Online Optimization with Untrusted Predictions

Daan Rutten*,[1], Nicolas Christianson[2], Debankur Mukherjee[1], and Adam Wierman[2]

[1]Georgia Institute of Technology
[2]California Institute of Technology

February 9, 2022

Keywords — online algorithms, online optimization, competitive analysis, metrical task systems

Abstract

We examine the problem of online optimization, where a decision maker must sequentially choose points in a general metric space to minimize the sum of per-round, non-convex hitting costs and the costs of switching decisions between rounds. The decision maker has access to a black-box oracle, such as a machine learning model, that provides untrusted and potentially inaccurate predictions of the optimal decision in each round. The goal of the decision maker is to exploit the predictions if they are accurate, while guaranteeing performance that is not much worse than the hindsight optimal sequence of decisions, even when predictions are inaccurate. We impose the standard assumption that hitting costs are globally α-polyhedral. We propose a novel algorithm, Adaptive Online Switching (AOS), and prove that, for any desired $\delta > 0$, it is $(1 + 2\delta)$-competitive if predictions are perfect, while also maintaining a uniformly bounded competitive ratio of $2^{\tilde{\mathcal{O}}(1/(\alpha\delta))}$ even when predictions are adversarial. Further, we prove that this trade-off is necessary and nearly optimal in the sense that *any* deterministic algorithm which is $(1 + \delta)$-competitive if predictions are perfect must be at least $2^{\tilde{\Omega}(1/(\alpha\delta))}$-competitive when predictions are inaccurate.

1 Introduction

We consider online optimization with switching costs in a general metric space (M, d) wherein, at each time t, a decision-maker observes a non-convex *hitting cost* function $f_t : M \to [0, \infty]$ and must decide upon some $x_t \in M$, paying $f_t(x_t) + d(x_t, x_{t-1})$, where d characterizes the *switching cost*. Moreover, we assume that the decision maker has access to an *untrusted prediction* $\tilde{x}_t$ of the optimal decision during each round, such as the decision suggested by a black-box AI tool for that round.

Online optimization is a problem with applications to many real-world problems such as capacity scaling for data centers [33, 36], electrical vehicle charging [24], portfolio management [16], network routing [8], load balancing [47], video streaming [23], and thermal management of circuits [48, 49]. In many of these applications, there is a penalty for switching decisions too much: the goal of a decision maker is not just to minimize the hitting cost functions that arrive in each round, but to also minimize the switching cost between rounds. The switching cost acts to "smooth" the sequence of decisions. Online optimization with switching costs has received considerable attention in the learning, networking, and control communities in recent years [3, 19, 21, 31, 32, 44]. Moreover, the

*Email: drutten@gatech.edu

crux of many fundamental problems in online algorithms, such as the k-server problem [26], metrical task systems [10, 11], and online convex body chasing [7, 43] is to effectively deal with switching costs.

The bulk of the literature on online optimization with switching costs has sought to design *competitive* algorithms for the task, i.e., algorithms with finite competitive ratios. At this point, competitive algorithms for scenarios with both convex and non-convex hitting costs have been developed [14, 35, 50]. However, while competitive analysis yields strong performance guarantees, it has often been criticized as being unduly pessimistic, since algorithms are characterized by their *worst-case* performance, while worst-case conditions may never actually occur in practice. As a result, an algorithm with an improved competitive ratio may not actually lead to better performance in realistic scenarios.

On the other hand, many real-world applications have access to vast amounts of historical data which could be leveraged by modern black-box AI tools in order to achieve significantly improved performance in the typical case. For example, in the context of capacity scaling for data centers, the historical data reveals reoccurring patterns in the weekly load of a data center. AI models that are trained on these historical patterns can potentially outperform competitive algorithms in the typical case. This approach has been successfully used by Google in data center cooling [17].

Making use of modern black-box AI tools is potentially transformational for online optimization; however, such machine-learned algorithms fail to provide any uncertainty quantification and thus do not have formal guarantees on their worst-case performance. As such, while their performance may improve upon competitive algorithms in typical cases, they may perform arbitrarily worse in scenarios where the training examples are not representative of the real world workloads due to, e.g., distribution shift. This represents a significant drawback when considering the use of AI tools for safety-critical applications.

A challenging open question is whether it is possible to provide guarantees that allow black-box AI tools to be used in safety-critical applications. This paper aims to provide an algorithm that can achieve the best of both worlds – making use of black-box AI tools to provide good performance in the typical case while integrating competitive algorithms to ensure formal worst-case guarantees. We formalize this goal using the notions of *consistency* and *robustness* introduced by [37] and used in the emerging literature on untrusted advice in online algorithms [22, 28, 29, 37, 38, 40, 42, 46]. We consider that the online algorithm is given untrusted advice/predictions, e.g., the output of a black-box AI tool. The predictions are fully arbitrary, i.e., we impose no statistical assumptions on the predictions, and the decision maker has no *a priori* knowledge of the predictions' accuracy. The goal of the decision maker is to be both *consistent* – that is, to achieve performance comparable to that of the predictions if they are accurate, while remaining *robust*, i.e., having cost that is never much worse than the hindsight optimal, even if predictions are completely inaccurate. Thus, an algorithm that is consistent and robust is able to match the performance of the black-box AI tool when the predictions are accurate while also ensuring a worst-case performance bound (something black-box AI tools cannot provide).

1.1 Our Contributions

We make four main contributions in this paper. *First,* we introduce a new algorithm for online non-convex optimization in a general metric space with untrusted predictions, *Adaptive Online Switching* (AOS), and provide formal bounds on its consistency and robustness. Our analysis shows that AOS can be used in combination with a black-box AI tool to match the performance of the black-box AI while also ensuring provable worst-case guarantees.

The AOS algorithm works as follows. At each time t, AOS either follows the predictions $\tilde{x}_t$ or adopts a robust strategy that does not adapt to the predictions, and adaptively switches between these two. The challenge in the design of AOS is that, on the one hand, switching must be infrequent in order to limit the switching cost, but, on the other hand, switching must be frequent enough to ensure that the algorithm does not get stuck following a suboptimal sequence of decisions from either the predictions or the robust strategy.

