changing the packet route, which can be costly. Therefore, one needs a *lazy* agent whose amount of switches (or its expectation) [KV05, GVW10, AT18, CYLK20, SK21] is as low as possible – a good modeling candidate is $\mathbf{1}[x_t \neq x_{t+1}]$. Alternatively, one could take a *smooth* view [ABL+13, BCKP21, WWYZ21, ZJLY21] where the agent can perform as many switches as it wishes, as long as the cumulative distance of switching is low – in this view, switching cost can be a norm $\|x_t - x_{t+1}\|$ or its smoothed variant $\|x_t - x_{t+1}\|^2$. The present work primarily considers the L_1 norm switching cost motivated by the transaction cost in some financial applications. Notably, for LEA, the L_1 norm unifies the lazy view and the smooth view [DM19, Section 5.2].

Although switching costs have been extensively studied, existing works on the combination of adaptivity and switching cost are quite sparse. As one should carefully trade off these two opposite requirements, there have been interesting impossibility results [Gof14, DM19], highlighted in our introduction. In this regard, one should not believe that every classical adaptivity can be naturally achieved with switching costs. The present paper shows that the optimal comparator adaptivity can indeed be achieved, thus improving the suboptimal result from [ZCP22a].

Relation to downstream problems More generally, incorporating switching costs amounts to considering a *history-dependent* adversary: it can pick loss functions that depend not only on the instantaneous prediction x_t, but also on the previous prediction x_{t-1}. One could further generalize this setting to *online learning with memory* [CBDS13, AHM15], where the loss depends on a fixed-length prediction history, and finally to *dynamical systems* [ABH+19, SSH20, Sim20], where the entire history matters. In fact, a common procedure in *nonstochastic control* [ABH+19] is to bound the risk in the future by a properly scaled switching cost. Achieving comparator adaptivity with switching costs may benefit these important downstream problems as well, by making algorithms *easy to combine* [Cut19, Cut20, ZCP22a].

Continuous-time approach to online learning Finally, on the technical side, our methodology builds on an emerging continuous-time perspective of online learning. From a theoretical angle, Kohn and Serfaty [KS10] demonstrated a rigorous connection between *Partial Differential Equations* (PDE) and discrete-time repeated games. More recently, such a connection has led to *algorithmic benefits* in minimax LEA [Zhu14, Rok17, DK20, KKW20, HLPR20, BEZ20a, BEZ20b, GHP22], ε-quantile bounds [PLH22] and comparator adaptive OLO [ZCP22b]. The key idea is that online learning algorithms in the continuous-time limit can be more easily parameterized and analyzed. In this paper we will show an additional bonus: generalizing algorithmic insights is also easier in the continuous-time limit.

1.3 Notation

Let f^* be the Fenchel conjugate of a function f. $\Delta(d)$ represents the d-dimensional probability simplex; KL and TV denote the KL divergence and the total variation distance, respectively. For two integers $a \leq b$, $[a : b]$ is the set of all integers c such that $a \leq c \leq b$. $\log$ represents the natural logarithm when the base is omitted. Throughout this paper, "increasing" and "positive" are not strict (i.e., include equality as well).

For a twice differentiable function $V(t, S)$ where t represents time and S represents a spatial variable, let $\nabla_t V$, $\nabla_{tt} V$, $\nabla_S V$ and $\nabla_{SS} V$ be the first and second order partial derivatives. In addition, we define discrete derivatives as

$$\begin{aligned} \bar{\nabla}_t V(t,S) &:= V(t,S) - V(t-1,S), \\ \bar{\nabla}_S V(t,S) &:= \frac{1}{2}\left[V(t,S+1) - V(t,S-1)\right], \\ \bar{\nabla}_{SS} V(t,S) &:= V(t,S+1) + V(t,S-1) - 2V(t,S). \end{aligned} \tag{1}$$

2 OLO with switching cost

This section presents our main result, a comparator adaptive OLO algorithm with switching costs. We will focus on the 1D unconstrained setting. Due to limited space, extensions to general settings are deferred to Appendix B.

2.1 The 1D unconstrained setting

We consider the domain $\mathcal{X} = \mathbb{R}$, a Lipschitz constant $G > 0$ for the loss gradients, and a weight $\lambda \geq 0$ for switching costs. In the t-th round, the agent predicts $x_t \in \mathbb{R}$, receives a loss gradient $g_t \in [-G, G]$ that depends on past predictions $x_{1:t}$, and suffers an augmented loss $g_t x_t + \lambda |x_t - x_{t-1}|$ (w.l.o.g., let $x_0 = x_1 = 0$). The performance metric is the augmented regret for all $u \in \mathbb{R}$ and $T \in \mathbb{N}_+$:

$$\mathrm{Regret}_T^\lambda(u) := \sum_{t=1}^{T} g_t(x_t - u) + \lambda \sum_{t=1}^{T-1} |x_t - x_{t+1}| . \tag{2}$$

Ignoring the dependence on G for now, our goal is to show a comparator adaptive bound $\tilde{O}\left(|u| \sqrt{\lambda T}\right)$, more specifically the optimal rates $C + |u| O\left(\sqrt{\lambda T \log(C^{-1} \lambda |u| T)}\right)$ or $C\sqrt{\lambda T} + |u| O\left(\sqrt{\lambda T \log(C^{-1} |u|)}\right)$ for any hyperparameter $C > 0$. These two cases are equivalent via the standard doubling trick [SS11], as discussed in [ZCP22b].

For minimax algorithms like bounded domain gradient descent [Zin03], one can use scaled learning rates $\eta_t \propto 1/\sqrt{\lambda t}$ to ensure that both sums in (2) are $O\left(\sqrt{\lambda T}\right)$, thus obtaining a combined $O\left(\sqrt{\lambda T}\right)$ regret bound. However, such a divide-and-conquer approach does not apply to comparator adaptive algorithms, as one cannot *separately* show the desirable bound on the two sums in (2). To see this, suppose one could guarantee the second sum alone is at most $1 + |u| O\left(\sqrt{T \log(|u| T)}\right)$; here we only focus on the dependence on $|u|$ and T. Since this cumulative switching cost is an algorithmic quantity *independent of the comparator*, we can take infimum with respect to u and obtain a "budget" of 1 for this sum. Following this argument, $|x_T| \leq |x_1| + \sum_{t=1}^{T-1} |x_t - x_{t+1}| = O(1)$. That is, the algorithm should only predict around the origin, which clearly leads to large regret with respect to far-away comparators, under certain loss sequences.

The challenge can be motivated in another way. As shown in [Ora19, Figure 9.1], the one-step switching cost $|x_t - x_{t+1}|$ of comparator adaptive algorithms can grow exponentially with respect to t, whereas such a quantity is uniformly bounded in gradient descent. In fact, the exponential growth is the key mechanism for comparator adaptive algorithms to cover an unconstrained domain fast enough (thus improving minimax algorithms). This is however problematic when switching is also penalized, as one can no longer control the switching cost by uniformly scaling $|x_t - x_{t+1}|$.

2.2 Switching-adjusted potential

To address these issues, one should bound the switching cost and the standard OLO regret in a *unified* framework, instead of treating them separately. Our prior work [ZCP22a] used the classical coin-betting approach from [OP16, CO18], which is included in Appendix A.1 for completeness. In the t-th round, the algorithm maintains a quantity Wealth_{t-1}; by picking a *betting fraction* $\beta_t \in [0, 1]$, the prediction is set to $x_t = \beta_t \mathrm{Wealth}_{t-1}$. To further ensure low switching costs, the betting fraction β_t is capped by a decreasing upper bound $O(1/\sqrt{t})$. Although it is analytically nontrivial, such a hard threshold is conservative, which could be the reason of our previous suboptimal result.

In contrast, the present paper follows the *potential framework* explored by a parallel line of works [MO14, FRS18, MK20, ZCP22b]. Generally, these algorithms are defined by a potential function $V(t, S)$, where t represents the time index, and S represents a "sufficient statistic" that summarizes the history. In each round, the algorithm computes $S_{t-1} = -\sum_{i=1}^{t-1} g_i/G$, and the prediction x_t is the derivative $\nabla_S V$ evaluated at (t, S_{t-1}). We will specifically consider Algorithm 1, which is a variant based on the discrete derivative $\overline{\nabla}_S V$, cf. (1).

One could think of the potential framework as the dual approach of FTRL – the potential function and the regularizer are naturally Fenchel conjugates. While the FTRL analysis relies on a one-step regret bound on the *primal space* (the domain $\mathcal{X}$, cf. [Ora19, Lemma 7.1]), the potential framework constructs a similar one-step relation on the *dual space* (the space of S_t, cf. [ZCP22b, Lemma 3.1]). Along this interpretation, **our key idea is to incorporate switching costs by scaling on the dual space, rather than only on the primal space.** That is, given a potential function that works without switching costs, we scale the sufficient statistic sent to its second argument by a function of λ.

Algorithm 1 One-dimensional unconstrained OLO with switching costs.

Require: A hyperparameter $C > 0$, the Lipschitz constant G, and a potential function $V(t, S)$ that implicitly depends on λ and G. Initialize $S_0 = 0$.

1: **for** $t = 1, 2, \ldots$ **do**
2: Predict $x_t = \bar{\nabla}_S V(t, S_{t-1})$, and receive the loss gradient g_t. Let $S_t = S_{t-1} - g_t/G$.
3: **end for**

To better demonstrate this idea, let us first consider a quadratic potential $V(t, S) = (1/2) \cdot CGS^2$. The potential method suggests the prediction $x_t = \nabla_S V(t, S_{t-1}) = C \sum_{i=1}^{t-1} g_i = x_{t-1} - Cg_{t-1}$, which is simply gradient descent with learning rate C. Scaling on the primal space means scaling V directly, while scaling on the dual space means scaling the sufficient statistic S. It is clear that both cases are *equivalent* to scaling the effective learning rate, which is the standard way to incorporate switching costs in bounded domain gradient descent. In other words, for this gradient descent potential, the two types of scaling are essentially the same.

Now, to achieve optimal comparator adaptivity, we need a better potential where scaling on the dual space actually makes a difference. With a parameter α that will eventually depend on λ, we consider Algorithm 1 induced by the potential

$$V_\alpha(t, S) = C\sqrt{\alpha t}\left[2\int_0^{S/\sqrt{4\alpha t}}\left(\int_0^u \exp(x^2)dx\right)du - 1\right]. \tag{3}$$

When the Lipschitz constant $G = 1$, it has been shown [ZCP22b] that $\alpha = 1/2$ leads to comparator adaptivity without switching costs. Here we use $\alpha = 4\lambda G^{-1} + 2$, which amounts to scaling *both* the primal space and the dual space: on the primal space, we scale up the overall prediction by $\Theta(\sqrt{\lambda G^{-1} + 1})$, and on the dual space we scale down the sufficient statistic S by $\Theta(1/\sqrt{\lambda G^{-1} + 1})$. The latter gives us the optimal comparator adaptive bound (i.e., Pareto-optimal rate in $|u|$ and T), while the former helps us obtain the optimal rate in λ. Due to incorporating λ into the potential function V_α, we call our approach the *switching-adjusted potential method*.

Although the dual space scaling strategy and the particular structure of V_α may seem mysterious at first glance, they are actually *derived* from a continuous-time analysis. To proceed, we will first present the performance guarantee in the next subsection, and then revisit the derivation of this strategy in Section 2.4.

2.3 Optimal comparator adaptive bound

Despite its simplicity, our approach improves the result from our prior work [ZCP22a] by a considerable margin.

Theorem 1. *If $\alpha = 4\lambda G^{-1} + 2$, then Algorithm 1 induced by the potential V_α guarantees*

$$\mathrm{Regret}_T^\lambda(u) \le \sqrt{(4\lambda G + 2G^2)T}\left[C + |u|\left(\sqrt{4\log\left(1 + \frac{|u|}{C}\right)} + 2\right)\right],$$

for all $u \in \mathbb{R}$ and $T \in \mathbb{N}_+$.

Theorem 1 simultaneously achieves several forms of optimality.

1. Pareto-optimal loss-regret trade-off: considering the dependence on u and T, $\mathrm{Regret}_T^\lambda(u) = O\left(|u|\sqrt{T\log|u|}\right)$, while the *cumulative loss* satisfies $\mathrm{Regret}_T^\lambda(0) = O(\sqrt{T})$. An existing lower bound [ZCP22b, Theorem 10] shows that even without switching costs, all algorithms satisfying a $O(\sqrt{T})$ loss bound must suffer a $\Omega\left(|u|\sqrt{T\log|u|}\right)$ regret bound. In this sense, our algorithm attains a *Pareto-optimal* loss-regret trade-off, in a strictly generalized setting with switching costs.
2. On T alone: $\mathrm{Regret}_T^\lambda(u) = O(\sqrt{T})$. Despite achieving comparator adaptivity, the asymptotic rate on T is still the optimal one, matching the well-known minimax lower bound.
3. On λ alone: $\mathrm{Regret}_T^\lambda(u) = O(\sqrt{\lambda})$. Our bound has the optimal dependence on the switching cost weight [GVW10, Theorem 5].

To compare Theorem 1 to [ZCP22a], we have to convert them to the same loss-regret trade-off, i.e., both guaranteeing $\mathrm{Regret}_T^\lambda(0) = O(1)$ or $\mathrm{Regret}_T^\lambda(0) = O(\sqrt{T})$. Here we take the first approach – details are presented in Appendix A.4. Let us only consider the dependence on u and T.[4] By a doubling trick, our bound can be converted to $C + |u|\, O\left(\sqrt{T \log(C^{-1}\,|u|\,T)}\right)$, which improves the rate $C + |u|\, O\left(\sqrt{T}\log(C^{-1}\,|u|\,T)\right)$ from [ZCP22a, Theorem 1]. Specifically, our converted upper bound also attains Pareto-optimality in this regime (i.e., matching the lower bound $\Omega\left(|u|\,\sqrt{T\log(|u|\,T)}\right)$ in [Ora13]), whereas the existing approach does not.

The proof of Theorem 1 is sketched below, with the formal analysis deferred to Appendix A.3. It mostly follows a standard potential argument, which is another benefit over the existing approach – the idea of this proof is easier to interpret and generalize.

Proof sketch of Theorem 1 To begin with, the first step is to show a one-step bound on the growth rate of the potential. If there is no switching cost, then the *Discrete Itô formula* can serve this purpose, which applies to any convex potential V. It is an established result in the probability literature [Kud82, Fuj08, Kle13], and Harvey et al. [HLPR20] first applied it to minimax LEA. The version below is from our prior work, which is a small variant that removes the LEA context.

Lemma 2.1 (Lemma 3.1 of [ZCP22b])**.** *If the potential function $V(t,S)$ is convex in S, then against any adversary, Algorithm 1 guarantees for all $t \in \mathbb{N}_+$,*

$$V(t,S_t) - V(t-1,S_{t-1}) \le -G^{-1}g_t x_t + \bar{\nabla}_t V(t,S_{t-1}) + (1/2)\cdot\bar{\nabla}_{SS}V(t,S_{t-1}).$$

Our key observation is the following lemma, which incorporates switching costs into V_α. Note that the structure of V_α is important here.

Lemma 2.2. *For all $\alpha > 0$, consider Algorithm 1 induced by the potential V_α. For all $t \in \mathbb{N}_+$,*

$$|x_t - x_{t+1}| \le \bar{\nabla}_S V_\alpha(t, S_{t-1}+1) - \bar{\nabla}_S V_\alpha(t, S_{t-1}-1).$$

Combining the above, if we define

$$\Delta_t := \bar{\nabla}_t V_\alpha(t,S_{t-1}) + \frac{1}{2}\bar{\nabla}_{SS}V_\alpha(t,S_{t-1}) + G^{-1}\lambda\left[\bar{\nabla}_S V_\alpha(t,S_{t-1}+1) - \bar{\nabla}_S V_\alpha(t,S_{t-1}-1)\right], \tag{4}$$

then a telescopic sum yields a *cumulative loss bound*

$$\mathrm{Regret}_T^\lambda(0) \le \sum_{t=1}^T (g_t x_t + \lambda\,|x_t - x_{t+1}|) \le -G\cdot V_\alpha(T,S_T) + G\sum_{t=1}^T \Delta_t.$$

To proceed, we need to control the residual term Δ_t, which may seem problematic due to its complicated form. Fortunately, a careful analysis shows that Δ_t *vanishes* with a proper choice of α.

Lemma 2.3. *If $\alpha \ge 4\lambda G^{-1} + 2$, then for all t and against any adversary, $\Delta_t \le 0$.*

Finally, with the updated loss bound $\mathrm{Regret}_T^\lambda(0) \le -G\cdot V_\alpha(T,S_T)$, our regret bound follows from the classical loss-regret duality [MO14, Ora19].