We characterize the robustness and consistency of AOS using the *competitive ratio*, which is the worst-case ratio between the cost of the algorithm and the hindsight optimal sequence of decisions (see Definition 2.4). Theorem 3.1 proves that, if the hitting costs are globally α-polyhedral (see Definition 2.1), then the competitive ratio $\mathrm{CR}(\eta)$ of AOS is a function of the accuracy η of the predictions and is at most

$$\mathrm{CR}(\eta) \leq \min\left\{(1+2\delta)(1+2\eta), \frac{12+o(1)}{\delta}\left(\frac{2}{\alpha+\delta(1+\alpha)}\right)^{2/(\alpha\delta)}\right\}. \tag{1}$$

Here, η is an appropriate measure of the accuracy of the predictions (see Definition 2.3) and relates to the total metric distance between the prediction $\tilde{x}_t$ and the optimal decision. Note that, even though the competitive ratio is a function of the accuracy, the algorithm is oblivious to it. The competitive ratio is the minimum of two terms. If the predictions are accurate, i.e., if $\eta = 0$, then the competitive ratio of AOS is $1 + 2\delta$. In other words, AOS is $(1 + 2\delta)$-consistent and almost reproduces the hindsight optimal sequence of decisions if the predictions are accurate. However, even if predictions are completely inaccurate, i.e., if $\eta = \infty$, the competitive ratio of AOS is uniformly bounded by $\frac{12+o(1)}{\delta}\left(\frac{2}{\alpha+\delta(1+\alpha)}\right)^{2/(\alpha\delta)}$. The trade-off between consistency and robustness is characterized by the confidence hyperparameter δ, where the robustness bound depends exponentially on both δ and α. As a proof technique, the conventional potential function approach fails due to the non-convexity of the problem and the incorporation of predictions. Hence, significant novelty in the technique is required for the proof of Theorem 3.1 (see Remark 3.3 for a discussion).

We complement the theoretical analysis of AOS's performance with empirical validation in Appendix A, where we report on experiments using AOS to "robustify" the decisions of a machine-learned algorithm for the problem of optimal microgrid dispatch with added noise and distribution shift in the predictions. These experiments confirm that AOS effectively bridges the good average-case performance of black-box AI with worst-case robust performance.

Second, we prove that the exponential trade-off between consistency and robustness observed in AOS is, in fact, necessary. Thus, AOS achieves the order-optimal trade-off between consistency and robustness. More formally, Theorem 3.2 proves that any algorithm that is $(1 + \delta)$-consistent must be at least

$$\frac{\alpha\delta}{4}\left(\frac{2}{\alpha+\delta(1+\alpha)}\right)^{\frac{2-\alpha(1-\delta^2)}{\alpha\delta(1+\delta)}} - \mathcal{O}(1) \quad \text{-robust.} \tag{2}$$

This lower bound shows the hardness of the problem brought about by non-convexity of hitting costs, and in particular implies that no algorithm can be 1-consistent and robust.

Third, we characterize the importance of memory for algorithms seeking to use untrusted predictions. Interestingly, AOS requires full memory of all predictions. This is a stark contrast with well-known memoryless algorithms for online optimization with switching costs, which do not make use of any information about previous hitting costs or actions. We prove in Theorem 3.4 that this contrast is no accident and that memory is needed in order to simultaneously achieve robustness and consistency. Thus, *memory is necessary to benefit from untrusted predictions.*

Fourth, we consider an important special case when the metric space is $M = \mathbb{R}$ and each function is convex and show that it is possible to provide an improved trade-off between robustness and consistency using a memoryless algorithm in this special case. The one-dimensional online convex optimization setting has received significant attention in the literature, see [1, 9]. In this context, we introduce a new algorithm, called Adaptive Online Balanced Descent (AOBD), which is inspired by Online Balanced Descent [12]. AOBD has two tunable hyperparameters $\bar{\beta} \geq \underline{\beta} > 0$ that represent the confidence of the decision maker in the predictions. Theorem 3.5 proves that the competitive ratio $\mathrm{CR}(\eta)$ of AOBD is at most

$$\mathrm{CR}(\eta) \leq \min\left\{\left(1 + (2 + \bar{\beta}^{-1})\underline{\beta}\right)(1 + 2\eta), 1 + (2 + \underline{\beta}^{-1})\bar{\beta}\right\}. \tag{3}$$

The competitive ratio of AOBD has a similar structure to AOS, but improves the robustness bound significantly by taking advantage of the additional structure available compared to the general non-convex case. In particular, if $\bar{\beta} = 1/\delta$ and $\underline{\beta} = \delta/(2+\delta)$ for $\delta \leq 2$, then AOBD is $(1+\delta)$-consistent and $1 + 3/\delta + 2/\delta^2$-robust. The result is complemented by a lower bound in Theorem 3.6 that proves that for $0 < \delta < 1/2$, any algorithm that is $(1+\delta)$-consistent must be at least $1/(2\delta)$-robust.

1.2 Related Work

This paper connects and contributes to three growing literatures: (i) online optimization without predictions, (ii) online optimization with trusted predictions, and (iii) online algorithms with untrusted predictions. We discuss each of these in the following.

Online Optimization without Predictions. The problem of "smoothed" online convex optimization was first introduced by Lin et al. in (2012b) in the case of a one-dimensional action space. The one-dimensional version of the problem was further studied by Bansal et al. (2015), who designed a 2-competitive algorithm and later by Antoniadis and Schewior (2017), who prove a matching lower bound. The algorithm in [9] is deterministic, but consumes memory proportional to the number of time steps to date. The authors also design a memoryless 3-competitive algorithm and prove a matching lower bound. Andrew et al. (2013) investigate the compatibility between competitive ratio and sublinear regret, and prove that no algorithm may achieve both simultaneously. Beyond the one-dimensional setting, Chen et al. (2018) prove that in $\mathbb{R}^d$ for general convex functions, the competitive ratio of any online algorithm is at least $\Omega(\sqrt{d})$ with ℓ_2 switching costs and at least $\Omega(d)$ with ℓ_∞ switching costs. The authors introduce the Online Balanced Descent (OBD) algorithm which achieves a competitive ratio of $3 + O(1/\alpha)$ for α-polyhedral functions and ℓ_2 switching costs. Later, Goel and Wierman (2019) prove that OBD is also $3 + O(1/\alpha)$ competitive if the switching cost is the squared ℓ_2 norm and the hitting cost functions f_t are m-strongly convex. In this context, the authors prove that any online algorithm must have a competitive ratio of at least $\Omega(m^{-1/2})$ and introduce two algorithms, Greedy OBD (G-OBD) and Regularized OBD (R-OBD), that both have a competitive ratio of $O(m^{-1/2})$.

Note that all results in the case of online *non-convex* optimization are very limited. Lin et al. (2020) provide an algorithm, in the setting with no perfect predictions of future hitting costs, that greedily follows the minimizers and achieves competitive ratio $1 + 2/\alpha$. Later, Zhang et al. (2021) refined this result, proving that the greedy algorithm is $\max\{1, 2/\alpha\}$-competitive.

Online Optimization with Trusted Predictions. A separate but related line of work seeks to provide improved competitive bounds through the use of trusted predictions. Most typically, such predictions are assumed to have no error, e.g., [31, 35], but some papers assume known stochastic predictions error, e.g., [12, 13, 15]. An early paper in this area by Chen et al. (2015) introduces the most common prediction error model, which is a stochastic prediction model (a colored noise model).

Chen et al. (2016) consider a similar stochastic prediction model, but weaken the assumptions on the hitting cost functions. A different style of algorithm was introduced by Li et al. (2018), which assumes that the next w functions are known perfectly. Later, Li and Li (2020) generalize this to imperfect predictions.

The case of online non-convex optimization has proven difficult and the only result we are aware of is from Lin et al. (2020), which assumes perfect predictions and provides two sufficient conditions on the structure of f_t and the switching cost (an order of growth and an approximate triangle inequality condition) such that their algorithm, Synchronized Fixed Horizon Control (SFHC), achieves a competitive ratio of $1 + O(1/w)$.

Online Algorithms with Untrusted Predictions. This work is the first to look at the use of untrusted predictions in the setting of online optimization. The framework for studying untrusted predictions that we adapt was introduced by Lykouris and Vassilvitskii (2018) in the context of caching and by Mahdian et al. (2012) in the context of load balancing and facility location. At this point, the framework has received significant attention and has been applied to bipartite matching [28], bloom-filters [40], frequency estimation [22], capacity scaling in data-centers [42], and more [29, 38, 46].

Recent work in [5] proposes combining individually robust and consistent algorithms by applying classic online algorithms for the cow problem [18] and the k-experts problem [10]. However, their results do not yield tight bounds on robustness and consistency in our setting. If $\alpha < 2$, their deterministic algorithm achieves 9-consistency and $\frac{18}{\alpha}$-robustness, and their randomized algorithm achieves $(1 + \delta)$-consistency and $\frac{2(1+\delta)}{\alpha}$-robustness in expectation but has a large additive factor of $\mathcal{O}(D/\delta)$ on both bounds, where D is the diameter of the metric space. However, our main result shows that it is possible to obtain robustness and near-perfect consistency *without* an additive factor.

2 Model

We examine the problem of online non-convex optimization with switching costs on an arbitrary metric space (M, d). In this problem, the decision maker starts at an arbitrary point $x_0 \in M$ and is presented at each time $t = 1, \ldots, T$ with a non-convex function $f_t : M \to [0, \infty]$. The decision maker must then choose an $x_t \in M$ and pay cost $f_t(x_t) + d(x_t, x_{t-1})$, i.e., the sum of the *hitting cost* of x_t and the *switching cost* from x_{t-1} to x_t, determined by the metric d. The goal of the decision maker is to solve

$$\begin{aligned} &\min_{x_t, 1 \le t \le T} \sum_{t=1}^{T} \left(f_t(x_t) + d(x_t, x_{t-1}) \right) \\ &\text{s.t.} \quad x_t \in M \qquad \text{for } 1 \le t \le T. \end{aligned} \tag{4}$$

Note that the problem is presented in an online fashion: at time t, the algorithm only knows $f_1, \ldots, f_t$, but does not know $f_{t+1}, \ldots, f_T$ or the finite time horizon T. We call a collection $(x_0, f_1, \ldots, f_T, T)$ an *instance* of the online non-convex optimization problem. The performance of the decision maker is compared to the hindsight optimal sequence of decisions, defined as

$$(o_1, \ldots, o_T) \in \arg\min_{y_1, \ldots, y_T \in M} \sum_{t=1}^{T} \left(f_t(y_t) + d(y_t, y_{t-1}) \right), \tag{5}$$

where we assume that the minimizer exists. We denote by $\textsc{Opt}(t) := f_t(o_t) + d(o_t, o_{t-1})$ the cost of the hindsight optimal algorithm at time t.

We would like to emphasize the generality of our model. The decisions take values in an arbitrary metric space and the hitting cost functions can be non-convex, allowing our results to be applied to online decision-making in a variety of settings, including non-Euclidean and function spaces, as well as to problems where decisions are inherently discrete in nature such as right-sizing data centers [2]. Throughout, unless otherwise mentioned, we assume that the hitting cost functions are globally α-polyhedral for some $\alpha > 0$, defined as follows.

Definition 2.1. *We say that a function $f_t : M \to [0, \infty]$ is **globally α-polyhedral** if it has a unique minimizer $v_t \in M$, and, for all $x \in M$,*

$$f_t(x) \geq f_t(v_t) + \alpha \cdot d(x, v_t). \tag{6}$$

The assumption that hitting cost functions are α-polyhedral is standard in the literature on online optimization with switching costs [14, 35, 50]. This type of structural assumption on hitting costs is in fact necessary to ensure the existence of a competitive algorithm for online optimization with switching costs on general metric spaces.[1] The assumption that the switching cost is a metric is also crucial to our results; as we show in Appendix B, it is impossible to achieve simultaneous robustness and non-trivial consistency when the switching cost is a Bregman divergence, another common choice of switching cost, of which the squared Euclidean norm is a special case [20].

Remark 2.2. The fact that the hitting cost may be non-convex introduces new algorithmic challenges. For example, one cannot simply interpolate two points to obtain a convex combination of the hitting cost as done in state-of-the-art literature on online optimization without predictions [9, 14, 20]. Rather than focusing on the computational challenge of how to solve non-convex optimization problems offline, we assume that we have an oracle that is able to solve these problems and focus on the algorithmic challenges for online decision making.

Untrusted Predictions. We assume that the decision maker has access to potentially inaccurate predictions of the hindsight optimal sequence of decisions. At time t, before choosing an action, the decision maker receives a suggested action $\tilde{x}_t \in M$. We want to emphasize that these predictions are of the *optimal decisions*, and not of the hitting cost functions, as has been studied previously [12, 13, 30, 31, 35]. This new model captures scenarios where, for example, a black-box machine learning algorithm is available and outputs $\tilde{x}_t$ as the suggested action for time t. Alternatively, $\tilde{x}_t$ could be the result of using imperfect forecasts of future hitting cost functions via a look-ahead algorithm for online optimization such as SFHC [35].