2.4 Continuous-time derivation

Now let us derive our dual space scaling strategy from a continuous-time perspective. Technically, the procedure is analogous to another prior work of ours [ZCP22b], which studies optimal potential functions for the standard OLO setting without switching costs. Before starting, we need a generalized definition of the discrete derivative, with a tunable gap increment δ.

$$\bar{\nabla}_S^\delta V(t,S) = \frac{1}{2\delta}\left[V(t,S+\delta) - V(t,S-\delta)\right].$$

Note that the choice of $\delta = 1$ recovers $\bar{\nabla}_S V(t,S)$ in Algorithm 1. The Lipschitz constant G will be set to 1 for the ease of exposition.

[4] Comparing the dependence on λ is more subtle, as discussed in Appendix A.1.

Step 1: discrete-time recursive inequality First, let us consider the following inequality that characterizes "admissible" potentials for Algorithm 1. For all t and S,

$$V(t-1,S) \geq \max_{g\in[-1,1]} \left\{ V(t,S-g) + g\bar{\nabla}_S^1 V(t,S) + \lambda \left| \bar{\nabla}_S^1 V(t,S) - \bar{\nabla}_S^1 V(t+1,S-g) \right| \right\}. \tag{5}$$

Finding solutions of this inequality is sufficient for constructing regret bounds. To see this, suppose the above holds for some V. We can then plug in $S = S_{t-1}$ and guarantee that for all $g_t \in [-1,1]$,

$$g_t x_t + \lambda |x_t - x_{t+1}| \leq V(t-1, S_{t-1}) - V(t, S_t).$$

A telescopic sum further leads to a cumulative loss bound $\mathrm{Regret}_T^\lambda(0) \leq V(0,0) - V(T,S_T)$, and a regret bound on $\mathrm{Regret}_T^\lambda(u)$ then follows from the standard loss-regret duality [MO14].

Step 2: ε-scaled recursion Since we ideally need *optimal potential functions* that satisfy the inequality (5) without any slack, let us turn (5) into an *equality* and try to approximately solve it. Intuitively this is a challenging task, as there is no natural way to parameterize the dependence of V on the discrete time t. However, if we decrease the discrete time interval, solutions V will be "smoother" and easier to describe. Concretely, let $\varepsilon > 0$ be a parameter that will later approach 0. On (5), we scale

1. the unit time by ε^2;
2. the loss gradient g, the switching cost weight λ and the gap increment by ε.

Both scaling factors are justified in the switching-free setting [ZCP22b, Appendix A.2]. Notably, since g and λ have the same "unit", it is natural that they are scaled by the same rate. With that, we obtain a *scaled recursion*

$$\begin{aligned} &V(t-\varepsilon^2, S) \\ &\quad = \max_{g\in[-1,1]} \left\{ V(t,S-\varepsilon g) + \varepsilon g \bar{\nabla}_S^\varepsilon V(t,S) + \varepsilon\lambda \left| \bar{\nabla}_S^\varepsilon V(t,S) - \bar{\nabla}_S^\varepsilon V(t+\varepsilon^2, S-\varepsilon g) \right| \right\}. \end{aligned} \tag{6}$$

Step 3: continuous-time PDE To proceed, we take the second-order Taylor approximation on all components of (6). Calculations are simple, and we defer the details to Appendix A.5. Both the zeroth and the first order terms of ε naturally vanish. Only keeping the second order terms, we have

$$\nabla_t V(t,S) + \max_{g\in[-1,1]} \left(\frac{1}{2} g^2 \nabla_{SS} V(t,S) + \lambda \left| g \nabla_{SS} V(t,S) \right| \right) = 0.$$

As typical potential functions are convex in the sufficient statistic S, it is reasonable to impose an additional condition $\nabla_{SS} V(t,S) \geq 0$. Then, the above becomes the 1D *backward heat equation*

$$\nabla_t V + \alpha \nabla_{SS} V = 0,$$

where $\alpha = \lambda + 1/2$. Compared to the switching-free setting [ZCP22b, Eq. 5], we obtain the same PDE, but change the *negative thermal diffusivity* α from $1/2$ to $1/2 + \lambda$. This concisely characterizes the effect of switching costs on the structure of the online learning problem.

Step 4: solving the PDE The final step is to solve the backward heat equation. With a hyperparameter c, consider solutions of the form

$$V_\alpha(t,S) = t^c g\left(\frac{S}{\sqrt{4\alpha t}} \right).$$

Plug it in, the backward heat equation reduces to the Hermite *Ordinary Differential Equation* (ODE)

$$g''(z) - 2z g'(z) + 4c g(z) = 0,$$

which is *independent of* α. This is a crucial observation, as it reveals the correct way to generalize the knowledge from the switching-free setting to the setting with switching costs. More specifically,

- In the switching-free setting, we can take a solution $g(z)$ of the Hermite ODE, plug in the argument $z = S/\sqrt{2t}$ and obtain a potential function V_α.
- When switching costs are considered, the above derivation suggests us to take the *same* function $g(z)$ as before, and plug in a scaled argument $z = S/\sqrt{4\alpha t}$. **This is precisely dual space scaling.**

Finally, as shown in [ZCP22b], a particularly good choice of c is $1/2$. Using this choice yields the switching-adjusted potential (3).

Remark To summarize, through this derivation we aim to demonstrate a key benefit of the continuous-time analysis: it makes the generalization of algorithmic structures easier. This was not presented in our prior work [ZCP22b], but could be useful in the broader online learning context.

Meanwhile, we do not intend to overclaim its strength – although the continuous-time analysis provides useful intuition, we ultimately care about discrete-time regret bounds. Discretizing such arguments relies on an obscure argument that has not been made concrete yet: "V_α derived in the continuous time also serves as a good potential in the discrete time." Indeed, verifying this property is technically nontrivial (Section 2.3), and doing so requires a slightly more conservative choice of α (i.e., $4\lambda + 2$) than what is suggested above.

2.5 Extension beyond the 1D unconstrained setting

So far we have only considered the 1D unconstrained setting. Our results can be extended to higher dimensional domains and bounded domains, which is deferred to Appendix B.

Most notably, we present an algorithm (Algorithm 5) for 1D bounded domain: if the domain has diameter D, then the *switching cost alone* of this algorithm is bounded by $\tilde{O}(D\sqrt{\tau})$ on *any time interval of length* τ. Such a property is crucial in [ZCP22a] for the construction of a *strongly adaptive OCO with memory* algorithm. However, the proof in [ZCP22a] critically relies on hard-thresholding a betting fraction, which, as we demonstrated in Section 2.2, is suboptimal. In contrast, our new result simultaneously achieves this property and the optimal augmented regret bound.

3 LEA with switching cost

Our improved results can also be applied to *LEA with switching cost*, leading to the first comparator adaptive algorithm there. Conversion techniques (from OLO to LEA) without switching costs were studied in [LS15, OP16], and since then, they have become standard tools for the online learning community. Here we present a different view on this conversion problem, based on its connection to the well-known constrained domain reduction [CO18] (surveyed in Appendix B). In particular, it leads to a mechanism for incorporating switching costs, with a clear geometric interpretation.

The setting of *LEA with switching cost* is a special case of the high-dimensional OLO problem (Appendix B). Let d be the number of experts, and we define the domain $\mathcal{X}$ as the d-dimensional probability simplex $\Delta(d)$. Loss gradients g_t satisfy $\|g_t\|_\infty \le G$, and switching costs are measured by the L_1 norm. The performance metric is still the augmented regret, now defined as

$$\mathrm{Regret}_T^\lambda(u) := \sum_{t=1}^{T} \langle g_t, x_t - u\rangle + \lambda \sum_{t=1}^{T-1} \|x_t - x_{t+1}\|_1 .$$

However, the main difference with OLO is the form of comparator adaptive bounds – here we aim at $\mathrm{Regret}_T^\lambda(u) = O(\sqrt{T \cdot \mathrm{KL}(u\|\pi)})$, where $\pi \in \Delta(d)$ is a prior chosen at the beginning. Achieving such a root KL bound relies on techniques different from the OLO setting.

Existing approaches [LS15, OP16] have the following procedure. Given a 1D OLO algorithm that predicts on $\mathbb{R}_+$, independent copies are created for each coordinate and updated using certain surrogate losses. A meta-algorithm queries the coordinate-wise predictions $\{w_{t,i}; i \in [1:d]\}$, collects them into a weight vector $w_t = [w_{t,1}, \dots, w_{t,d}]$, and finally predicts the scaled weight $x_t = w_t / \|w_t\|_1$ on $\Delta(d)$. Despite its general success, such an approach has a discontinuity problem when switching costs are incorporated – if two consecutive weights w_t and w_{t+1} are both close to the origin, then simply scaling them to $\Delta(d)$ can lead to a large switching cost, even when $\|w_t - w_{t+1}\|_1$ is small. This problem is exacerbated by the typical setting[5] of $w_1 = 0$, due to the associated analysis. A graphical demonstration is provided in Figure 1 (Left).

In contrast, our solution is based on *a unified view* of the LEA-OLO reduction and the constrained domain reduction [CO18]. Starting without switching costs, we observe that the general Banach version of the latter can also convert OLO to LEA, therefore specialized techniques are not required

[5] When $w_t = 0$, x_t can be arbitrary on $\Delta(d)$ by definition. However, as w_t changes continuously w.r.t. the observed information, it could hover around 0 at some point, thus experiencing the sketched problem.

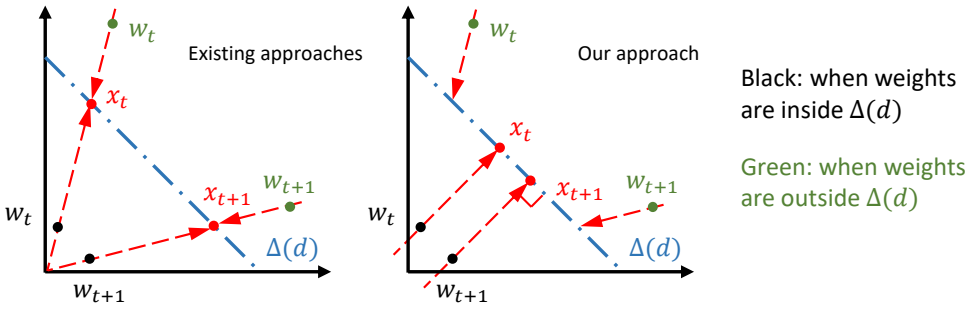

Figure 1: Switching costs in LEA-OLO reductions. Left: existing approaches. Right: ours, where the projection of w_t contains two cases. (i) $\|w_t\|_1 \geq 1$, shown in green; (ii) $\|w_t\|_1 < 1$, shown in black.

for this task. Algorithmically, we set $x_t \in \arg\min_{x \in \Delta(d)} \|x - w_t\|_1$ as opposed to $x_t = w_t / \|w_t\|_1$. The surrogate losses for the base algorithms are also different, which we elaborate in Appendix C.3.

A major benefit of this unified view is the non-uniqueness of the L_1 norm projection – if $\|w_t\| < 1$, then any $x_t \in \Delta(d)$ satisfying $\{x_{t,i} \geq w_{t,i}; \forall i\}$ minimizes $\|x - w_t\|_1$ on $\Delta(d)$. This brings more flexibility to the algorithm design. Specifically, we adopt

1. the orthogonal projection $x_t = w_t + d^{-1}(1 - \|w_t\|_1)$ when $\|w_t\|_1 \leq 1$;
2. the scaling $x_t = w_t / \|w_t\|_1$ when $\|w_t\|_1 > 1$.

The orthogonal projection is better for controlling switching costs, as shown in Figure 1 (Right). Concretely, this leads to the first comparator adaptive algorithm for LEA with switching costs.

Theorem 2. *For LEA with switching cost, given any prior π in the relative interior of $\Delta(d)$, Algorithm 7 from Appendix C.2 guarantees*

$$\sum_{t=1}^{T} \langle g_t, x_t - u \rangle + \lambda \sum_{t=1}^{T-1} \|x_t - x_{t+1}\|_1 = \left[\sqrt{\mathrm{TV}(u||\pi) \cdot \mathrm{KL}(u||\pi)} + 1 \right] \cdot O\left(\sqrt{(\lambda G + G^2)T} \right),$$

for all $u \in \Delta(d)$ and $T \in \mathbb{N}_+$.

We emphasize two strengths of this bound.

1. Since it is comparator adaptive, such a bound only implicitly depends on d through the divergence term $\sqrt{\mathrm{TV} \cdot \mathrm{KL}}$. In favorable cases we may have a good prior π such that $\mathrm{TV}(u||\pi) \cdot \mathrm{KL}(u||\pi) = O(1)$; this will save us a $\sqrt{\log d}$ factor compared to minimax algorithms (with switching costs), such as *Follow the Lazy Leader* [KV05] and *Shrinking Dartboard* [GVW10].
2. Even without switching costs, we improve the $\sqrt{\mathrm{KL}}$ divergence term in existing comparator adaptive bounds [CFH09, LS15, OP16] to $\sqrt{\mathrm{TV} \cdot \mathrm{KL}}$. The latter is better since (i) TV is always less than 1, and (ii) there exist $p, q \in \Delta(d)$ such that $\mathrm{TV}(p||q) \cdot \mathrm{KL}(p||q) \leq 1$ but $\mathrm{KL}(p||q) \geq \sqrt{\log d} - o(1)$ (cf. Appendix C.3). In other words, compared to $\sqrt{\mathrm{KL}}$, the $\sqrt{\mathrm{TV} \cdot \mathrm{KL}}$ bound is never worse (up to constants), and can save at least a $(\log d)^{1/4}$ factor in certain cases. Generalizations of root KL to *f-divergences* have been considered in [Alq21, NBC+21], but to our knowledge, no existing algorithm guarantees a better divergence term than root KL.

Experiment We complement our theoretical results by experiments on a portfolio selection task. Due to limited space, it is presented in Appendix D.

Conclusion We defer discussions on the conclusion, limitations and future works to Appendix E.

Acknowledgments and Disclosure of Funding

We thank the anonymous reviewers for their feedback. This research was partially supported by the NSF under grants IIS-1914792, DMS-1664644, and CNS-1645681, by the ONR under grants N00014-19-1-2571 and N00014-21-1-2844, by the DOE under grants DE-AR-0001282 and DE-EE0009696, by the NIH under grants R01 GM135930 and UL54 TR004130, and by Boston University.

References

[ABH+19] Naman Agarwal, Brian Bullins, Elad Hazan, Sham Kakade, and Karan Singh. Online control with adversarial disturbances. In *International Conference on Machine Learning*, pages 111–119. PMLR, 2019.

[ABL+13] Lachlan Andrew, Siddharth Barman, Katrina Ligett, Minghong Lin, Adam Meyerson, Alan Roytman, and Adam Wierman. A tale of two metrics: Simultaneous bounds on competitiveness and regret. In *Conference on Learning Theory*, pages 741–763. PMLR, 2013.

[AHM15] Oren Anava, Elad Hazan, and Shie Mannor. Online learning for adversaries with memory: price of past mistakes. *Advances in Neural Information Processing Systems*, 28, 2015.

[Alq21] Pierre Alquier. Non-exponentially weighted aggregation: Regret bounds for unbounded loss functions. In *International Conference on Machine Learning*, pages 207–218. PMLR, 2021.

[AT18] Jason Altschuler and Kunal Talwar. Online learning over a finite action set with limited switching. In *Conference On Learning Theory*, pages 1569–1573. PMLR, 2018.

[BCKP21] Aditya Bhaskara, Ashok Cutkosky, Ravi Kumar, and Manish Purohit. Power of hints for online learning with movement costs. In *International Conference on Artificial Intelligence and Statistics*, pages 2818–2826. PMLR, 2021.

[BEZ20a] Erhan Bayraktar, Ibrahim Ekren, and Xin Zhang. Finite-time 4-expert prediction problem. *Communications in Partial Differential Equations*, 45(7):714–757, 2020.

[BEZ20b] Erhan Bayraktar, Ibrahim Ekren, and Yili Zhang. On the asymptotic optimality of the comb strategy for prediction with expert advice. *The Annals of Applied Probability*, 30(6):2517–2546, 2020.

[BK99] Avrim Blum and Adam Kalai. Universal portfolios with and without transaction costs. *Machine Learning*, 35(3):193–205, 1999.

[BLLS19] Sébastien Bubeck, Yin Tat Lee, Yuanzhi Li, and Mark Sellke. Competitively chasing convex bodies. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 861–868, 2019.

[CBDS13] Nicolo Cesa-Bianchi, Ofer Dekel, and Ohad Shamir. Online learning with switching costs and other adaptive adversaries. *Advances in Neural Information Processing Systems*, 26, 2013.