The accuracy of the predictions is measured in terms of the distance to the hindsight optimal sequence of decisions, which we quantify as follows.

Definition 2.3. *Consider an instance $(x_0, f_1, \dots, f_T, T)$ for the online non-convex optimization problem and let $\tilde{x} = (\tilde{x}_1, \dots, \tilde{x}_T) \in M^T$ be a prediction sequence. We say that the prediction $\tilde{x}$ is **η-accurate** for the instance if*

$$\sum_{t=1}^{T} d(o_t, \tilde{x}_t) \leq \eta \sum_{t=1}^{T} \textsc{Opt}(t), \tag{7}$$

where o_t is the optimal sequence of decisions as defined in (5)*.*

[1] If hitting costs are allowed to be arbitrary, or if the only imposed assumption is that hitting costs are convex, then the problem class contains convex body chasing (CBC) as a special case [43]. The competitive ratio of any algorithm for CBC on any d-dimensional normed vector space is $\Omega(\sqrt{d})$, so no algorithm can be competitive for CBC on an infinite-dimensional vector space.

Note that the accuracy is normalized in terms of the total cost of the hindsight optimal algorithm to make the notion scale-invariant. For example, if the metric d is doubled, then both the left-hand side of (7) and the optimal cost double, and hence, η stays the same. This is natural since the quality of the predictions in this case does not change.

Defining Consistency and Robustness. We measure the performance of an algorithm using the competitive ratio, which is a function of the accuracy in our setting.

Definition 2.4. *Let $\mathcal{A}$ be an algorithm for the online non-convex optimization problem in* (4) *that adapts to the predictions. The* ***competitive ratio*** *of $\mathcal{A}$ is $\mathrm{CR}(\eta)$, or $\mathcal{A}$ is $\mathrm{CR}(\eta)$-**competitive**, if*

$$\sum_{i=1}^{T} (f_t(x_t) + d(x_t, x_{t-1})) \leq \mathrm{CR}(\eta) \cdot \sum_{t=1}^{T} \textsc{Opt}(t), \tag{8}$$

for all instances $(x_0, f_1, \ldots, f_T, T)$ and all η-accurate predictions $\tilde{x}$.

The aim of this work is to design an algorithm for which the competitive ratio improves with the quality of the predictions. We quantify this in terms of the notions of *consistency* and *robustness*, which have recently emerged as important measures for the ability of algorithms to effectively make use of untrusted predictions in online settings [22, 28, 29, 37, 38, 40, 42, 46]. In particular, we strive for the following desiderata.

Definition 2.5. *Let $\mathcal{A}$ be an algorithm for the online non-convex optimization problem in* (4) *and let $\mathrm{CR}(\eta)$ be its competitive ratio when it has access to η-accurate predictions. We say that $\mathcal{A}$ is*

1. *$(1+\delta)$-**consistent** if $\mathrm{CR}(0) \leq 1 + \delta$;*
2. *γ-**robust** if $\mathrm{CR}(\eta) \leq \gamma$ for any $\eta \in [0, \infty]$.*

An algorithm with these qualities of consistency and robustness is guaranteed to have near-optimal performance when predictions are perfect, along with a constant competitive ratio even when predictions are arbitrarily bad. If predictions are, for example, the decisions made by a black-box AI algorithm, a robust and consistent algorithm ensures a worst-case performance guarantee without having to sacrifice the AI algorithm's typically excellent performance. Moreover, the algorithm we propose in this work has performance that *degrades gracefully in η*: that is, even if predictions are not exactly perfect but are near-optimal, our algorithm will still achieve near-optimal cost.

3 Main Results

The desiderata of consistency and robustness are nontrivial to achieve simultaneously, but it is straightforward to obtain consistency and robustness individually. On the one hand, an algorithm that blindly follows the predictions $\tilde{x}_t$ is 1-consistent, but not robust. On the other hand, an algorithm that follows the minimizers v_t is $\max\{2/\alpha, 1\}$-robust [50], but not consistent. Hence, our goal in this section is to combine the features of both algorithms to construct an algorithm that is both consistent and robust.

A natural algorithm to consider is a 'switching type' algorithm, which dynamically chooses between following the minimizers and the predictions. These switching-type algorithms have already been used successfully in general problem settings [5]. In fact, due to the non-convexity of the hitting cost functions, any reasonable algorithm must be a switching-type algorithm, since the cost of any state in between the minimizer v_t and the prediction $\tilde{x}_t$ is generally unrelated to and potentially much higher than the hitting cost at v_t or $\tilde{x}_t$ itself. The difficulty of designing these algorithms in our case lies in the notion of consistency, which dictates that the cost of the algorithm must only be a small fraction higher than the cost of the predictions. This requires a careful design of the algorithm, which should only switch away from the prediction if it holds a strong conviction that robustness would otherwise be violated.

Algorithm 1 Adaptive Online Switching

1: Let $p_t := \arg\min_{p\in M}(f_t(p) + d(p, p_{t-1}) + d(p, \tilde{x}_t))$, $\textsc{Rob}(t) := f_t(v_t) + d(v_t, v_{t-1})$, and $\textsc{Adv}(t) := f_t(p_t) + d(p_t, p_{t-1})$ for $t = 1, 2, \dots, T$.
2: $T_1 \leftarrow 1,\ t \leftarrow 1$
3: **for** $k = 1, 2, \dots$ **do**
4:   **while** $\sum_{i=T_k}^{t-1} \textsc{Adv}(i) + \textsc{Rob}(t) + d(p_{t-1}, v_{t-1}) + d(v_t, p_t) \geq (1+\delta)\sum_{i=T_k}^{t} \textsc{Adv}(i)$ **do**
5:     $x_t \leftarrow p_t,\ t \leftarrow t+1$ and stop if $t = T+1$
6:   **end while**
7:   $M_k \leftarrow t$ ▷ *Start following the minimizers*
8:   $x_t \leftarrow v_t,\ t \leftarrow t+1$ and stop if $t = T+1$
9:   **while** $\sum_{i=M_k+1}^{t} \textsc{Rob}(i) + d(v_t, p_t) - d(v_{M_k}, p_{M_k}) \leq (1+\delta)\sum_{i=M_k+1}^{t} \textsc{Adv}(i) + \delta \sum_{i=T_k}^{t} \textsc{Adv}(i)$ **do**
10:     $x_t \leftarrow v_t,\ t \leftarrow t+1$ and stop if $t = T+1$
11:   **end while**
12:   $T_{k+1} \leftarrow t$ ▷ *Start following the predictions*
13:   $x_t \leftarrow p_t,\ t \leftarrow t+1$ and stop if $t = T+1$
14: **end for**

We propose Algorithm 1, named *Adaptive Online Switching* (AOS), which adaptively switches between following the minimizers and filtered versions of the predictions in a manner that ensures both robustness and consistency. The conditions in lines 4 and 9 dictate when the algorithm should switch and are carefully designed to simultaneously guarantee consistency and robustness. The AOS algorithm has a hyperparameter $\delta > 0$, which represents the confidence in the predictions. The algorithm and its performance depend explicitly on δ.[2] The next theorem shows that the parameter δ conveniently controls the trade-off between performance in the typical case when η is small and the worst-case when η is large, in hindsight.

Theorem 3.1. *Let* $\mathrm{CR}(\eta)$ *be the competitive ratio of AOS (Algorithm 1) and assume that* $\frac{2}{\alpha\delta} \in \mathbb{N}$. *Then,*

$$\mathrm{CR}(\eta) \leq \min\left\{(1+2\delta)(1+2\eta), \frac{12+o(1)}{\delta}\left(\frac{2}{\alpha+\delta(1+\alpha)}\right)^{2/(\alpha\delta)}\right\}. \tag{9}$$

We note that the assumption that $2/(\alpha\delta) \in \mathbb{N}$ is without loss of generality, but it prevents rounding symbols from appearing in the notation. The asymptotic notation $o(1)$ refers to the asymptotic regime $\alpha, \delta \to 0$. Note also that δ *is essentially upper bounded by* $1/\alpha$, otherwise, even if the predictions are perfect, the competitive ratio exceeds $\max\{2/\alpha, 1\}$, which is the competitive ratio of the greedy algorithm that always follows the minimizers [50]. The competitive ratio of AOS is the minimum of two terms. If the predictions are accurate, i.e., η is close to 0, then the competitive ratio will be close to $1 + 2\delta$. In particular, this means that the AOS algorithm is $(1 + 2\delta)$-consistent. Moreover, even for inaccurate predictions, i.e., as $\eta \to \infty$, the AOS algorithm

[2] An interesting and challenging topic for future work is to adaptively learn δ over time in order to optimize performance.

has a bounded competitive ratio, implying robustness. These simultaneous properties of AOS enable its use to filter the decisions of untrusted, black-box AI algorithms, yielding worst-case performance guarantees without giving up good average-case performance in safety-critical settings. To validate the performance guarantees on AOS offered by Theorem 3.1, we detail in Appendix A the results of experiments using AOS to filter the decisions of a machine-learned algorithm for the problem of optimal microgrid dispatch, which confirm that AOS can effectively bridge good average-case performance with worst-case guarantees in case predictions are inaccurate.

The confidence hyperparameter δ describes how much faith the algorithm designer has in the predictions and consequently influences the performance guarantees. As mentioned before, δ describes the trade-off between the performance in the cases when predictions are accurate and completely inaccurate, respectively. Theorem 3.1 explicitly describes the trade-off between these two scenarios and interpolates to any scenario in between. Note that the robustness bound increases exponentially in the confidence hyperparameter δ and in α. However, as the next theorem shows, due to the non-convexity of the hitting costs, the exponential dependence on both α and δ is, in fact, necessary.

Theorem 3.2. *Fix any $\delta > 0$, and assume that $(2 - \alpha(1 - \delta^2))/(\alpha\delta(1 + \delta)) \in \mathbb{N}$. Let $\mathcal{A}$ be any deterministic online algorithm for the non-convex optimization problem in (4). If there exists $\varepsilon < \delta$ such that $\mathcal{A}$ is $(1 + \varepsilon)$-consistent, then $\mathcal{A}$ is at least L-robust with*

$$L = \frac{\alpha\delta}{4}\left(\frac{2}{\alpha + \delta(1 + \alpha)}\right)^{\frac{2-\alpha(1-\delta^2)}{\alpha\delta(1+\delta)}} - \mathcal{O}(1). \tag{10}$$

Note that the assumption that $(2 - \alpha(1 - \delta^2))/(\alpha\delta(1 + \delta)) \in \mathbb{N}$ is again without loss of generality, but prevents rounding symbols from appearing in the notation. Theorem 3.2 highlights that any deterministic online algorithm must experience a trade-off between consistency and robustness. In particular, no deterministic algorithm can be 1-consistent and robust. So far, it has been rare in the literature to have provably optimal guarantees for the trade-off between robustness and consistency [42, 46], and even more unique if the bound is actually achievable. The proofs of Theorem 3.1 and 3.2 are provided in Appendices C and D, respectively.

Remark 3.3. It is worthwhile to highlight that the proof of Theorem 3.1 does not follow the conventional potential function approach for competitive analysis, since this approach is hindered by its dependence on both the optimal trajectory and the trajectory of the predictions (as opposed to only the optimal trajectory in conventional competitive analysis). The proof instead views the problem as a non-convex optimization problem, where the goal of the adversary is to maximize the cost of the AOS algorithm while minimizing the cost of the optimal trajectory simultaneously. The optimization problem is then reduced to a linear optimization problem by a non-linear transformation. Finally, we apply strong duality to upper bound the linear problem by a feasible solution to its dual. Our proof of Theorem 3.2 similarly uses an optimization problem formulation. The problem is then lower bounded by a feasible solution to the primal.

3.1 Limitations of Memoryless Algorithms

Algorithm 1 requires the accumulated cost of following the minimizers and the predictions until the current time to be stored in memory, in contrast to most previous algorithms for online optimization. These algorithms have typically been memoryless, i.e., the decision at time t is only based on x_{t-1} and the current hitting cost function f_t [14, 20, 50]. In light of this discrepancy, it is natural to ask the following question: *can there exist a memoryless algorithm that is $(1 + \delta)$-consistent and robust?*

We answer this question in the negative in Theorem 3.4. We show that, under mild assumptions of scale- and rotation-invariance (see Appendix E for the precise definitions), any memoryless algorithm achieving finite robustness must have consistency of $\Omega\left(\alpha^{-1/2}\right)$.