[CBL06] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[CFH09] Kamalika Chaudhuri, Yoav Freund, and Daniel J Hsu. A parameter-free hedging algorithm. *Advances in neural information processing systems*, 22, 2009.

[CGW18] Niangjun Chen, Gautam Goel, and Adam Wierman. Smoothed online convex optimization in high dimensions via online balanced descent. In *Conference On Learning Theory*, pages 1574–1594. PMLR, 2018.

[CLO22] Keyi Chen, John Langford, and Francesco Orabona. Better parameter-free stochastic optimization with ode updates for coin-betting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6239–6247, 2022.

[CLW21] Liyu Chen, Haipeng Luo, and Chen-Yu Wei. Impossible tuning made possible: A new expert algorithm and its applications. In *Conference on Learning Theory*, pages 1216–1259. PMLR, 2021.

[CO96] Thomas M Cover and Erik Ordentlich. Universal portfolios with side information. *IEEE Transactions on Information Theory*, 42(2):348–363, 1996.

[CO18] Ashok Cutkosky and Francesco Orabona. Black-box reductions for parameter-free online learning in banach spaces. In *Conference On Learning Theory*, pages 1493–1529. PMLR, 2018.

[Cov91] Thomas M Cover. Universal portfolios. *Mathematical Finance*, 1(1):1–29, 1991.

[Cut19] Ashok Cutkosky. Combining online learning guarantees. In *Conference on Learning Theory*, pages 895–913. PMLR, 2019.

[Cut20] Ashok Cutkosky. Parameter-free, dynamic, and strongly-adaptive online learning. In *International Conference on Machine Learning*, pages 2250–2259. PMLR, 2020.

[CV10] Alexey Chernov and Vladimir Vovk. Prediction with advice of unknown number of experts. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 117–125, 2010.

[CYLK20] Lin Chen, Qian Yu, Hannah Lawrence, and Amin Karbasi. Minimax regret of switching-constrained online convex optimization: No phase transition. *Advances in Neural Information Processing Systems*, 33:3477–3486, 2020.

[DGSS15] Amit Daniely, Alon Gonen, and Shai Shalev-Shwartz. Strongly adaptive online learning. In *International Conference on Machine Learning*, pages 1405–1411. PMLR, 2015.

[DHS11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[DK20] Nadejda Drenska and Robert V Kohn. Prediction with expert advice: A PDE perspective. *Journal of Nonlinear Science*, 30(1):137–173, 2020.

[DM19] Amit Daniely and Yishay Mansour. Competitive ratio vs regret minimization: achieving the best of both worlds. In *Algorithmic Learning Theory*, pages 333–368. PMLR, 2019.

[Doc16] Robert Dochow. *Online algorithms for the portfolio selection problem*. Springer, 2016.

[Doo84] Joseph L Doob. *Classical potential theory and its probabilistic counterpart*. Springer, 1984.

[FRS15] Dylan J Foster, Alexander Rakhlin, and Karthik Sridharan. Adaptive online learning. *Advances in Neural Information Processing Systems*, 28:3375–3383, 2015.

[FRS18] Dylan J Foster, Alexander Rakhlin, and Karthik Sridharan. Online learning: Sufficient statistics and the Burkholder method. In *Conference On Learning Theory*, pages 3028–3064. PMLR, 2018.

[Fuj08] Takahiko Fujita. A random walk analogue of Lévy's theorem. *Studia scientiarum mathematicarum Hungarica*, 45(2):223–233, 2008.

[GHP22] Laura Greenstreet, Nicholas JA Harvey, and Victor Sanches Portella. Efficient and optimal fixed-time regret with two experts. In *International Conference on Algorithmic Learning Theory*, pages 436–464. PMLR, 2022.

[GLSW19] Gautam Goel, Yiheng Lin, Haoyuan Sun, and Adam Wierman. Beyond online balanced descent: An optimal algorithm for smoothed online optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

[Gof14] Eyal Gofer. Higher-order regret bounds with switching costs. In *Conference on Learning Theory*, pages 210–243. PMLR, 2014.

[GVW10] Sascha Geulen, Berthold Vöcking, and Melanie Winkler. Regret minimization for online buffering problems using the weighted majority algorithm. In *Conference on Learning Theory*, pages 132–143, 2010.

[Haz16] Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[HLPR20] Nicholas JA Harvey, Christopher Liaw, Edwin A Perkins, and Sikander Randhawa. Optimal anytime regret for two experts. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1404–1415. IEEE, 2020.

[HSSW98] David P Helmbold, Robert E Schapire, Yoram Singer, and Manfred K Warmuth. On-line portfolio selection using multiplicative updates. *Mathematical Finance*, 8(4):325–347, 1998.

[JC22] Andrew Jacobsen and Ashok Cutkosky. Parameter-free mirror descent. In *Conference on Learning Theory*, pages 4160–4211. PMLR, 2022.

[JO19] Kwang-Sung Jun and Francesco Orabona. Parameter-free locally differentially private stochastic subgradient descent. *arXiv preprint arXiv:1911.09564*, 2019.

[KKW20] Vladimir A Kobzar, Robert V Kohn, and Zhilei Wang. New potential-based bounds for prediction with expert advice. In *Conference on Learning Theory*, pages 2370–2405. PMLR, 2020.

[Kle13] Achim Klenke. *Probability theory: a comprehensive course*. Springer Science & Business Media, 2013.

[KS10] Robert V Kohn and Sylvia Serfaty. A deterministic-control-based approach to fully nonlinear parabolic and elliptic equations. *Communications on pure and applied mathematics*, 63(10):1298–1350, 2010.

[Kud82] R Kudžma. Ito's formula for a random walk. *Lithuanian Mathematical Journal*, 22(3):302–306, 1982.

[KV02] Adam Tauman Kalai and Santosh Vempala. Efficient algorithms for universal portfolios. *Journal of Machine Learning Research*, pages 423–440, 2002.

[KV05] Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

[KVE15] Wouter M Koolen and Tim Van Erven. Second-order quantile methods for experts and combinatorial games. In *Conference on Learning Theory*, pages 1155–1175. PMLR, 2015.

[LH14] Bin Li and Steven CH Hoi. Online portfolio selection: A survey. *ACM Computing Surveys (CSUR)*, 46(3):1–36, 2014.

[LQL20] Yingying Li, Guannan Qu, and Na Li. Online optimization with predictions and switching costs: Fast algorithms and the fundamental limit. *IEEE Transactions on Automatic Control*, 66(10):4761–4768, 2020.

[LS15] Haipeng Luo and Robert E Schapire. Achieving all with no parameters: Adanormalhedge. In *Conference on Learning Theory*, pages 1286–1304. PMLR, 2015.

[LWZ18] Haipeng Luo, Chen-Yu Wei, and Kai Zheng. Efficient online portfolio with logarithmic regret. *Advances in Neural Information Processing Systems*, 31, 2018.

[MK20] Zakaria Mhammedi and Wouter M Koolen. Lipschitz and comparator-norm adaptivity in online learning. In *Conference on Learning Theory*, pages 2858–2887. PMLR, 2020.

[MO14] H Brendan McMahan and Francesco Orabona. Unconstrained online linear learning in hilbert spaces: Minimax algorithms and normal approximations. In *Conference on Learning Theory*, pages 1020–1039. PMLR, 2014.

[MR22] Zakaria Mhammedi and Alexander Rakhlin. Damped Online Newton Step for portfolio selection. In *Conference on Learning Theory*, pages 5561–5595. PMLR, 2022.

[NBC+21] Jeffrey Negrea, Blair Bilodeau, Nicolò Campolongo, Francesco Orabona, and Dan Roy. Minimax optimal quantile and semi-adversarial regret via root-logarithmic regularizers. *Advances in Neural Information Processing Systems*, 34, 2021.

[OLL17] Laurent Orseau, Tor Lattimore, and Shane Legg. Soft-bayes: Prod for mixtures of experts with log-loss. In *International Conference on Algorithmic Learning Theory*, pages 372–399. PMLR, 2017.

[OP16] Francesco Orabona and Dávid Pál. Coin betting and parameter-free online learning. *Advances in Neural Information Processing Systems*, 29, 2016.

[Ora13] Francesco Orabona. Dimension-free exponentiated gradient. *Advances in Neural Information Processing Systems*, 26, 2013.

[Ora19] Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

[OT17] Francesco Orabona and Tatiana Tommasi. Training deep networks without learning rates through coin betting. *Advances in Neural Information Processing Systems*, 30:2160–2170, 2017.

[PLH22] Victor Sanches Portella, Christopher Liaw, and Nicholas JA Harvey. Continuous prediction with experts' advice. *arXiv preprint arXiv:2206.00236*, 2022.

[Rok17] Dmitry B Rokhlin. PDE approach to the problem of online prediction with expert advice: a construction of potential-based strategies. *arXiv preprint arXiv:1705.01091*, 2017.

[Sel20] Mark Sellke. Chasing convex bodies optimally. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1509–1518. SIAM, 2020.

[Sim20] Max Simchowitz. Making non-stochastic control (almost) as easy as stochastic. *Advances in Neural Information Processing Systems*, 33:18318–18329, 2020.

[SK21] Uri Sherman and Tomer Koren. Lazy OCO: Online convex optimization on a switching budget. In *Conference on Learning Theory*, pages 3972–3988. PMLR, 2021.

[SM12] Matthew Streeter and Brendan Mcmahan. No-regret algorithms for unconstrained online convex optimization. *Advances in Neural Information Processing Systems*, 25, 2012.

[SS11] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and trends in Machine Learning*, 4(2):107–194, 2011.

[SSH20] Max Simchowitz, Karan Singh, and Elad Hazan. Improper learning for non-stochastic control. In *Conference on Learning Theory*, pages 3320–3436. PMLR, 2020.

[vdH19] Dirk van der Hoeven. User-specified local differential privacy in unconstrained adaptive online learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[WWYZ21] Guanghui Wang, Yuanyu Wan, Tianbao Yang, and Lijun Zhang. Online convex optimization with continuous switching constraint. *Advances in Neural Information Processing Systems*, 34, 2021.

[ZAK22] Julian Zimmert, Naman Agarwal, and Satyen Kale. Pushing the efficiency-regret pareto frontier for online learning of portfolios and quantum states. In *Conference on Learning Theory*, pages 182–226. PMLR, 2022.

[ZCP22a] Zhiyu Zhang, Ashok Cutkosky, and Ioannis Paschalidis. Adversarial tracking control via strongly adaptive online learning with memory. In *International Conference on Artificial Intelligence and Statistics*, pages 8458–8492. PMLR, 2022.

[ZCP22b] Zhiyu Zhang, Ashok Cutkosky, and Ioannis Paschalidis. PDE-based optimal strategy for unconstrained online learning. In *International Conference on Machine Learning*, pages 26085–26115. PMLR, 2022.

[Zhu14] Kangping Zhu. *Two problems in applications of PDE*. PhD thesis, New York University, 2014.

[Zin03] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pages 928–936, 2003.

[ZJLY21] Lijun Zhang, Wei Jiang, Shiyin Lu, and Tianbao Yang. Revisiting smoothed online learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Appendix E.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] The present work is mainly theoretical.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] In supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix D.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] Our experiments are not computationally demanding.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix

Organization Appendix A presents omitted details of our 1D OLO results. Section B extends such results to more general OLO settings. Section C presents details on LEA. Section D numerically tests our approach in a portfolio selection problem. Limitations and future works are discussed in Section E.

A Details on 1D OLO

This section presents detailed discussions and omitted proofs for our 1D unconstrained OLO algorithm.

A.1 The suboptimal algorithm from [ZCP22a]

Let us start by summarizing the existing result from our prior work [ZCP22a, Algorithm 1], which is the first comparator adaptive algorithm for 1D unconstrained OLO with switching costs. The original version in [ZCP22a] considers a bounded domain and an extra regularization term, which are removed below for a clear comparison. Π denotes the absolute value projection.

Algorithm 2 The suboptimal algorithm from [ZCP22a]

Require: A hyperparameter $C > 0$, the Lipschitz constant G, and the switching cost weight λ.

1: Define $K = G + \lambda$. Initialize internal variables as $\mathrm{Wealth}_0 = C \cdot K$, and $\beta_1, x_1 = 0$.
2: **for** $t = 1, 2, \ldots$ **do**
3: Make a prediction x_t, observe a loss gradient g_t.
4: Define an *unprojected* betting fraction as $\hat{\beta}_{t+1} = -\sum_{i=1}^{t} g_i/(2K^2 t)$.
5: Define a hard threshold for the betting fraction, $\mathcal{B}_{t+1} = [-1/(K\sqrt{2t}), 1/(K\sqrt{2t})]$.
6: Update the *projected* betting fraction as $\beta_{t+1} = \Pi_{\mathcal{B}_{t+1}}(\hat{\beta}_{t+1})$.
7: Assign Wealth_t as the solution to the following equation (uniqueness can be proved),

$$\mathrm{Wealth}_t = (1 - g_t\beta_t)\mathrm{Wealth}_{t-1} - \lambda|\beta_t\mathrm{Wealth}_{t-1} - \beta_{t+1}\mathrm{Wealth}_t|. \tag{7}$$

8: Calculate the next prediction, $x_{t+1} = \beta_{t+1}\mathrm{Wealth}_t$.
9: **end for**

Both the algorithm and its analysis are analogous to [CO18], which recasts the selection of betting fractions as an "inner" online learning problem. Nonetheless, incorporating switching costs requires extra components (Line 5-7), making the whole analysis nontrivial.

Theorem 3 (Theorem 1 of [ZCP22a], adapted)**.** *Algorithm 2 guarantees for all $T \in \mathbb{N}_+$ and $u \in \mathbb{R}$,*

$$\mathrm{Regret}_T^{\lambda}(u) \le (G + \lambda)\left[C + |u|\sqrt{2T}\left(\frac{3}{2} + \log\frac{\sqrt{2}\,|u|\,T^{5/2}}{C}\right)\right].$$

For now, let us only consider the dependence on u and T. Compared to typical results on comparator adaptivity, the above bound has two limitations. First, the bound does not achieve the optimal loss-regret trade-off [ZCP22a] – the constraint $\mathrm{Regret}_T^{\lambda}(0) \le O(1)$ on the *cumulative loss* of the algorithm is too harsh, such that the leading regret term ($\mathrm{Regret}_T^{\lambda}(u)$ with large $|u|$) suffers a logarithmic penalty on T (relative to the usual $O(\sqrt{T})$ minimax rate). Second, even if we only consider this particular loss-regret trade-off, i.e., $\mathrm{Regret}_T^{\lambda}(0) \le O(1)$, the logarithmic terms are not optimal (being outside the square root). In other words, the bound is not Pareto-optimal. The present paper simultaneously improves these two suboptimalities.

On a separate note, let us consider its dependence on G and λ, which is more subtle.[6] In its vanilla form, the above bound has the leading term $\tilde{O}(\max\{G, \lambda\}\,|u|\,\sqrt{T})$, but we can run a meta-algorithm

[6] We thank the NeurIPS reviewer KR3f for insightful comments.

[ZCP22a, Algorithm 3] on the top to improve it to $\tilde{O}\left(|u|\sqrt{\max\{G,\lambda\}\sum_{t=1}^{T}|g_t|}\right)$. The pseudo-code is presented as Algorithm 3. Its main idea is to adaptively slow down the update of the base algorithm, depending on the observed gradients.

Algorithm 3 Meta-algorithm [ZCP22a, Algorithm 3], adapted

Require: The Lipschitz constant G and the switching cost weight λ.
1: Initialize a base algorithm $\mathcal{A}$ as a copy of Algorithm 2.
2: Initialize $i = 1$ and an accumulator $Z_i = 0$. Query the first output of $\mathcal{A}$ and assign it to w_i.
3: **for** $t = 1, 2, \ldots$ **do**
4: Predict $x_t \leftarrow w_i$, observe g_t, let $Z_i \leftarrow Z_i + g_t$.
5: **if** $\|Z_i\| > \max\{\lambda, G\}$ **then**
6: Send Z_i to $\mathcal{A}$ as the i-th loss. Let $i \leftarrow i + 1$.
7: Set $Z_i = 0$. Query the i-th output of $\mathcal{A}$ and assign it to w_i.
8: **end if**
9: **end for**

Theorem 4 (Theorem 6 of [ZCP22a], adapted)**.** *Algorithm 3 guarantees for all $T \in \mathbb{N}_+$ and $u \in \mathbb{R}$,*

$$\mathrm{Regret}_T^{\lambda}(u) \le (G+\lambda)C + |u|\,\tilde{O}\left(\max\{G,\lambda\} + \sqrt{\max\{G,\lambda\}\sum_{t=1}^{T}|g_t|}\right).$$

When λ is large, the $O(\sqrt{\lambda})$ rate on the leading term is optimal. Moreover, in the presence of switching costs, the gradient adaptivity term $\sqrt{\sum_{t=1}^{T}|g_t|}$ is a strong one, since second-order gradient adaptivity $\sqrt{\sum_{t=1}^{T}|g_t|^2}$ is not achievable [Gof14]. Note that we can also run this meta-algorithm on top of our improved base algorithm (Algorithm 1), such that the latter achieves gradient adaptivity as well. Due to this reason, when comparing the results of the present work to [ZCP22a], we mostly leave the dependence on λ and the gradient adaptivity out of the comparison.