Theorem 3.4. *Let $\mathcal{A}$ be any memoryless and scale- and rotation-invariant algorithm, and let $\alpha < 1/4$. The next two statements are mutually exclusive:*

(i) $\mathcal{A}$ is γ-robust for $\gamma < \infty$.

(ii) $\mathcal{A}$ is c-consistent with

$$c < \frac{1}{\sqrt{8\alpha}} - o\left(\frac{1}{\sqrt{\alpha}}\right). \tag{11}$$

The proof of Theorem 3.4 is provided in Appendix E. Theorem 3.4 shows that there is a fundamental lower bound of $\Omega\left(\alpha^{-1/2}\right)$ on the consistency of any robust, memoryless algorithm. In particular, as $\alpha \to 0$, this lower bound grows arbitrarily large and hence simultaneous robustness and near-perfect consistency cannot be expected from any memoryless algorithm. Thus, memoryless algorithms cannot match the performance of AOS.

Notably, the lower bound holds even for online *convex* optimization and matches the best known upper bound of $\mathcal{O}\left(\alpha^{-1/2}\right)$ on the competitive ratio of algorithms for online convex optimization without predictions [34]. In conjunction with the lower bound in Theorem 3.4, this implies that, within the restricted class of memoryless, scale- and rotation-invariant algorithms, optimal robustness and consistency can be achieved while ignoring predictions. In other words, memory is a fundamental requirement for any algorithm to benefit from unreliable predictions.

3.2 A Memoryless Algorithm for Online Convex Optimization in One Dimension

Although the previous section's lower bound gave a pessimistic view of the prospect of designing memoryless algorithms that achieve robustness and consistency, it leaves open the question of whether it is possible to do better when there are stronger assumptions on the problem. We answer this question in the affirmative for the following special case:

1. The metric space is $M = \mathbb{R}$ is one-dimensional and $d(x, y) = |x - y|$ is the Euclidean distance;

2. Each function $f_t : \mathbb{R} \to [0, \infty]$ is convex.

We assume the above two conditions throughout this subsection. Note that we do *not* assume that f_t is globally α-polyhedral, as opposed to the previous section and the general problem description.

The one-dimensional case has been considered in depth in the literature as an important problem in its own right, and specialized algorithms for this case are known [6, 9, 33]. We use the Online Balanced Descent algorithm [14] as inspiration for the robust algorithm.[3] However, since each f_t is now convex, unlike the AOS algorithm, the algorithm does not need to follow either the predictions or the robust algorithm exactly. Instead, it smoothly interpolates between the predictions and the robust algorithm.

Algorithm 2, named *Adaptive Online Balanced Descent* (AOBD), describes our proposed algorithm. AOBD has two confidence hyperparameters $\bar{\beta} \geq \underline{\beta} > 0$, which represent the confidence of the algorithm designer in the accuracy of the predictions. A higher ratio of $\bar{\beta}/\underline{\beta}$ means high confidence

[3]Note that extensions of OBD with improved competitive ratios are known (for example G-OBD and R-OBD [20]), but these do not work well in the case where the metric is the Euclidean distance.

Algorithm 2 Adaptive Online Balanced Descent

1: **for** $t = 1, 2, \ldots, T$ **do**
2: Observe f_t and $\tilde{x}_t$
3: Let $x(\lambda) := (1-\lambda)x_{t-1} + \lambda v_t$ for $\lambda \in [0, 1]$
4: Let $\underline{\lambda} \in [0, 1]$ be such that $|x(\underline{\lambda}) - x_{t-1}| = \underline{\beta} f_t(x(\underline{\lambda}))$ or $\underline{\lambda} = 1$ if such a $\underline{\lambda}$ does not exist
5: Let $\bar{\lambda} \in [0, 1]$ be such that $|x(\bar{\lambda}) - x_{t-1}| = \bar{\beta} f_t(x(\bar{\lambda}))$ or $\bar{\lambda} = 1$ if such a $\bar{\lambda}$ does not exist
6: $p_t \leftarrow \arg\min_{p \in M} f_t(p) + d(p, p_{t-1}) + d(p, \tilde{x}_t)$
7: $\lambda \leftarrow \arg\min_{\lambda \in [\underline{\lambda}, \bar{\lambda}]} |x(\lambda) - p_t|$
8: $x_t \leftarrow x(\lambda)$
9: **end for**

in the predictions, whereas a ratio of $\bar{\beta}/\underline{\beta}$ close to one means less confidence in the predictions. Note that both line 4 and 5 represent a single step in Online Balanced Descent [14] and can be efficiently solved by a binary search.

The benefit of adopting Online Balanced Descent as the robust algorithm in this context is that there is a close relationship between the switching cost and the hitting cost. For example, if the algorithm were to move to $x(\underline{\beta})$, then the switching cost at this step is exactly equal to $\underline{\beta}$ times the hitting cost. In fact, by convexity, for any $x = x(\lambda)$ for $\lambda \in [\underline{\lambda}, \bar{\lambda}]$ the switching cost is β times the hitting cost for a $\beta \in [\underline{\beta}, \bar{\beta}]$. Intuitively, since this is the basis of Online Balanced Descent, this guarantees robustness. Line 7 then chooses λ so as to minimize the distance to the advice, ensuring consistency.

The following theorem characterizes the competitive ratio of the AOBD algorithm.

Theorem 3.5. *Assume that $2\underline{\beta}\bar{\beta} + \underline{\beta} \geq 1$. Let $\mathrm{CR}(\eta)$ be the competitive ratio of AOBD (Algorithm 2). Then,*

$$\mathrm{CR}(\eta) \leq \min\left\{\left(1 + (2 + \bar{\beta}^{-1})\underline{\beta}\right)(1 + 2\eta), 1 + (2 + \underline{\beta}^{-1})\bar{\beta}\right\}. \tag{12}$$

In particular, if $\bar{\beta} = 1/\delta$ and $\underline{\beta} = \delta/(2+\delta)$ for $\delta \leq 2$, then,

$$\mathrm{CR}(\eta) \leq \min\left\{(1+\delta)(1+2\eta), 1 + \frac{3}{\delta} + \frac{2}{\delta^2}\right\}. \tag{13}$$

Note that if the predictions are accurate, i.e., if $\eta = 0$, then the competitive ratio of AOBD is at most $1 + \delta$. This means that AOBD is $(1+\delta)$-consistent. Moreover, even when $\eta = \infty$, the competitive ratio is uniformly bounded. Hence, AOBD is robust. Also, the algorithm smoothly interpolates to any scenario in between and depends linearly on η. As before, the confidence hyperparameter δ characterizes the trade-off between consistency and robustness.

The trade-off in Theorem 3.5 is in fact necessary, even in this special case, as the next theorem shows.

Theorem 3.6. *Fix any $0 < \delta < 1/2$. Let $\mathcal{A}$ be any deterministic algorithm for the convex, one-dimensional optimization problem. If $\mathcal{A}$ is $(1+\delta)$-consistent, then $\mathcal{A}$ is at least $1/(2\delta)$-robust.*

Provocatively, Theorem 3.6 is inconclusive about whether there exists an algorithm that has a slightly better trade-off between consistency and robustness in terms of δ than AOBD due to the $2/\delta^2$ term in Theorem 3.5. Still, Theorem 3.6 implies that, even in this special case, no algorithm can be 1-consistent and robust.

The proofs of Theorem 3.5 and 3.6 are provided in Appendices F and G, respectively. The proof of Theorem 3.5 follows a potential function argument, which is now feasible due to the convexity assumption. We use two potential functions: one that couples the AOBD algorithm with the optimal trajectory and provides the robustness bound, and the other that couples the AOBD algorithm with the advice and provides the consistency bound.

4 Conclusion

This paper considers online non-convex optimization with untrusted predictions. We have introduced Adaptive Online Switching (AOS) and proved that AOS nearly matches the performance of the hindsight optimal if predictions are accurate, while still ensuring a worst-case performance bound even if predictions are completely inaccurate. We validate the performance guarantees of AOS with experiments reported in Appendix A, and complement our results with a lower bound on the trade-off between consistency and robustness of *any* algorithm, showing that AOS has an order-optimal trade-off. Moreover, we have proven the necessity of memory in AOS and introduced a specialized algorithm, Adaptive Online Balanced Descent, for the special case of one-dimensional, convex optimization.

An open question, which we leave for future work, is how the trade-off between consistency and robustness translates to online *convex* optimization in general dimensions. Note that any algorithm for online convex optimization must again use memory due to Theorem 3.4, but the lower bound in Theorem 3.2 might be improved upon. Finally, in an ongoing work, we are examining how to incorporate untrusted predictions into algorithms for online convex body chasing.

References

[1] Albers, S. and Quedenfeld, J. (2018). Optimal Algorithms for Right-Sizing Data Centers. In *Proceedings of the 30th on Symposium on Parallelism in Algorithms and Architectures*, SPAA '18, pages 363–372, New York, NY, USA. Association for Computing Machinery.

[2] Albers, S. and Quedenfeld, J. (2021). Algorithms for Right-Sizing Heterogeneous Data Centers. In *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '21, pages 48–58, New York, NY, USA. Association for Computing Machinery.

[3] Altschuler, J. and Talwar, K. (2018). Online learning over a finite action set with limited switching. In *Proceedings of the 31st Conference On Learning Theory*, pages 1569–1573. PMLR.

[4] Andrew, L., Barman, S., Ligett, K., Lin, M., Meyerson, A., Roytman, A., and Wierman, A. (2013). A tale of two metrics: Simultaneous bounds on competitiveness and regret. In *Conference on Learning Theory*, pages 741–763. PMLR.

[5] Antoniadis, A., Coester, C., Elias, M., Polak, A., and Simon, B. (2020). Online metric algorithms with untrusted predictions. In *International Conference on Machine Learning*, pages 345–355. PMLR.

[6] Antoniadis, A. and Schewior, K. (2017). A tight lower bound for online convex optimization with switching costs. In *International Workshop on Approximation and Online Algorithms*, pages 164–175. Springer.

[7] Argue, C. J., Gupta, A., Tang, Z., and Guruganesh, G. (2021). Chasing convex bodies with linear competitive ratio. *Journal of the ACM*, 68(5):1–10.

[8] Bansal, N., Blum, A., Chawla, S., and Meyerson, A. (2003). Online oblivious routing. In *Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pages 44–49.

[9] Bansal, N., Gupta, A., Krishnaswamy, R., Pruhs, K., Schewior, K., and Stein, C. (2015). A 2-competitive algorithm for online convex optimization with switching costs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[10] Blum, A. and Burch, C. (2000). On-line learning and the metrical task system problem. *Machine Learning*, 39(1):35–58.

[11] Borodin, A., Linial, N., and Saks, M. E. (1992). An optimal on-line algorithm for metrical task system. *Journal of the ACM*, 39(4):745–763.

[12] Chen, N., Agarwal, A., Wierman, A., Barman, S., and Andrew, L. L. (2015). Online convex optimization using predictions. In *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 191–204.

[13] Chen, N., Comden, J., Liu, Z., Gandhi, A., and Wierman, A. (2016). Using predictions in online optimization: Looking forward with an eye on the past. *ACM SIGMETRICS Performance Evaluation Review*, 44(1):193–206.

[14] Chen, N., Goel, G., and Wierman, A. (2018). Smoothed online convex optimization in high dimensions via online balanced descent. In *Conference On Learning Theory*, pages 1574–1594. PMLR.

[15] Comden, J., Yao, S., Chen, N., Xing, H., and Liu, Z. (2019). Online optimization in cloud resource provisioning: Predictions, regrets, and algorithms. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 3(1):1–30.

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

[17] Evans, R. and Gao, J. (2016). DeepMind AI Reduces Google Data Centre Cooling Bill by 40%.

[18] Fiat, A., Rabani, Y., and Ravid, Y. (1994). Competitive k-server algorithms. *Journal of Computer and System Sciences*, 48(3):410–428.

[19] Goel, G., Chen, N., and Wierman, A. (2017). Thinking fast and slow: Optimization decomposition across timescales. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1291–1298. IEEE.

[20] Goel, G., Lin, Y., Sun, H., and Wierman, A. (2019). Beyond online balanced descent: An optimal algorithm for smoothed online optimization. *Advances in Neural Information Processing Systems*, 32:1875–1885.

[21] Goel, G. and Wierman, A. (2019). An online algorithm for smoothed regression and lqr control. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2504–2513. PMLR.

[22] Hsu, C.-Y., Indyk, P., Katabi, D., and Vakilian, A. (2019). Learning-based frequency estimation algorithms. In *International Conference on Learning Representations*.