A.2 A few basic lemmas

Before proving our main result (Theorem 1), we present a few basic lemmas on Algorithm 1 and the potential function V_α (3), which will be useful later on. The first lemma shows the monotonicity of the discrete derivative strategy, which is quite intuitive.

Lemma A.1. *If the potential $V(t,S)$ is even and convex in S, then $\bar{\nabla}_S V(t,S)$ is odd and monotonically increasing in S.*

Proof of Lemma A.1. $\bar{\nabla}_S V(t,S)$ is odd due to the simple relation

$$\begin{aligned}\bar{\nabla}_S V(t,-S) &= \frac{1}{2}\left[V(t,-S+1) - V(t,-S-1)\right] \\ &= \frac{1}{2}\left[V(t,S-1) - V(t,S+1)\right] = -\bar{\nabla}_S V(t,S).\end{aligned}$$

As for the monotonicity, it is equivalent to showing for all $\delta \ge 0$,

$$V(t,S+1+\delta) - V(S-1+\delta) \ge V(t,S+1) - V(S-1).$$

This follows from the convexity of $V(t,\cdot)$, as

$$\begin{aligned}V(t,S+1) &\le \frac{2}{2+\delta}V(t,S+1+\delta) + \frac{\delta}{2+\delta}V(t,S-1), \\ V(t,S-1+\delta) &\le \frac{\delta}{2+\delta}V(t,S+1+\delta) + \frac{2}{2+\delta}V(t,S-1).\end{aligned}$$

□

Now, for the potential function V_α, we compute its continuous partial derivatives. The proof is straightforward calculation, therefore omitted.

Lemma A.2. *For any $\alpha > 0$, V_α defined in (3) is even and convex. Moreover,*

$$\nabla_S V_\alpha(t,S) = C\int_0^{S/\sqrt{4\alpha t}} \exp\left(x^2\right) dx, \qquad \nabla_{SSS} V_\alpha(t,S) = \frac{CS}{4(\alpha t)^{3/2}} \exp\left(\frac{S^2}{4\alpha t}\right),$$

$$\nabla_{SS} V_\alpha(t,S) = \frac{C}{2\sqrt{\alpha t}} \exp\left(\frac{S^2}{4\alpha t}\right), \qquad \nabla_t V_\alpha(t,S) = -\frac{C\sqrt{\alpha}}{2\sqrt{t}} \exp\left(\frac{S^2}{4\alpha t}\right).$$

Based on the above, the discrete derivative $\bar{\nabla}_S V_\alpha$ has the following properties.

Lemma A.3. *For all $\alpha > 0$, $t \geq 0$ and $S \geq 0$,*

1. *$\bar{\nabla}_S V_\alpha(t,S)$ as a function of t is decreasing and convex;*
2. *$\bar{\nabla}_S V_\alpha(t,S)$ as a function of S is convex.*

Proof of Lemma A.3. Considering the first part of the lemma,

$$\begin{aligned}\nabla_t \left[\bar{\nabla}_S V_\alpha(t,S)\right] &= \frac{1}{2}\left[\nabla_t V_\alpha(t,S+1) - \nabla_t V_\alpha(t,S-1)\right] \\ &= -\frac{C\sqrt{\alpha}}{4\sqrt{t}} \exp\left(\frac{S^2+1}{4\alpha t}\right) \sinh\left(\frac{S}{2\alpha t}\right),\end{aligned}$$

which, when $S \geq 0$, is negative and increasing in t. Therefore, $\bar{\nabla}_S V_\alpha(t,S)$ as a function of t is decreasing and convex. Similarly,

$$\nabla_S \left[\bar{\nabla}_S V_\alpha(t,S)\right] = \frac{1}{2}\left[\nabla_S V_\alpha(t,S+1) - \nabla_S V_\alpha(t,S-1)\right] = \frac{C}{2}\int_{(S-1)/\sqrt{4\alpha t}}^{(S+1)/\sqrt{4\alpha t}} \exp\left(x^2\right) dx,$$

which is increasing in S. Therefore, $\bar{\nabla}_S V_\alpha(t,S)$ as a function of S is convex. □

A.3 Proof of Theorem 1

In this subsection, we prove Theorem 1, the regret bound of our 1D OLO algorithm with switching costs. As sketched in Section 2.3, our proof relies on two important lemmas, Lemma 2.2 and 2.3. We prove them first.

Lemma 2.2. *For all $\alpha > 0$, consider Algorithm 1 induced by the potential V_α. For all $t \in \mathbb{N}_+$,*

$$|x_t - x_{t+1}| \leq \bar{\nabla}_S V_\alpha(t, S_{t-1}+1) - \bar{\nabla}_S V_\alpha(t, S_{t-1}-1).$$

Proof of Lemma 2.2. First, since $\bar{\nabla}_S V_\alpha(t,S)$ is monotonic in S due to Lemma A.1, we have

$$\begin{aligned}|x_t - x_{t+1}| &= \left|\bar{\nabla}_S V_\alpha(t, S_{t-1}) - \bar{\nabla}_S V_\alpha(t+1, S_t)\right| \\ &\leq \max_{c=\pm 1} \left|\bar{\nabla}_S V_\alpha(t, S_{t-1}) - \bar{\nabla}_S V_\alpha(t+1, S_{t-1}+c)\right|.\end{aligned}$$

For clarity, from the RHS we define

$$f(t,S) := \max_{c=\pm 1} \left|\bar{\nabla}_S V_\alpha(t,S) - \bar{\nabla}_S V_\alpha(t+1, S+c)\right|.$$

It is even in S, as

$$\begin{aligned}f(t,-S) &= \max_{c=\pm 1} \left|\bar{\nabla}_S V_\alpha(t,-S) - \bar{\nabla}_S V_\alpha(t+1,-S+c)\right| \\ &= \max_{c=\pm 1} \left|-\bar{\nabla}_S V_\alpha(t,S) + \bar{\nabla}_S V_\alpha(t+1,S-c)\right| && \text{(Lemma A.1)} \\ &= \max_{c=\pm 1} \left|\bar{\nabla}_S V_\alpha(t,S) - \bar{\nabla}_S V_\alpha(t+1,S-c)\right| = f(t,S).\end{aligned}$$

Therefore, we can restrict the rest of the proof to $S \geq 0$, and the remaining task is to upper bound $f(t,S)$ for all $0 \leq S \leq t-1$.

From Lemma A.1 and A.3,

$$\bar{\nabla}_S V_\alpha(t+1, S-1) \leq \bar{\nabla}_S V_\alpha(t+1, S) \leq \bar{\nabla}_S V_\alpha(t, S),$$

$$\bar{\nabla}_S V_\alpha(t+1, S-1) \le \bar{\nabla}_S V_\alpha(t+1, S+1).$$

Therefore, if $\bar{\nabla}_S V_\alpha(t+1, S-1) \le \bar{\nabla}_S V_\alpha(t, S) \le \bar{\nabla}_S V_\alpha(t+1, S+1)$, then

$$\begin{aligned} f(t,S) &= \max\left\{\left|\bar{\nabla}_S V_\alpha(t,S) - \bar{\nabla}_S V_\alpha(t+1,S-1)\right|, \left|\bar{\nabla}_S V_\alpha(t,S) - \bar{\nabla}_S V_\alpha(t+1,S+1)\right|\right\} \\ &\le \bar{\nabla}_S V_\alpha(t+1,S+1) - \bar{\nabla}_S V_\alpha(t+1,S-1). \end{aligned}$$

On the other hand, if $\bar{\nabla}_S V_\alpha(t+1, S+1) \le \bar{\nabla}_S V_\alpha(t, S)$, then

$$f(t,S) = \bar{\nabla}_S V_\alpha(t,S) - \bar{\nabla}_S V_\alpha(t+1,S-1).$$

Combining the above,

$$\begin{aligned} &f(t,S) \\ &\le \max\left\{\bar{\nabla}_S V_\alpha(t,S) - \bar{\nabla}_S V_\alpha(t+1,S-1), \bar{\nabla}_S V_\alpha(t+1,S+1) - \bar{\nabla}_S V_\alpha(t+1,S-1)\right\}. \end{aligned}$$

Our goal next is to upper bound $f(t,S)$ by $\bar{\nabla}_S V_\alpha(t,S+1) - \bar{\nabla}_S V_\alpha(t,S-1)$, which can be divided into two cases.

Case 1 We aim to show that

$$\bar{\nabla}_S V_\alpha(t,S) - \bar{\nabla}_S V_\alpha(t+1,S-1) \le \bar{\nabla}_S V_\alpha(t,S+1) - \bar{\nabla}_S V_\alpha(t,S-1),$$

which is equivalent to

$$\bar{\nabla}_S V_\alpha(t,S-1) - \bar{\nabla}_S V_\alpha(t+1,S-1) \le \bar{\nabla}_S V_\alpha(t,S+1) - \bar{\nabla}_S V_\alpha(t,S). \tag{8}$$

Note that this trivially holds when $0 \le S < 1$: due to Lemma A.3, the RHS is always positive; however, the LHS is negative due to $\bar{\nabla}_S V_\alpha(t,S-1)$ being increasing in t (Lemma A.1 and A.3 Part 1). Therefore, we only need to show (8) for all $S \ge 1$.

To this end, with $S \ge 1$, we apply the convexity of $\bar{\nabla}_S V_\alpha$ from Lemma A.3:

$$\bar{\nabla}_S V_\alpha(t,S+1) - \bar{\nabla}_S V_\alpha(t,S) \ge \nabla_S\left[\bar{\nabla}_S V_\alpha(t,S)\right],$$

$$\bar{\nabla}_S V_\alpha(t,S-1) - \bar{\nabla}_S V_\alpha(t+1,S-1) \le -\nabla_t\left[\bar{\nabla}_S V_\alpha(t,S-1)\right].$$

Consequently, it suffices to show that

$$-\nabla_t\left[\bar{\nabla}_S V_\alpha(t,S-1)\right] \le \nabla_S\left[\bar{\nabla}_S V_\alpha(t,S)\right].$$

Now it is time to invoke the specific form of V_α. We may reuse $\nabla_S\left[\bar{\nabla}_S V_\alpha(t,S)\right]$ and $\nabla_t\left[\bar{\nabla}_S V_\alpha(t,S)\right]$ calculated from the proof of Lemma A.3.

$$\nabla_S\left[\bar{\nabla}_S V_\alpha(t,S)\right] = \frac{C}{2}\int_{(S-1)/\sqrt{4\alpha t}}^{(S+1)/\sqrt{4\alpha t}} \exp\left(x^2\right) dx \ge \frac{C}{2\sqrt{\alpha t}}\exp\left(\frac{S^2}{4\alpha t}\right),$$

and for all $1 \le S \le t-1$,

$$\begin{aligned} -\nabla_t\left[\bar{\nabla}_S V_\alpha(t,S-1)\right] &= \frac{C\sqrt{\alpha}}{4\sqrt{t}}\exp\left(\frac{(S-1)^2+1}{4\alpha t}\right)\sinh\left(\frac{S-1}{2\alpha t}\right) \\ &= \frac{C\sqrt{\alpha}}{8\sqrt{t}}\exp\left(\frac{S^2}{4\alpha t}\right)\left[1-\exp\left(\frac{-S+1}{\alpha t}\right)\right] \\ &\le \frac{C\sqrt{\alpha}}{8\sqrt{t}}\exp\left(\frac{S^2}{4\alpha t}\right)\left[1-\exp\left(-\frac{1}{\alpha}\right)\right] && (S-1 \le t) \\ &\le \frac{C}{8\sqrt{\alpha t}}\exp\left(\frac{S^2}{4\alpha t}\right). && (\exp(x) \ge x+1) \end{aligned}$$

Therefore, $-\nabla_t\left[\bar{\nabla}_S V_\alpha(t,S-1)\right] \le \nabla_S\left[\bar{\nabla}_S V_\alpha(t,S)\right]$, which proves (8) and concludes Case 1.

Case 2 We aim to show that

$$\bar{\nabla}_S V_\alpha(t+1,S+1) - \bar{\nabla}_S V_\alpha(t+1,S-1) \le \bar{\nabla}_S V_\alpha(t,S+1) - \bar{\nabla}_S V_\alpha(t,S-1).$$

This is straightforward, as

$$\begin{aligned}
& \nabla_t \left[\bar{\nabla}_S V_\alpha(t,S+1) - \bar{\nabla}_S V_\alpha(t,S-1)\right] \\
= & \frac{1}{2}\left[\nabla_t V_\alpha(t,S+2) + \nabla_t V_\alpha(t,S-2) - 2\nabla_t V_\alpha(t,S)\right] \\
= & -\frac{C\sqrt{\alpha}}{4\sqrt{t}}\left[\exp\left(\frac{(S+2)^2}{4\alpha t}\right) + \exp\left(\frac{(S-2)^2}{4\alpha t}\right) - 2\exp\left(\frac{S^2}{4\alpha t}\right)\right] \\
\le & \ 0. \qquad \text{(convexity)}
\end{aligned}$$

Combining the two cases, we can upper bound $f(t,S)$ by $\bar{\nabla}_S V_\alpha(t,S+1) - \bar{\nabla}_S V_\alpha(t,S-1)$, which completes the proof. □

Next, we present the proof of Lemma 2.3, which bounds the residual term Δ_t defined in (4).

Lemma 2.3. *If $\alpha \ge 4\lambda G^{-1} + 2$, then for all t and against any adversary, $\Delta_t \le 0$.*

Proof of Lemma 2.3. We restate the definition of Δ_t for easier reference.

$$\Delta_t = \bar{\nabla}_t V_\alpha(t,S_{t-1}) + \frac{1}{2}\bar{\nabla}_{SS} V_\alpha(t,S_{t-1}) + G^{-1}\lambda\left[\bar{\nabla}_S V_\alpha(t,S_{t-1}+1) - \bar{\nabla}_S V_\alpha(t,S_{t-1}-1)\right].$$

Let us define a function $g(t,S)$ as

$$\begin{aligned}
g(t,S) := & \frac{1}{2}V_\alpha(t,S+1) + \frac{1}{2}V_\alpha(t,S-1) - V_\alpha(t-1,S) \\
& + \frac{\lambda}{2G}\left[V_\alpha(t,S+2) + V_\alpha(t,S-2) - 2V_\alpha(t,S)\right],
\end{aligned}$$

then from the definition of discrete derivatives, $\Delta_t = g(t,S_{t-1})$. Also note that $g(t,S)$ is even in S, so we can only focus on positive values of S. The rest of the proof will show $g(t,S) \le 0$ for all $t \in \mathbb{N}_+$ and $S \ge 0$.

Let us start from the special case, $t = 1$. S can only take the value 0, therefore $g(1,S) = g(1,0)$. We now present a general result that upper bounds $g(t,0)$ for all $t \in \mathbb{N}_+$:

$$\begin{aligned}
& g(t,0) \\
= & V_\alpha(t,1) - V_\alpha(t-1,0) + G^{-1}\lambda V_\alpha(t,2) - G^{-1}\lambda V_\alpha(t,0) \\
= & C\sqrt{\alpha t}\left[2\int_0^{1/\sqrt{4\alpha t}}\left(\int_0^u \exp(x^2)dx\right)du + \frac{2\lambda}{G}\int_0^{1/\sqrt{\alpha t}}\left(\int_0^u \exp(x^2)dx\right)du + \sqrt{\frac{t-1}{t}} - 1\right] \\
\le & C\sqrt{\alpha t}\left[2\cdot\frac{1}{2}\cdot\frac{1}{\sqrt{4\alpha t}}\int_0^{1/\sqrt{4\alpha t}}\exp(x^2)dx + \frac{2\lambda}{G}\cdot\frac{1}{2}\cdot\frac{1}{\sqrt{\alpha t}}\int_0^{1/\sqrt{\alpha t}}\exp(x^2)dx + \sqrt{\frac{t-1}{t}} - 1\right] \\
& \qquad \text{(erfi}(x)\text{ is increasing and convex on } \mathbb{R}_+) \\
\le & C\sqrt{\alpha t}\left[\frac{1}{4\alpha t}\exp\left(\frac{1}{4\alpha t}\right) + \frac{\lambda}{G\alpha t}\exp\left(\frac{1}{\alpha t}\right) + \sqrt{\frac{t-1}{t}} - 1\right].
\end{aligned}$$

Since $\sqrt{1+x} \le 1 + x/2$ for all $x \ge -1$, we have $\sqrt{(t-1)/t} - 1 \le -1/(2t)$. Therefore, if $\alpha \ge 4\lambda G^{-1} + 2$, then

$$\begin{aligned}
g(t,0) & \le C\sqrt{\alpha t}\left[\frac{\lambda G^{-1} + (1/4)}{\alpha t}\exp\left(\frac{1}{\alpha t}\right) - \frac{1}{2t}\right] \\
& \le \frac{C\sqrt{\alpha}}{\sqrt{t}}\left[\frac{\lambda G^{-1} + (1/4)}{\alpha}\exp\left(\frac{1}{2}\right) - \frac{1}{2}\right] \le 0. \qquad (9)
\end{aligned}$$

As its special case, we have $g(1,0) \le 0$, which concludes the proof of the special case ($t = 1$).

Next, we prove $g(t,S) \le 0$ for general t, i.e., $t \ge 2$. Our overall strategy is to show that for all $0 \le S \le t-1$, $g(t,S) \le g(t,0)$, and then using the argument above we have $g(t,0) \le 0$. Concretely, let us calculate the first and second order derivatives of $g(t,S)$, using Lemma A.2.

$$\begin{aligned}
&\nabla_S g(t,S) \\
&= \frac{C}{2}\left[\int_0^{(S+1)/\sqrt{4\alpha t}} \exp(x^2)dx + \int_0^{(S-1)/\sqrt{4\alpha t}} \exp(x^2)dx - 2\int_0^{S/\sqrt{4\alpha(t-1)}} \exp(x^2)dx\right] \\
&\quad + \frac{\lambda C}{2G}\left[\int_0^{(S+2)/\sqrt{4\alpha t}} \exp(x^2)dx + \int_0^{(S-2)/\sqrt{4\alpha t}} \exp(x^2)dx - 2\int_0^{S/\sqrt{4\alpha t}} \exp(x^2)dx\right],
\end{aligned}$$

$$\begin{aligned}
&\nabla_{SS} g(t,S) \\
&= \frac{C}{4\sqrt{\alpha t}}\left[\frac{\lambda}{G}\exp\left(\frac{(S+2)^2}{4\alpha t}\right) + \exp\left(\frac{(S+1)^2}{4\alpha t}\right) - \frac{2\lambda}{G}\exp\left(\frac{S^2}{4\alpha t}\right)\right. \\
&\qquad \left. + \exp\left(\frac{(S-1)^2}{4\alpha t}\right) + \frac{\lambda}{G}\exp\left(\frac{(S-2)^2}{4\alpha t}\right)\right] - \frac{C}{2\sqrt{\alpha(t-1)}}\exp\left(\frac{S^2}{4\alpha(t-1)}\right) \\
&= \frac{C}{2\sqrt{\alpha t}}\exp\left(\frac{S^2}{4\alpha t}\right)\left[\frac{\lambda}{G}\exp\left(\frac{1}{\alpha t}\right)\cosh\left(\frac{S}{\alpha t}\right) + \exp\left(\frac{1}{4\alpha t}\right)\cosh\left(\frac{S}{2\alpha t}\right)\right. \\
&\qquad \left. - \frac{\lambda}{G} - \sqrt{\frac{t}{t-1}}\exp\left(\frac{S^2}{4\alpha t(t-1)}\right)\right]. \qquad (10)
\end{aligned}$$

Notice that $\nabla_S g(t,0) = 0$. To proceed, we aim to prove $\nabla_{SS} g(t,S) \le 0$ for all $S \ge 0$, which then shows $g(t,S) \le g(t,0)$. To this end, we will show the sum inside the bracket in (10) is negative. Denote it as $h(t,S)$, and more specifically,

$$\begin{aligned}
&h(t,S) \\
&:= \frac{\lambda}{G}\exp\left(\frac{1}{\alpha t}\right)\cosh\left(\frac{S}{\alpha t}\right) + \exp\left(\frac{1}{4\alpha t}\right)\cosh\left(\frac{S}{2\alpha t}\right) - \frac{\lambda}{G} - \sqrt{\frac{t}{t-1}}\exp\left(\frac{S^2}{4\alpha t(t-1)}\right).
\end{aligned}$$

The rest of the proof is divided into two steps: we first prove (i) $h(t,0) \le 0$; and then prove (ii) $\nabla_S h(t,S) \le 0$ for all $S \ge 0$.

Step 1: prove $h(t,0) \le 0$**.** From the definition of $h(t,S)$,

$$h(t,0) = \frac{\lambda}{G}\exp\left(\frac{1}{\alpha t}\right) + \exp\left(\frac{1}{4\alpha t}\right) - \frac{\lambda}{G} - \sqrt{\frac{t}{t-1}}.$$

Letting $x = 1/t$, then to prove $h(t,0) \le 0$ for all $t \ge 2$, it suffices to prove

$$\psi(x) := \frac{\lambda}{G}\exp\left(\frac{x}{\alpha}\right) + \exp\left(\frac{x}{4\alpha}\right) - \frac{\lambda}{G} - \sqrt{\frac{1}{1-x}} \le 0,$$

on the range $x \in (0, 1/2]$. $\psi(0) = 0$, and

$$\begin{aligned}
\nabla_x \psi(x) &= \frac{\lambda}{\alpha G}\exp\left(\frac{x}{\alpha}\right) + \frac{1}{4\alpha}\exp\left(\frac{x}{4\alpha}\right) - \frac{1}{2}(1-x)^{-3/2} \\
&\le \frac{4\lambda G^{-1}+1}{4\alpha}\exp\left(\frac{1}{2\alpha}\right) - \frac{1}{2},
\end{aligned}$$

which is negative when $\alpha \ge 4\lambda G^{-1} + 2$. Therefore, $h(t,0) \le 0$ for all $t \ge 2$.

Step 2: prove $\nabla_S h(t,S) \le 0$. Taking the derivative of $h(t,S)$,

$$\begin{aligned}\nabla_S h(t,S) &= \frac{\lambda}{\alpha t G}\exp\left(\frac{1}{\alpha t}\right)\sinh\left(\frac{S}{\alpha t}\right) + \frac{1}{2\alpha t}\exp\left(\frac{1}{4\alpha t}\right)\sinh\left(\frac{S}{2\alpha t}\right) \\ &\quad - \sqrt{\frac{t}{t-1}}\cdot\frac{S}{2\alpha t(t-1)}\exp\left(\frac{S^2}{4\alpha t(t-1)}\right) \\ &\le \left(\frac{\lambda}{\alpha t G} + \frac{1}{2\alpha t}\right)\exp\left(\frac{1}{\alpha t}\right)\sinh\left(\frac{S}{\alpha t}\right) - \frac{S}{2\alpha t^2}\sqrt{\frac{t}{t-1}}.\end{aligned}$$

Note that for all x, $\exp(-x) \ge 1-x$, therefore for all $0 \le x < 1$, $\exp(x/2) \le \sqrt{1/(1-x)}$. Assigning x to $1/t$ which is less than 1, we have for all $\alpha \ge 2$,

$$\exp\left(\frac{1}{\alpha t}\right) \le \exp\left(\frac{1}{2t}\right) \le \sqrt{\frac{t}{t-1}}.$$

Moreover, for all $0 \le x \le 1$, $\sinh(x) \le 2x$. Therefore,

$$\nabla_S h(t,S) \le \sqrt{\frac{t}{t-1}}\left[\frac{\lambda G^{-1} + (1/2)}{\alpha t}\sinh\left(\frac{S}{\alpha t}\right) - \frac{S}{2\alpha t^2}\right] \le \frac{S}{\alpha^2 t^2}\sqrt{\frac{t}{t-1}}\left[2\lambda G^{-1} + 1 - \frac{\alpha}{2}\right].$$

When $\alpha \ge 4\lambda G^{-1} + 2$, $\nabla_S h(t,S) \le 0$ for all $t \ge 2$ and $S \ge 0$.

Concluding the above two steps, we have shown $h(t,S) \le 0$. Plugging it back into (10), we have $\nabla_{SS} g(t,S) \le 0$, which shows that for all $t \ge 2$ and $S \ge 0$, $g(t,S) \le g(t,0)$. Finally, $g(t,0) \le 0$ following (9). □

Now, given the two important lemmas above (Lemma 2.2 and 2.3), our Theorem 1 follows from a standard loss-regret duality. Details are presented below.

Theorem 1. *If $\alpha = 4\lambda G^{-1} + 2$, then Algorithm 1 induced by the potential V_α guarantees*

$$\mathrm{Regret}_T^\lambda(u) \le \sqrt{(4\lambda G + 2G^2)T}\left[C + |u|\left(\sqrt{4\log\left(1 + \frac{|u|}{C}\right)} + 2\right)\right],$$

for all $u \in \mathbb{R}$ and $T \in \mathbb{N}_+$.

Proof of Theorem 1. Combining Lemma 2.1, 2.2 and 2.3, we have

$$\sum_{t=1}^T \left(g_t x_t + \lambda\left|x_t - x_{t+1}\right|\right) \le -G \cdot V_\alpha(T, S_T).$$

Consider $V_\alpha(T, S_T)$ as a function of S_T; let us write $V^*_{\alpha,T}(\cdot)$ as its Fenchel conjugate. Then, the augmented regret can be bounded as

$$\begin{aligned}\mathrm{Regret}_T^\lambda(u) &= \sum_{t=1}^T g_t(x_t - u) + \lambda\sum_{t=1}^{T-1}|x_t - x_{t+1}| \\ &\le G\cdot uS_T + \sum_{t=1}^T\left(g_t x_t + \lambda\left|x_t - x_{t+1}\right|\right) \\ &\le G\left[uS_T - V_\alpha(T, S_T)\right] \le G\cdot V^*_{\alpha,T}(u).\end{aligned}$$

The last step is to bound $V^*_{\alpha,T}(u)$, which also follows from a standard proof strategy.

$$V^*_{\alpha,T}(u) = \sup_{S\in\mathbb{R}} uS - V_\alpha(T,S).$$

It is clear that the supremum is uniquely achieved; let S^* be the maximizing argument. Then,

$$u = \nabla_S V_\alpha(T, S^*) = C\int_0^{S^*/\sqrt{4\alpha T}} \exp\left(x^2\right) dx.$$

If we define $\operatorname{erfi}(z) = \int_0^z \exp(x^2)dx$ (note that it a scaled version of the conventional *imaginary error function*), then $S^* = \sqrt{4\alpha T} \cdot \operatorname{erfi}^{-1}\left(uC^{-1}\right)$.

$$V^*_{\alpha,T}(u) = uS^* - V_\alpha(T, S^*) \le uS^* - V_\alpha(T, 0) = C\sqrt{\alpha T} + |u|\sqrt{4\alpha T} \cdot \operatorname{erfi}^{-1}\left(uC^{-1}\right).$$

Finally, as shown in [ZCP22b, Theorem 4], $\operatorname{erfi}^{-1}(x) \le 1 + \sqrt{\log(1+x)}$. Combining the above completes the proof. □

A.4 Conversion of loss-regret trade-offs

In this subsection we discuss the conversion of loss-regret trade-offs in unconstrained OLO. Our Theorem 1 guarantees a loss bound $\operatorname{Regret}^\lambda_T(0) = O(\sqrt{T})$ and an accompanying regret bound $\operatorname{Regret}^\lambda_T(u) = O\left(|u|\sqrt{T\log|u|}\right)$. By a doubling trick (effectively, a meta-algorithm), we can turn such guarantees into $\operatorname{Regret}^\lambda_T(0) = O(1)$ and $\operatorname{Regret}^\lambda_T(u) = O\left(|u|\sqrt{T\log(|u|\,T)}\right)$. These can be directly compared to [ZCP22a]. Concretely, we present the classical doubling trick as Algorithm 4.

Algorithm 4 Conversion of loss-regret trade-offs.

Require: A hyperparameter $C > 0$, and a base unconstrained OLO algorithm $\mathcal{A}$. Here we define $\mathcal{A}$ as the algorithm considered in Theorem 1, with $\alpha = 8\lambda G^{-1} + 2$.

1: **for** $m = 0, 1, 2, \dots$ **do**
2: Initialize a copy of $\mathcal{A}$ as $\mathcal{A}_m$, whose hyperparameter is set to $C \cdot 2^{-m}$.
3: Run $\mathcal{A}_m$ for 2^m rounds: $t = 2^m, 2^m + 1, \dots, 2^{m+1} - 1$.
4: **end for**

Theorem 5. *Let $\alpha = 8\lambda G^{-1} + 2$. With any hyperparameter $C > 0$, Algorithm 4 guarantees for all $u \in \mathbb{R}$ and $T \in \mathbb{N}_+$,*

$$\operatorname{Regret}^\lambda_T(u) \le \frac{\sqrt{2\alpha}G}{\sqrt{2}-1}\left[C + |u|\sqrt{T}\left(\sqrt{8\log\left(1 + \frac{|u|\,T}{C}\right)} + 2\sqrt{2}\right)\right].$$

Proof of Theorem 5. Algorithm 4 divides the time axis into intervals of doubling lengths. On the m-th interval, following Theorem 1, Algorithm 4 guarantees

$$\begin{aligned}
&\sum_{t=2^m}^{2^{m+1}-1} \left[g_t(x_t - u) + \lambda\,|x_t - x_{t+1}|\right] \\
\le\ &\sum_{t=2^m}^{2^{m+1}-1} g_t(x_t - u) + 2\lambda \sum_{t=2^m}^{2^{m+1}-2} |x_t - x_{t+1}| \qquad (x_{2^{m+1}} = x_{2^m} = 0;\ \text{Triangle inequality}) \\
\le\ &\sqrt{\alpha}G\left[\frac{C}{\sqrt{2^m}} + |u|\sqrt{2^m}\left(\sqrt{4\log\left(1 + \frac{|u|\cdot 2^m}{C}\right)} + 2\right)\right].
\end{aligned}$$

Now consider any time horizon T.

$$\begin{aligned}
\operatorname{Regret}^\lambda_T(u) &\le \sum_{m=0}^{\lceil\log_2 T\rceil} \sum_{t=2^m}^{2^{m+1}-1} \left[g_t(x_t - u) + \lambda\,|x_t - x_{t+1}|\right] \\
&\le \sqrt{\alpha}G \sum_{m=0}^{\lceil\log_2 T\rceil}\left[\frac{C}{\sqrt{2^m}} + |u|\sqrt{2^m}\left(\sqrt{4\log\left(1 + \frac{|u|\cdot 2^m}{C}\right)} + 2\right)\right] \\
&\le \sqrt{\alpha}G\left[C\sum_{m=0}^{\lceil\log_2 T\rceil}\left(\frac{1}{\sqrt{2}}\right)^m + |u|\left(\sqrt{4\log\left(1 + \frac{|u|\,T}{C}\right)} + 2\right)\sum_{m=0}^{\lceil\log_2 T\rceil}\sqrt{2^m}\right] \\
&\le \frac{\sqrt{2\alpha}G}{\sqrt{2}-1}\left[C + |u|\sqrt{T}\left(\sqrt{8\log\left(1 + \frac{|u|\,T}{C}\right)} + 2\sqrt{2}\right)\right].
\end{aligned}$$

□

Let us compare it to Theorem 3, i.e., [ZCP22a, Theorem 1], which guarantees

$$\mathrm{Regret}_T^\lambda(u) \le (G+\lambda)\left[C + |u|\sqrt{2T}\left(\frac{3}{2} + \log\frac{\sqrt{2}\,|u|\,T^{5/2}}{C}\right)\right].$$

If we only care about the dependence on $|u|$ and T, then with the same loss bound $\mathrm{Regret}_T^\lambda(0) = O(1)$, our algorithm improves the regret bound $\mathrm{Regret}_T^\lambda(u)$ from $O(|u|\sqrt{T\log(|u|\,T)})$ to $O(|u|\sqrt{T\log(|u|\,T)})$. The latter matches a lower bound [Ora13], therefore achieves Pareto-optimality in this regime.

A.5 Details on the continuous-time derivation

In Step 3 of Section 2.4, we need to perform second-order Taylor approximations on the scaled recursion (6). Details are included here for completeness.

$$V(t-\varepsilon^2, S) = V(t,S) - \varepsilon^2\nabla_t V(t,S) + o(\varepsilon^2),$$
$$V(t, S-\varepsilon g) = V(t,S) - \varepsilon g\nabla_S V(t,S) + \frac{1}{2}\varepsilon^2 g^2\nabla_{SS}V(t,S) + o(\varepsilon^2),$$
$$\bar\nabla_S^\varepsilon V(t,S) = \frac{1}{2\varepsilon}\left[V(t,S+\varepsilon) - V(t,S-\varepsilon)\right] = \nabla_S V(t,S) + o(\varepsilon),$$
$$\bar\nabla_S^\varepsilon V(t+\varepsilon^2, S-\varepsilon g) = \frac{1}{2\varepsilon}\left[V(t+\varepsilon^2, S-\varepsilon g+\varepsilon) - V(t+\varepsilon^2, S-\varepsilon g-\varepsilon)\right],$$

where

$$V(t+\varepsilon^2, S-\varepsilon g+\varepsilon) = V(t,S) + \varepsilon^2\nabla_t V(t,S) + (1-g)\varepsilon\nabla_S V(t,S) + \frac{1}{2}(1-g)^2\varepsilon^2\nabla_{SS}V(t,S) + o(\varepsilon^2),$$

$$V(t+\varepsilon^2, S-\varepsilon g-\varepsilon) = V(t,S) + \varepsilon^2\nabla_t V(t,S) + (-1-g)\varepsilon\nabla_S V(t,S) + \frac{1}{2}(1+g)^2\varepsilon^2\nabla_{SS}V(t,S) + o(\varepsilon^2).$$

B Extension to general OLO settings

This section presents extensions of our 1D unconstrained OLO algorithm to more general settings.

B.1 Constrained domain

First, consider a constrained domain $\mathcal{X} \subset \mathbb{R}$. We can use a well-known black-box reduction [CO18, Cut20] on top of our 1D unconstrained algorithm (Algorithm 1), such that the *exact* bound in Theorem 1 carries over (w.r.t. any constrained comparator $u \in \mathcal{X}$). Concretely, the pseudo-code is shown as Algorithm 5, where Π denotes the absolute value projection in 1D.

Similar strategies apply to higher-dimensional problems, but here we emphasize the 1D special case due to an additional feature: if the domain $\mathcal{X}$ has a finite diameter D, then the switching cost alone of the combined algorithm has a $\tilde O(D\sqrt{\tau})$ bound on any time interval of length τ. This could be useful when switching costs have high priority [SK21, WWYZ21] and should be independently bounded. Moreover, it allows the combination of comparator adaptive algorithms [ZCP22a] in settings with long term prediction effects (e.g., switching cost or memory).

Theorem 6. *Let x^* be an arbitrary point in $\mathcal{X}$. For all $C > 0$, Algorithm 5 guarantees*

$$\mathrm{Regret}_T^\lambda(u) \le \sqrt{(4\lambda G + 2G^2)T}\left[C + |u-x^*|\left(\sqrt{4\log\left(1 + \frac{|u-x^*|}{C}\right)} + 2\right)\right],$$

for all $u \in \mathcal{X}$ and $T \in \mathbb{N}_+$. Moreover, if $\mathcal{X}$ has a finite diameter D, then on any time interval $[T_1 : T_2] \subset \mathbb{N}_+$, the same algorithm guarantees

$$\sum_{t=T_1}^{T_2-1} |x_t - x_{t+1}| \le 22\sqrt{T_2 - T_1}\left[2D + C + 2D\sqrt{\log(1 + DC^{-1})}\right].$$

Algorithm 5 1D constrained OLO with switching costs.

Require: A hyperparameter $C > 0$, a closed and convex domain $\mathcal{X} \subset \mathbb{R}$, and an unconstrained algorithm $\mathcal{A}$ (Algorithm 1 induced by $V_{4\lambda G^{-1}+2}$ and the hyperparameter C). Let x^* be an arbitrary point in $\mathcal{X}$.

1: **for** $t = 1, 2, \dots$ **do**
2: Query $\mathcal{A}$ for its prediction $\tilde{x}_t$.
3: Predict $x_t = \Pi_{\mathcal{X}}(\tilde{x}_t + x^*)$ and receive a loss gradient g_t.
4: Define a surrogate loss gradient $\tilde{g}_t$ as

$$\tilde{g}_t = \begin{cases} g_t, & \text{if } g_t(\tilde{x}_t + x^*) \geq g_t x_t, \\ 0, & \text{otherwise,} \end{cases}$$

and send $\tilde{g}_t$ to $\mathcal{A}$.
5: **end for**

Before presenting the proof, let us discuss its technical significance. Typically, the constrained-to-unconstrained reduction is used as a black box. However, the second part of Theorem 6 relies on a *non-black-box* use of this approach – we characterize how this reduction implicitly controls the unconstrained base algorithm, resulting in the "concentration" of its sufficient statistic S_t (i.e., $S_t = O(\sqrt{t})$), as if losses are stochastic. Such an observation could be of independent interest.

Proof of Theorem 6. The first part of the theorem directly follows from [Cut20, Theorem 2] and the contraction property of 1D projections. As for the second part, we divide the proof into five steps.

Step 1: the "concentration" of S_t Without loss of generality, assume $S_{t-1} \geq 0$. Considering the prediction $\tilde{x}_t = \bar{\nabla}_S V_\alpha(t, S_{t-1})$ of the unconstrained base algorithm, there are two cases.

- **Case 1: $\tilde{x}_t \leq D$.** Due to convexity,

$$\tilde{x}_t = \bar{\nabla}_S V_\alpha(t, S_{t-1}) = C\sqrt{\alpha t} \int_{(S_{t-1}-1)/\sqrt{4\alpha t}}^{(S_{t-1}+1)/\sqrt{4\alpha t}} \left(\int_0^u \exp(x^2) dx \right) du \geq C \int_0^{S_{t-1}/\sqrt{4\alpha t}} \exp(x^2) dx.$$

  Similar to the proof of Theorem 1, if we define $\operatorname{erfi}(z) = \int_0^z \exp(x^2) dx$, then $S_{t-1} \leq \sqrt{4\alpha t} \cdot \operatorname{erfi}^{-1}(DC^{-1})$. As for the next round, $|S_t| \leq S_{t-1} + |g_t| / G \leq \sqrt{4\alpha t} \cdot \operatorname{erfi}^{-1}(DC^{-1}) + 1$.

- **Case 2: $\tilde{x}_t > D$.** In this case, since $x^* \in \mathcal{X}$, we have $\tilde{x}_t + x^*$ larger than the maximum element of $\mathcal{X}$, leading to $\tilde{x}_t + x^* > x_t$. Due to the definition of the surrogate loss, $\tilde{g}_t \geq 0$. Therefore, $|S_t| \leq \max\{|S_{t-1}|, |\tilde{g}_t / G|\} \leq \max\{|S_{t-1}|, 1\}$.

Combining the two cases and their analogous arguments for $S_{t-1} \leq 0$, we can see that for all t, $|S_t| \leq \max\left\{\sqrt{4\alpha t} \cdot \operatorname{erfi}^{-1}(DC^{-1}) + 1, |S_{t-1}|, 1\right\}$. By induction, we obtain for all t,

$$|S_t| \leq \sqrt{4\alpha t} \cdot \operatorname{erfi}^{-1}(DC^{-1}) + 1.$$

Step 2: bounding the switching cost using S_t Still, assume $S_{t-1} \geq 0$ without loss of generality. From Lemma 2.2,

$$\begin{aligned}
& |x_t - x_{t+1}| \\
\leq\ & \bar{\nabla}_S V_\alpha(t, S_{t-1} + 1) - \bar{\nabla}_S V_\alpha(t, S_{t-1} - 1) \\
=\ & C\sqrt{\alpha t} \left[\int_{S_{t-1}/\sqrt{4\alpha t}}^{(S_{t-1}+2)/\sqrt{4\alpha t}} \left(\int_0^u \exp(x^2) dx \right) du - \int_{(S_{t-1}-2)/\sqrt{4\alpha t}}^{S_{t-1}/\sqrt{4\alpha t}} \left(\int_0^u \exp(x^2) dx \right) du \right] \\
\leq\ & C\sqrt{\alpha t} \left[\frac{2}{\sqrt{4\alpha t}} \int_0^{(S_{t-1}+2)/\sqrt{4\alpha t}} \exp(x^2) dx - \frac{2}{\sqrt{4\alpha t}} \int_0^{(S_{t-1}-2)/\sqrt{4\alpha t}} \exp(x^2) dx \right] \\
=\ & C \int_{(S_{t-1}-2)/\sqrt{4\alpha t}}^{(S_{t-1}+2)/\sqrt{4\alpha t}} \exp(x^2) dx \leq \frac{2C}{\sqrt{\alpha t}} \exp\left(\frac{(S_{t-1}+2)^2}{4\alpha t} \right).
\end{aligned}$$

Step 3: plug in the concentration of S_t Next, we use the upper bound on S_{t-1} to show that $|x_t - x_{t+1}| = O(Ct^{-1/2}\exp[(\text{erfi}^{-1}(DC^{-1}))^2])$. To this end, we discuss two cases regarding how the "concentration" bound (i.e., $O(\sqrt{t})$) compares to the trivial bound (i.e., $S_t \le t$).

- **Case 1:** $\sqrt{4\alpha t}\cdot\text{erfi}^{-1}(DC^{-1}) \ge t$**.** In this case, note that $S_{t-1}+1 \le t$ and $\alpha \ge 2$,

$$|x_t - x_{t+1}| \le \frac{2C}{\sqrt{\alpha t}}\exp\left(\frac{(S_{t-1}+2)^2}{4\alpha t}\right) = \frac{2C}{\sqrt{\alpha t}}\exp\left(\frac{S_{t-1}^2}{4\alpha t}\right)\exp\left(\frac{4S_{t-1}+4}{4\alpha t}\right)$$
$$\le \frac{2\sqrt{e}C}{\sqrt{\alpha t}}\exp\left(\frac{S_{t-1}^2}{4\alpha t}\right).$$

  Since $\sqrt{4\alpha t}\cdot\text{erfi}^{-1}(DC^{-1}) \ge t$, we have $t \le 4\alpha(\text{erfi}^{-1}(DC^{-1}))^2$. Therefore,

$$\exp\left(\frac{S_{t-1}^2}{4\alpha t}\right) \le \exp\left(\frac{t}{4\alpha}\right) \le \exp\left[\left(\text{erfi}^{-1}(DC^{-1})\right)^2\right],$$
$$|x_t - x_{t+1}| \le \frac{2\sqrt{e}C}{\sqrt{\alpha t}}\exp\left[\left(\text{erfi}^{-1}(DC^{-1})\right)^2\right].$$

- **Case 2:** $\sqrt{4\alpha t}\cdot\text{erfi}^{-1}(DC^{-1}) < t$**.** Plugging the $O(\sqrt{t})$ bound for S_{t-1} into $|x_t - x_{t+1}|$,

$$|x_t - x_{t+1}| \le \frac{2C}{\sqrt{\alpha t}}\exp\left(\frac{(\sqrt{4\alpha(t-1)}\cdot\text{erfi}^{-1}(DC^{-1})+3)^2}{4\alpha t}\right)$$
$$\le \frac{2C}{\sqrt{\alpha t}}\exp\left[\left(\text{erfi}^{-1}(DC^{-1})\right)^2\right]\exp\left(\frac{6t+9}{4\alpha t}\right)$$
$$\le \frac{2e^2C}{\sqrt{\alpha t}}\exp\left[\left(\text{erfi}^{-1}(DC^{-1})\right)^2\right].$$

Combining the above, we have

$$|x_t - x_{t+1}| \le \frac{2e^2C}{\sqrt{\alpha t}}\exp\left[\left(\text{erfi}^{-1}(DC^{-1})\right)^2\right].$$

Step 4: upper-bound $\exp[(\text{erfi}^{-1}(x))^2]$**.** Let us consider a related function $\int_0^x \text{erfi}(u)du$. Using integration by parts,

$$\int_0^x \text{erfi}(u)du = u\cdot\text{erfi}(u)\Big|_{u=0}^{x} - \int_0^x u\exp(u^2)du$$
$$= x\cdot\text{erfi}(x) - \frac{1}{2}\exp(x^2) + \frac{1}{2}.$$

Plugging in $x = \text{erfi}^{-1}(DC^{-1})$, we have

$$\exp\left[\left(\text{erfi}^{-1}(DC^{-1})\right)^2\right] = 2DC^{-1}\cdot\text{erfi}^{-1}(DC^{-1}) + 1 - 2\int_0^{\text{erfi}^{-1}(DC^{-1})}\text{erfi}(u)du$$
$$\le 2DC^{-1}\cdot\text{erfi}^{-1}(DC^{-1}) + 1.$$

Then, as we did in Theorem 1, we plug in $\text{erfi}^{-1}(x) \le 1 + \sqrt{\log(1+x)}$ and obtain

$$|x_t - x_{t+1}| \le \frac{11}{\sqrt{t}}\left\{2D\left[1 + \sqrt{\log(1+DC^{-1})}\right] + C\right\}.$$

Step 5: final bits. Note that

$$\sum_{t=T_1}^{T_2-1}\frac{1}{\sqrt{t}} \le \int_{T_1-1}^{T_2-1}\frac{1}{\sqrt{x}}dx \le 2\sqrt{T_2-1} - 2\sqrt{T_1-1} \le 2\sqrt{T_2-T_1}.$$

Combining it with our bound on $|x_t - x_{t+1}|$ completes the proof. □

B.2 Higher dimensional domain

Now we present generalizations of our 1D algorithm to higher dimensional domains. We will primarily consider switching costs measured by the L_1 norm. This serves as a nice bridge towards our LEA approach and financial applications. Extensions to other norms, e.g., L_2 norm, is sketched at the end.

Concretely, let the domain $\mathcal{X} = \mathbb{R}^d$, $\|g_t\|_\infty \leq G$, and the switching costs are measured by the L_1 norm. We run Algorithm 1 on each coordinate separately [SM12], and scale the hyperparameter C by $1/d$. The pseudo-code is presented as Algorithm 6.

Algorithm 6 d-dimensional OLO with L_1 norm switching costs.

Require: A hyperparameter $C > 0$ and Algorithm 1.

1: For each dimension $i \in [1 : d]$, initialize a copy of Algorithm 1 as $\mathcal{A}_i$. It uses the hyperparameter C/d and our potential V_α, with $\alpha = 4\lambda G^{-1} + 2$.
2: **for** $t = 1, 2, \dots$ **do**
3: For all i, query $\mathcal{A}_i$ and assign its prediction to $x_{t,i}$. Define a vector as $x_t = [x_{t,1}, \dots, x_{t,d}] \in \mathbb{R}^d$.
4: Predict x_t and receive a loss gradient $g_t = [g_{t,1}, \dots, g_{t,d}]$.
5: For all i, send $g_{t,i}$ to $\mathcal{A}_i$ as a new surrogate loss gradient.
6: **end for**

Theorem 7. *For all $C > 0$, Algorithm 6 guarantees ($\alpha = 4\lambda G^{-1} + 2$)*

$$\sum_{t=1}^{T} \langle g_t, x_t - u \rangle + \lambda \sum_{t=1}^{T-1} \|x_t - x_{t+1}\|_1 \leq G\sqrt{\alpha T} \left[C + \|u\|_1 \left(\sqrt{4 \log \left(1 + \frac{\|u\|_\infty d}{C} \right)} + 2 \right) \right],$$

for all $u \in \mathbb{R}^d$ and $T \in \mathbb{N}_+$.

Proof of Theorem 7. We simply combine the regret on each coordinate:

$$\begin{aligned}
& \sum_{t=1}^{T} \langle g_t, x_t - u \rangle + \lambda \sum_{t=1}^{T-1} \|x_t - x_{t+1}\|_1 \\
= & \sum_{i=1}^{d} \left[\sum_{t=1}^{T} g_{t,i}(x_{t,i} - u_i) + \lambda \sum_{t=1}^{T-1} |x_{t,i} - x_{t+1,i}| \right] \\
\leq & \sqrt{(4\lambda G + 2G^2)T} \sum_{i=1}^{d} \left[\frac{C}{d} + |u_i| \left(\sqrt{4 \log \left(1 + \frac{|u_i| d}{C} \right)} + 2 \right) \right] \\
\leq & \sqrt{(4\lambda G + 2G^2)T} \left[C + \|u\|_1 \left(\sqrt{4 \log \left(1 + \frac{\|u\|_\infty d}{C} \right)} + 2 \right) \right].
\end{aligned}$$

□

As for L_2 norm switching costs, we can follow the polar decomposition technique from [ZCP22a, Section 2.2], which uses our 1D unconstrained OLO algorithm as the base algorithm. The only required modification is that the base algorithm should account for an extra regularization term. Concretely, instead of bounding the augmented regret (2), we should bound

$$\sum_{t=1}^{T} g_t(x_t - u) + \lambda \sum_{t=1}^{T-1} |x_t - x_{t+1}| + \frac{\gamma}{\sqrt{t}} \sum_{t=1}^{T} |x_t|,$$

for any given weight γ.

To this end, we can consider the Online Convex Optimization problem with switching costs, where the loss functions are defined by $l_t(x) = g_t x + \gamma t^{-1/2} |x|$. Such a loss function is Lipschitz, therefore we can use the OCO-OLO reduction, and run our Algorithm 1 on its linearized surrogate. Details are omitted.

C Details on LEA

In this section we present techniques that extend our 1D OLO algorithm to LEA with switching costs. We show that with a streamlined analysis, the general Banach version of the constrained domain reduction [CO18] can already convert 1D OLO algorithms to LEA, thus appears to be more general than the specialized techniques [LS15, OP16]. Our approach is presented as Algorithm 7.

Algorithm 7 Converting OLO to LEA via the constrained domain reduction.

Require: A prior $\pi = [\pi_1, \dots, \pi_d]$ in the relative interior of $\Delta(d)$, and Algorithm 5.

1: For each dimension $i \in [1 : d]$, initialize a copy of Algorithm 5 as $\mathcal{A}_i$. We set $\tilde{\lambda} = 4\lambda$ and $\tilde{G} = 2G$ as the switching cost weight and the Lipschitz constant considered by A_i. Moreover, A_i uses the domain $\mathcal{X} = \mathbb{R}_+$, the offset $x^* = \pi_i$, the hyperparameter π_i, and our potential V_α, where $\alpha = 4\tilde{\lambda}\tilde{G}^{-1} + 2$.

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

3: For all i, query $\mathcal{A}_i$ and assign its prediction to $w_{t,i}$. Define the weight vector as $w_t = [w_{t,1}, \dots, w_{t,d}] \in \mathbb{R}_+^d$.

4: Compute the LEA prediction $x_t = [x_{t,1}, \dots, x_{t,d}]$ from

$$x_{t,i} = \frac{w_{t,i} + \frac{1}{d}\max\{0, 1 - \|w_t\|_1\}}{\max\{\|w_t\|_1, 1\}}.$$

5: Predict x_t and receive a loss vector $g_t \in [-G, G]^d$.

6: For all i, compute

$$z_{t,i} = \begin{cases} g_{t,i} - \|g_t\|_\infty, & \text{if } \|w_t\|_1 < 1, \\ g_{t,i}, & \text{if } \|w_t\|_1 = 1, \\ g_{t,i} + \|g_t\|_\infty, & \text{if } \|w_t\|_1 > 1, \end{cases}$$

and return $z_{t,i}$ to $\mathcal{A}_i$ as a new surrogate loss.

7: **end for**

C.1 An auxiliary lemma

Before presenting the performance guarantee of Algorithm 7, we first prove an auxiliary lemma.

Lemma C.1. *For all $x \geq 0$,*

$$|x - 1| \log(1 + |x - 1|) \leq 2(1 - x + x \log x).$$

Proof of Lemma C.1. Define LHS $-$ RHS as $h(x)$. Clearly, $h(1) = 0$. When $x > 1$,

$$h'(x) = 1 - \log x - x^{-1}.$$

It equals 0 when $x = 1$, and $h''(x) = (1 - x)/x^2$ which is negative for all $x > 1$. Therefore, $h(x) \leq 0$ for all $x \geq 1$.

As for the case of $x < 1$,

$$h'(x) = -\log(2 - x) - \frac{1 - x}{2 - x} - 2\log x,$$

$$h''(x) = -\frac{x^2 - x + 2}{(x - 2)^2 x} < 0,$$

therefore $h(x) \leq 0$ for all $0 \leq x \leq 1$. □

C.2 Analysis of Algorithm 7

Next, we present our result on LEA with switching cost.

Theorem 2. *For LEA with switching cost, given any prior π in the relative interior of $\Delta(d)$, Algorithm 7 from Appendix C.2 guarantees*

$$\sum_{t=1}^{T} \langle g_t, x_t - u \rangle + \lambda \sum_{t=1}^{T-1} \|x_t - x_{t+1}\|_1 = \left[\sqrt{\mathrm{TV}(u||\pi) \cdot \mathrm{KL}(u||\pi)} + 1 \right] \cdot O\left(\sqrt{(\lambda G + G^2)T} \right),$$

for all $u \in \Delta(d)$ and $T \in \mathbb{N}_+$.

Proof of Theorem 2. We divide the proof into three steps.

Step 1 The first step is to show that (i) for all $u \in \Delta(d)$, $\langle g_t, x_t - u \rangle \leq \langle z_t, w_t - u \rangle$; and (ii) $\|x_t - x_{t+1}\|_1 \leq O(\|w_t - w_{t+1}\|_1)$. In this way, we can translate the LEA problem to a d-dimensional OLO problem with the loss vector z_t, despite not achieving the root KL bound yet.

To prove the goal (i), we consider two cases, $\|w_t\|_1 < 1$ and $\|w_t\|_1 > 1$ (the case of $\|w_t\|_1 = 1$ trivially holds). If $\|w_t\|_1 < 1$, we have $x_t = w_t + d^{-1}(1 - \|w_t\|_1)$ and $z_t = g_t - \|g_t\|_\infty$.

$$\langle g_t, x_t - u \rangle = \langle g_t, w_t - u \rangle + (1 - \|w_t\|_1) \left(\sum_i g_{t,i}/d \right),$$

$$\langle z_t, w_t - u \rangle = \langle g_t, w_t - u \rangle + (1 - \|w_t\|_1) \|g_t\|_\infty,$$

therefore $\langle g_t, x_t - u \rangle \leq \langle z_t, w_t - u \rangle$. As for the case of $\|w_t\|_1 > 1$, similarly, $x_t = w_t / \|w_t\|_1$, $z_t = g_t + \|g_t\|_\infty$, $\langle g_t, x_t - u \rangle = \langle g_t, w_t/\|w_t\|_1 - u \rangle$, and $\langle z_t, w_t - u \rangle = \langle g_t, w_t - u \rangle + \|g_t\|_\infty (\|w_t\|_1 - 1)$.

$$\langle g_t, x_t - u \rangle - \langle z_t, w_t - u \rangle = -\left(\langle g_t, x_t \rangle + \|g_t\|_\infty \right) (\|w_t\|_1 - 1) \leq 0.$$

Now consider the goal (ii). To avoid cluttered notations, define $a_t = w_t + d^{-1} \max\{0, 1 - \|w_t\|_1\}$ and $A_t = \max\{\|w_t\|_1, 1\}$. Note that $A_t = \|a_t\|_1$.

$$\begin{aligned}
\|x_t - x_{t+1}\|_1 &= \left\| \frac{a_t}{A_t} - \frac{a_{t+1}}{A_{t+1}} \right\|_1 \\
&= \left\| \frac{(a_t - a_{t+1})A_{t+1} + a_{t+1}(A_{t+1} - A_t)}{A_t A_{t+1}} \right\|_1 \\
&\leq \frac{1}{A_t} \|a_t - a_{t+1}\|_1 + \frac{1}{A_t}(A_{t+1} - A_t) \leq 2 \|a_t - a_{t+1}\|_1 .
\end{aligned}$$

$$\begin{aligned}
\|a_t - a_{t+1}\|_1 &= \left\| w_t + d^{-1} \max\{0, 1 - \|w_t\|_1\} - w_{t+1} - d^{-1} \max\{0, 1 - \|w_{t+1}\|_1\} \right\|_1 \\
&\leq \|w_t - w_{t+1}\|_1 + |\max\{0, 1 - \|w_t\|_1\} - \max\{0, 1 - \|w_{t+1}\|_1\}| \\
&\leq \|w_t - w_{t+1}\|_1 + |\|w_t\|_1 - \|w_{t+1}\|_1| \leq 2 \|w_t - w_{t+1}\|_1 .
\end{aligned}$$

Therefore, $\|x_t - x_{t+1}\|_1 \leq 4 \|w_t - w_{t+1}\|_1$.

Step 2 The second step is to add up the regret bound for each coordinates. Consider the i-th coordinate. Note that $|z_{t,i}| \leq 2G$. Using Theorem 6, for all $u_{1d} \in \mathbb{R}_+$,

$$\begin{aligned}
&\sum_{t=1}^{T} z_{t,i}(w_{t,i} - u_{1d}) + \tilde{\lambda} \sum_{t=1}^{T-1} |w_{t,i} - w_{t+1,i}| \\
\leq\ & \sqrt{(4\tilde{\lambda}\tilde{G} + 2\tilde{G}^2)T} \left[\pi_i + |u_{1d} - \pi_i| \left(\sqrt{4 \log\left(1 + \frac{|u_{1d} - \pi_i|}{\pi_i} \right)} + 2 \right) \right] \\
=\ & \sqrt{(32\lambda G + 8G^2)T} \left[\pi_i + |u_{1d} - \pi_i| \left(\sqrt{4 \log\left(1 + \frac{|u_{1d} - \pi_i|}{\pi_i} \right)} + 2 \right) \right].
\end{aligned}$$

Then, by summing up all the coordinates, for all $u \in \Delta(d)$,

$$\begin{aligned}
&\sum_{t=1}^{T} \langle g_t, x_t - u \rangle + \lambda \sum_{t=1}^{T-1} \|x_t - x_{t+1}\|_1 \\
\leq\ & \sum_{t=1}^{T} \langle z_t, w_t - u \rangle + 4\lambda \sum_{t=1}^{T-1} \|w_t - w_{t+1}\|_1 \\
=\ & \sum_{i=1}^{d} \left[\sum_{t=1}^{T} z_{t,i}(w_{t,i} - u_i) + \tilde{\lambda} \sum_{t=1}^{T-1} |w_{t,i} - w_{t+1,i}| \right] \\
\leq\ & \sqrt{(32\lambda G + 8G^2)T} \left[1 + 2\|u - \pi\|_1 + 2 \sum_{i=1}^{d} |u_i - \pi_i| \sqrt{\log\left(1 + \frac{|u_i - \pi_i|}{\pi_i}\right)} \right] \\
\leq\ & \sqrt{(32\lambda G + 8G^2)T} \left[1 + 2\|u - \pi\|_1 + 2\sqrt{\|u - \pi\|_1} \sqrt{\sum_{i=1}^{d} |u_i - \pi_i| \log\left(1 + \frac{|u_i - \pi_i|}{\pi_i}\right)} \right]. \qquad \text{(Cauchy-Schwarz)}
\end{aligned}$$

Observe that since u and π both belong to $\Delta(d)$, $\|u - \pi\|_1 \leq 2$. If we define a function f as

$$f := |x - 1| \log(1 + |x - 1|),$$

then using the standard definition of *f-divergence*

$$D_f(u||\pi) := \sum_{i=1}^{d} \pi_i f\left(\frac{u_i}{\pi_i}\right),$$

we have

$$\sum_{t=1}^{T} \langle g_t, x_t - u \rangle + \lambda \sum_{t=1}^{T-1} \|x_t - x_{t+1}\|_1 = \left[\sqrt{\mathrm{TV}(u||\pi) \cdot D_f(u||\pi)} + 1 \right] \cdot O\left(\sqrt{(\lambda G + G^2)T} \right).$$

Step 3 The last step is to upper bound $D_f(u||\pi)$ by $\mathrm{KL}(u||\pi)$. To this end, notice that $\mathrm{KL}(u||\pi) = D_g(u||\pi)$, where

$$g(x) := 1 - x + x \log x.$$

By Lemma C.1, $f(x) \leq 2g(x)$ for all $x \geq 0$, therefore $D_f(u||\pi) \leq 2D_g(u||\pi) = 2\mathrm{KL}(u||\pi)$. □

C.3 Discussion on Algorithm 7

Here are some discussions to conclude our LEA analysis. First, the surrogate loss z_t defined in Line 6 follows exactly the definition in [CO18, Algorithm 3]. We adopt this choice just to show the power of this general reduction technique. However, one could use other choices of z_t and obtain the same guarantee, although the empirical performance could be different. For example, one can use

$$z_{t,i} = \begin{cases} g_{t,i} - \max_i g_{t,i}, & \text{if } \|w_t\|_1 < 1, \\ g_{t,i}, & \text{if } \|w_t\|_1 = 1, \\ g_{t,i} - \min_i g_{t,i}, & \text{if } \|w_t\|_1 > 1, \end{cases}$$

and clearly, the exact same proof still holds. Another possible choice is

$$z_{t,i} = \begin{cases} g_{t,i} - \sum_i g_{t,i}, & \text{if } \|w_t\|_1 < 1, \\ g_{t,i}, & \text{if } \|w_t\|_1 = 1, \\ g_{t,i} - \langle g_t, x_t \rangle, & \text{if } \|w_t\|_1 > 1. \end{cases}$$

This is more analogous to the surrogate losses in existing specialized approaches [LS15, OP16].

Also, to justify the improvement of $\sqrt{\mathrm{TV} \cdot \mathrm{KL}}$ over $\sqrt{\mathrm{KL}}$, here is an example. For all $d \geq 3$, define $p, q \in \Delta(d)$ from

$$p_1 = \frac{1}{\sqrt{\log d}}, \quad q_1 = \frac{1}{d\sqrt{\log d}},$$

and

$$p_i = \frac{1-p_1}{d-1}, \quad q_i = \frac{1-q_1}{d-1}, \quad \forall i \in [2:d].$$

Then,

$$\mathrm{TV}(p||q) = \frac{1}{2}\left[|p_1 - q_1| + (d-1)\left|\frac{1-p_1}{d-1} - \frac{1-q_1}{d-1}\right|\right] = |p_1 - q_1| = \frac{d-1}{d\sqrt{\log d}},$$

$$\begin{aligned}
\mathrm{KL}(p||q) &= p_1 \log \frac{p_1}{q_1} + (d-1)\cdot\frac{1-p_1}{d-1}\log\frac{1-p_1}{1-q_1} \\
&= \sqrt{\log d} + \left(1 - \frac{1}{\sqrt{\log d}}\right)\log\left(1 - \frac{d-1}{d\sqrt{\log d}-1}\right) \\
&\geq \sqrt{\log d} + \log\left(1 - \frac{d}{d\sqrt{\log d}-1}\right) = \sqrt{\log d} - o(1).
\end{aligned}$$

Since we also have

$$\mathrm{KL}(p||q) = \sqrt{\log d} + (1-p_1)\log\frac{1-p_1}{1-q_1} \leq \sqrt{d},$$

we can combine the above and obtain $\mathrm{TV}(p||q)\cdot\mathrm{KL}(p||q) \leq 1$ and $\mathrm{KL}(p||q) \geq \sqrt{\log d} - o(1)$. If our comparator u and prior π take the value of p and q respectively, then even without switching costs, Theorem 2 saves a $(\log d)^{1/4}$ factor from the existing comparator adaptive bounds.

D Application to portfolio selection

To complement our theoretical results, we present applications to a portfolio selection problem with transaction costs.[7] Online portfolio selection has been studied by multiple communities, resulting in a large amount of literature (see [LH14, Doc16] for general expositions). Here we focus on an *unconstrained* setting, allowing both short selling (i.e., holding negative amount of assets) and margin trading (i.e., borrowing money to buy assets). This is related, but different from classical settings in the literature, as discussed in Appendix D.2.

D.1 Problem setting

We consider a market with d assets and discrete trading period $t \in \mathbb{N}_+$. In the t-th round, an algorithm chooses a portfolio vector $x_t = [x_{t,1}, \ldots, x_{t,d}] \in \mathbb{R}^d$, where $x_{t,i}$ is the *number of shares* of the i-th asset that the algorithm suggests to hold. Compared to the previous round, we need to buy $x_{t,i} - x_{t-1,i}$ shares[8] (or sell, if negative), which requires paying a $\lambda\,|x_{t,i} - x_{t-1,i}|$ transaction cost. Then, the market reveals a number $g_{t,i} \in [-G, G]$, which represents the price change per share (of the i-th asset) in this round. This effectively increases the value of our portfolio by $\langle g_t, x_t\rangle$.

The considered performance metric is the increased amount of *wealth* on any time horizon $[1:T] \subset \mathbb{N}_+$, and such wealth includes the total value of our portfolio *plus cash*. Our goal is to show that the performance of our algorithm is never much worse than that of any unconstrained *Buy-and-Hold* (BAH) strategy, which picks a portfolio vector $u \in \mathbb{R}^d$ at the beginning and holds that amount throughout the considered time horizon. That is, we aim to upper bound $\sum_{t=1}^{T}\langle -g_t, x_t - u\rangle + \lambda\sum_{t=1}^{T-1}\|x_t - x_{t+1}\|_1$ for all $u \in \mathbb{R}^d$ and $T \in \mathbb{N}_+$. This is exactly the setting of our high dimensional OLO problem (Appendix B.2) with flipped gradients, therefore if we use our high dimensional OLO algorithm (Algorithm 6), then the same theoretical result (Theorem 7) carries over.

D.2 Comparison to the rebalancing setting

The online portfolio selection problem has been studied both empirically and theoretically. Most theoretical works with adversarial guarantees consider the *rebalancing* setting, pioneered by Cover [Cov91] and followed by a series of works [CO96, HSSW98, KV02, OLL17, LWZ18, MR22, ZAK22]. Differences to our setting are discussed as follows.

[7] Code is available at https://github.com/zhiyuzz/NeurIPS2022-Adaptive-Switching.

[8] W.l.o.g., assume $x_0 = x_1$.

1. First, the rebalancing setting forbids short selling (i.e., $x_{t,i} < 0$) and margin trading (i.e., borrowing cash to buy an asset), therefore the decision is modeled as a distribution $p_t \in \Delta(d)$ – the algorithm redistributes its wealth according to this distribution in each round. In contrast, our setting allows both[9], so we call it "unconstrained". Similar to the loss-regret trade-off in OLO, allowing margin trading introduces a *risk-return trade-off* in some sense: based on its own risk tolerance, one can trade off the best-case return with the worst-case loss on a Pareto-optimal frontier.
2. Related to the above, existing works consider *Constant Rebalanced Portfolios* (CRP, i.e., $p_t = p^* \in \Delta(d)$) as the benchmark class, and the goal is to lower bound the *ratio* of the growth rate of the considered algorithm to the growth rate of the benchmark. Here we consider unconstrained *Buy-and-Hold* (BAH) strategies as benchmarks, and we aim at an additive bound on the wealth. There have been discussions on the correct choice of benchmarks, but as suggested by a series of works [Cov91, HSSW98, BK99], a major weakness of CRPs is the incorporation of transaction costs: such benchmark strategies lose money due to constant rebalancing in every round, which makes the performance guarantee vacuous in certain cases. In contrast, BAH benchmarks do not suffer from this issue.
3. Finally, transaction costs can take many forms. Here we consider the special case that charges a fixed price *per share*. This is different from the *proportional transaction cost* in some prior works [BK99, Gof14], which is proportional to the *total value* of the transaction.

We also note that our Algorithm 7 for LEA with switching cost is essentially a comparator adaptive improvement of the *Exponentiated Gradient* (EG) algorithm adopted in [HSSW98]. Therefore, it can be applied to the rebalancing setting, following the same argument there.

D.3 Synthetic market

In this subsection, we present numerical results on synthetic markets.

Two algorithms are tested, our high dimensional OLO algorithm (Algorithm 6, i.e., "ours"), and the baseline from [ZCP22a] (its 1D version is surveyed as Algorithm 2 in Appendix A.1, and we extend it to high dimensions using the same coordinate-wise construction). Both algorithms require a confidence parameter C – they are both set to 1 following the practice of comparator adaptive algorithms [OP16, CLO22, ZCP22a]. A detailed justification is provided later.

As for the synthetic market, we let $G = 1$, $\lambda = 0.1$, and the market contains five assets with different return characteristics. Each $g_{t,i}$ is the summation of a i.i.d. noise, a periodic fluctuation and a constant trend. Specifically, we consider three different market return models. The first is

$$\begin{aligned}
g_{t,1} &= 0.4 \cdot \mathrm{Uniform}[-1,1] + 0.4\sin[(t/500)\cdot\pi] + 0.2,\\
g_{t,2} &= 0.5 \cdot \mathrm{Uniform}[-1,1] + 0.3\sin[(t/500+1/2)\cdot\pi] + 0.2,\\
g_{t,3} &= 0.6 \cdot \mathrm{Uniform}[-1,1] + 0.2\sin[(t/500+1)\cdot\pi] + 0.2,\\
g_{t,4} &= 0.7 \cdot \mathrm{Uniform}[-1,1] + 0.1\sin[(t/500+3/2)\cdot\pi] + 0.2,\\
g_{t,5} &= 0.8 \cdot \mathrm{Uniform}[-1,1] + 0.2.
\end{aligned}$$

The second model is

$$\begin{aligned}
g_{t,1} &= 0.2 \cdot \mathrm{Uniform}[-1,1] + 0.4\sin[(t/500)\cdot\pi] + 0.4,\\
g_{t,2} &= 0.3 \cdot \mathrm{Uniform}[-1,1] + 0.3\sin[(t/500+1/2)\cdot\pi] + 0.4,\\
g_{t,3} &= 0.4 \cdot \mathrm{Uniform}[-1,1] + 0.2\sin[(t/500+1)\cdot\pi] + 0.4,\\
g_{t,4} &= 0.5 \cdot \mathrm{Uniform}[-1,1] + 0.1\sin[(t/500+3/2)\cdot\pi] + 0.4,\\
g_{t,5} &= 0.55 \cdot \mathrm{Uniform}[-1,1] + 0.45.
\end{aligned}$$

The third model is the same as the second one, except we replace $g_{t,5}$ by

$$g_{t,5} = 0.5 \cdot \mathrm{Uniform}[-1,1] + 0.5.$$

For each market return model, we test both algorithms in 50 random trials, and the increased wealth $\sum_{\tau=1}^{t} \langle g_\tau, x_\tau \rangle - \lambda \sum_{\tau=1}^{t-1} \|x_\tau - x_{\tau+1}\|_1$ (mean $\pm$ std) is plotted in Figure 2, higher is better. In all three setting, our algorithm beats the baseline by a considerable margin, due to being a lot less conservative.

[9] Although we only consider the ideal case with zero interest rate on loans.

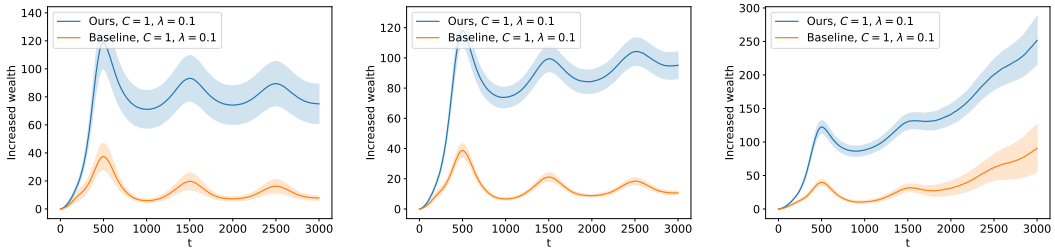

Figure 2: Synthetic market experiment with different market models. From left to right: the first, the second and the third market model.

Discussion on C We remark that setting $C = 1$ for both algorithms may create confusion. Let us give it a detailed justification.

As surveyed in the Introduction, comparator adaptive algorithms are called "parameter-free" due to historical reasons. As the name suggests, one may question the existence of *any* hyperparameter in such "parameter-free" algorithms. The classical rationale is the following: comparator adaptive regret bounds depend on the hyperparameter C logarithmically, whereas minimax regret bounds depend on the learning rate (and its inverse) linearly. In this regard, comparator adaptive algorithms are provably less sensitive to the correct setup, therefore as a rule of thumb, most practices [OP16, CLO22, ZCP22a] simply use $C = 1$ without requiring any domain knowledge. Such a default setup removes hyperparameter tuning, which is the most attractive feature of these algorithms in practice. Figure 2 shows the advantage of our algorithm when both algorithms are in this default, parameter-free implementation.

Nonetheless, for specific tasks like portfolio selection, tuning C can affect the actual performance one cares about (although violating the main purpose of comparator adaptivity). Intuitively, fixing the market, an aggressive trader with a worse strategy could make more profit than a conservative trader with a better strategy. Reflected in our experiment, since the market model does not depend on the invested amount, the baseline with a 5 times larger C simply obtains a 5 times larger increased wealth and beats our algorithm (at certain t), cf. Figure 3 (Left). Of course, one can also tune our algorithm with a correspondingly scaled C and beat the baseline again, cf. Figure 3 (Right), just like when both algorithms are in their parameter-free implementation.

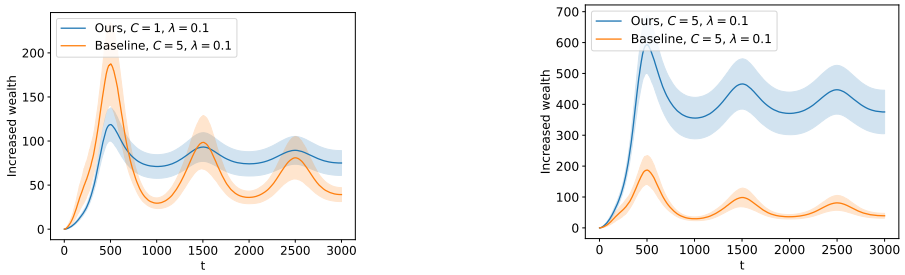

Figure 3: Synthetic market experiment with tuned C; not a parameter-free implementation. Left: only tuning the baseline. Right: tuning both our algorithm and the baseline.

Therefore, if tuning C is allowed, then comparing our algorithm to the baseline amounts to comparing two *algorithm classes* both parameterized by C. A skeptical reader may wonder if the superior performance of our algorithm in the parameter-free setting is due to the confidence encoding rather than a better algorithm design. That is, is it possible that a baseline with a larger C can consistently outperform our algorithm with $C = 1$? We provide evidence against this hypothesis, by increasing λ while keeping $C = 1$ for our algorithm and $C = 5$ for the baseline; results are plotted in Figure 4. It shows that even when the baseline is given an advantage ($C = 5$), our algorithm is still better at handling transaction costs due to an improved design. This is aligned with the superiority of our theoretical results.

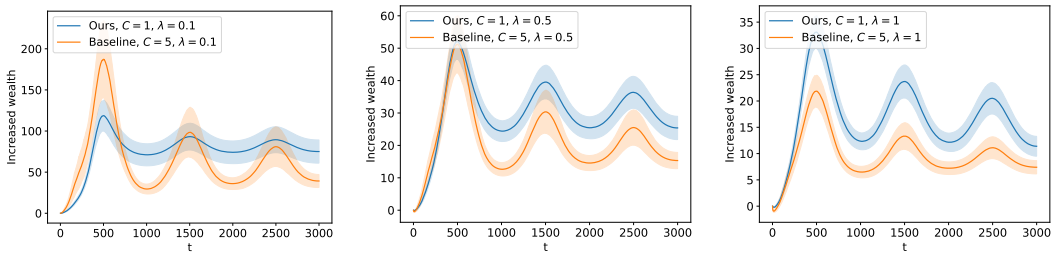

Figure 4: Synthetic market experiment with increasing λ. Left: $\lambda = 0.1$. Middle: $\lambda = 0.5$. Right: $\lambda = 1$. The baseline is given an advantage ($C = 5$), while our algorithm is in its default parameter-free implementation ($C = 1$). It shows our algorithm indeed handles transaction costs better.

D.4 Historical stock data

Finally, we present some preliminary numerical results on historical US stock data[10]. Eight stocks (Table 1) are considered on a time period of 5 years (1/1/2013 to 1/1/2018). Our algorithm trades once per day after the market closes, based on the closing price. We take the difference between the closing price on the $(t+1)$-th day and the closing price on the t-th day, and define it as the market vector g_t. The largest single day price change for any stock is less than \$15, therefore G is set in a posterior manner to 15. We consider a hypothetical broker that charges \$0.1 per share, therefore define $\lambda = 0.1$.

Table 1: List of considered stocks

Company	Symbol
Apple Inc.	AAPL
Berkshire Hathaway Inc. Class B	BRK.B
Meta Platforms Inc.	FB
Johnson & Johnson	JNJ
JPMorgan Chase & Co.	JPM
Microsoft Corporation	MSFT
Pfizer Inc.	PFE
Exxon Mobil Corporation	XOM

Same as the synthetic market experiment, we test our algorithm against the baseline from [ZCP22a]. Our algorithm is in its default parameter-free implementation ($C = 1$). However, setting $C = 1$ also for the baseline is too conservative, which means the baseline hardly makes any investment, making the comparison less interesting. Therefore we set $C = 10$ for the baseline, thus giving it an advantage at the beginning. In this way, the increased wealth of the two algorithms is roughly comparable.

We plot the results in Figure 5. Specifically, Figure 5 (Left) shows the increased wealth (in USD) over the considered time period. Figure 5 (Right) shows the cumulative amount of investment (in USD), which is the total net amount of cash the investor uses to buy stocks (i.e., increases when buying, and decreases when selling), plus the transaction costs paid to the broker. Before analyzing this result, we note that such a "cumulative investment" only makes sense in our setting, due to a fundamentally different mechanism compared to the rebalancing approach [Cov91]: in the latter, the investor is *self-financed*, i.e., it is given a certain budget at the beginning and never adds more money from external sources after that. In contrast, the investor in our setting can add more money at any time it wishes.

From the plot we can see that the baseline is more aggressive at the beginning, due to a much larger C. Therefore, it slightly makes more profit during 2013-2014. When the market oscillates and declines in 2015 and 2016, the two algorithms perform roughly the same, while the baseline has a lower risk due to holding a smaller portfolio at the time. However, the major difference starts after mid-2016, when the market grows rapidly. Our algorithm is able to identify this trend and quickly increase

[10] US stock price data is publicly available. We retrieved the data from Yahoo Finance website. `https://finance.yahoo.com/`

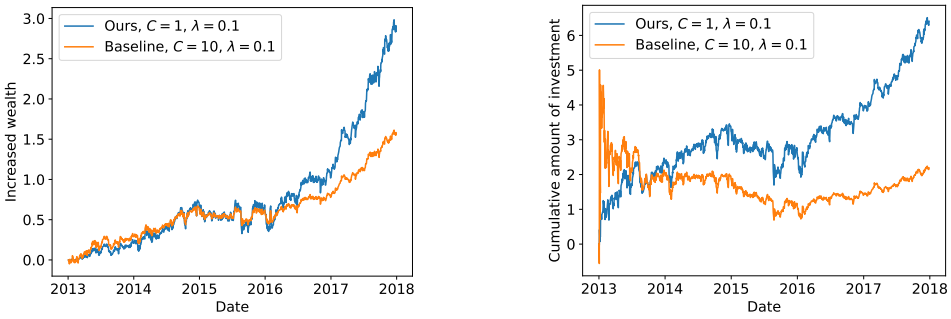

Figure 5: Experiment on historical US stock data. Left: the increased wealth of the two algorithms. Right: total amount of investment since the start of the experiment (1/1/2013), including the transaction costs paid to the broker.

the amount of investment. This brings a lot more profit than the baseline, which hardly recovers its confidence from the declining market in the previous year. Such an advantage of our algorithm is partly due to the better control of switching costs, and partly due to a better risk-return trade-off discussed in Appendix A.4.

Our experiment also shows a limitation of our unconstrained investment setting. Throughout this five year period, our algorithm invests a total amount of $\sim$\$6.5 (including the transaction costs), and makes a total profit of $\sim$\$3. However, in practice, one typically invests a lot more than this (let's say, \$10,000), and expect a similar rate of return. Our setting does not model such a budget explicitly; instead, it relies on the comparator adaptivity of the trading algorithms to increase the invested amount. Such a process can be slow, especially since we only consider trading once per day. Therefore, to use our algorithm in real trading situations, one has to tune the confidence parameter C to implicitly take his budget and tolerable risk into account. For example, using our algorithm with $C = 1000$ would result in investing \$6,500 throughout the five year period, and make a total profit of \$3,000. The connection of this approach to rebalancing could be an interesting direction for future works.

E Conclusion, limitation and future work

The present work investigates the design of comparator adaptive algorithms in the presence of switching costs. By carefully trading off these two opposite considerations, we propose a simple algorithm for OLO with switching costs, improving the suboptimal bound from our prior work [ZCP22a] to the optimal rate. Notably, the key idea of this algorithm is not guessed, but derived from a continuous-time analysis. Extensions lead to new results for comparator adaptive LEA.

Limitation and future work Our result requires a time-invariant λ, which could be generalized in future works. Different from [ZCP22a], we did not discuss applications to control theory, which is interesting on its own. Also, one may combine our portfolio selection approach with adversarial rebalancing and stochastic modeling, in order to further improve its practical performance.

More generally, through this paper we aim to demonstrate a key strength of the continuous-time PDE analysis – it makes the generalization of algorithmic structures much easier. Such an observation could open up exciting possibilities:

- Does this approach apply to other variants of the online learning problem?
- Can we use it to generalize other forms of adaptivity?
- Continuous-time potentials have been extensively studied under the framework of *potential theory* [Doo84]. Can we borrow techniques from there to further improve the workflow of algorithm design?