[23] Joseph, V. and de Veciana, G. (2012). Jointly optimizing multi-user rate adaptation for video transport over wireless systems: Mean-fairness-variability tradeoffs. In *2012 Proceedings IEEE INFOCOM*, pages 567–575. IEEE.

[24] Kim, S.-J. and Giannakis, G. B. (2014). Real-time electricity pricing for demand response using online convex optimization. In *ISGT 2014*, pages 1–5. IEEE.

[25] King, J., Clifton, A., and Hodge, B.-M. (2014). Validation of power output for the wind toolkit. Technical Report NREL/TP-5D00-61714, National Renewable Energy Lab.(NREL), Golden, CO (United States).

[26] Koutsoupias, E. and Papadimitriou, C. H. (1995). On the k-server conjecture. *Journal of the ACM (JACM)*, 42(5):971–983.

[27] Kumar, N., Besuner, P., Lefton, S., Agan, D., and Hilleman, D. (2012). Power plant cycling costs. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States).

[28] Kumar, R., Purohit, M., Schild, A., Svitkina, Z., and Vee, E. (2018). Semi-online bipartite matching. *arXiv preprint arXiv:1812.00134*.

[29] Lee, R., Maghakian, J., Hajiesmaili, M., Li, J., Sitaraman, R., and Liu, Z. (2021). Online peak-aware energy scheduling with untrusted advice.

[30] Li, Y. and Li, N. (2020). Leveraging predictions in smoothed online convex optimization via gradient-based algorithms. *arXiv preprint arXiv:2011.12539*.

[31] Li, Y., Qu, G., and Li, N. (2018). Using predictions in online optimization with switching costs: A fast algorithm and a fundamental limit. In *2018 Annual American Control Conference (ACC)*, pages 3008–3013. IEEE.

[32] Lin, M., Liu, Z., Wierman, A., and Andrew, L. L. (2012a). Online algorithms for geographical load balancing. In *2012 International Green Computing Conference (IGCC)*, pages 1–10. IEEE.

[33] Lin, M., Wierman, A., Andrew, L. L., and Thereska, E. (2012b). Dynamic right-sizing for power-proportional data centers. *IEEE/ACM Transactions on Networking*, 21(5):1378–1391.

[34] Lin, Y. (2019). Personal correspondence.

[35] Lin, Y., Goel, G., and Wierman, A. (2020). Online optimization with predictions and non-convex losses. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 4(1):1–32.

[36] Lu, T., Chen, M., and Andrew, L. L. (2012). Simple and effective dynamic provisioning for power-proportional data centers. *IEEE Transactions on Parallel and Distributed Systems*, 24(6):1161–1171.

[37] Lykouris, T. and Vassilvtiskii, S. (2018). Competitive caching with machine learned advice. In *International Conference on Machine Learning*, pages 3296–3305. PMLR.

[38] Maghakian, J., Lee, R., Hajiesmaili, M., Li, J., Liu, Z., and Sitaraman, R. (2021). Leveraging different types of predictors for online optimization. In *2021 55th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–1. IEEE.

[39] Mahdian, M., Nazerzadeh, H., and Saberi, A. (2012). Online optimization with uncertain information. *ACM Transactions on Algorithms (TALG)*, 8(1):1–29.

[40] Mitzenmacher, M. (2018). A model for learned bloom filters and related structures. *arXiv preprint arXiv:1802.00884*.

[41] Pang, C. and Chen, H. (1976). Optimal short-term thermal unit commitment. *IEEE Transactions on Power Apparatus and Systems*, 95(4):1336–1346.

[42] Rutten, D. and Mukherjee, D. (2021). A new approach to capacity scaling augmented with unreliable machine learning predictions. *arXiv preprint arXiv:2101.12160*.

[43] Sellke, M. (2020). Chasing convex bodies optimally. In *Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '20, pages 1509–1518, USA. Society for Industrial and Applied Mathematics.

[44] Shi, M., Lin, X., and Fahmy, S. (2021). Competitive Online Convex Optimization With Switching Costs and Ramp Constraints. *IEEE/ACM Transactions on Networking*, 29(2):876–889.

[45] Singh, M., Singh, M., and Kaur, S. (2019). Ti-2016 dns dataset.

[46] Sun, B., Lee, R., Hajiesmaili, M., Wierman, A., and Tsang, D. H. (2021). Pareto-optimal learning-augmented algorithms for online conversion problems. *arXiv preprint arXiv:2109.01556*.

[47] Wang, H., Huang, J., Lin, X., and Mohsenian-Rad, H. (2014). Exploring smart grid and data center interactions for electric power load balancing. *ACM SIGMETRICS Performance Evaluation Review*, 41(3):89–94.

[48] Zanini, F., Atienza, D., Benini, L., and De Micheli, G. (2009). Multicore thermal management with model predictive control. In *2009 European Conference on Circuit Theory and Design*, pages 711–714. IEEE.

[49] Zanini, F., Atienza, D., De Micheli, G., and Boyd, S. P. (2010). Online convex optimization-based algorithm for thermal management of mpsocs. In *Proceedings of the 20th symposium on Great lakes symposium on VLSI*, pages 203–208.

[50] Zhang, L., Jiang, W., Lu, S., and Yang, T. (2021). Revisiting smoothed online learning. *arXiv preprint arXiv:2102.06933*.

A Experiments

In this section, we empirically evaluate the performance of AOS (Algorithm 1) in robustifying the decisions made by a machine-learned algorithm for microgrid operation. Specifically, we consider a model of a DNS server receiving power from an islanded microgrid with an 8 MW wind turbine and six 2 MW dispatchable generators. At each time $t = 1, \ldots, T$, the server incurs DNS traffic $d_t \in [0, 12]$ (assumed to be reported in MW of equivalent required power) and the wind turbine produces power $w_t \in [0, 8]$ (in MW). In response, the microgrid operator must choose some generator dispatch $u_t \in \{0, 1\}^6$, where $u_{i,t}$ (the i^{th} entry of u_t) indicates the commitment status of generator i at time t: if $u_{i,t} = 1$, then generator i is producing 2 MW at time t, and if $u_{i,t} = 0$, then generator i is off at time t.[4] The goal of the microgrid operator is to minimize costs while approximately balancing dispatchable generation with net load $\ell_t := d_t - w_t$, i.e., achieving $\mathbf{2}^\top u_t \approx \ell_t$ at each time t. More specifically, at each time t, the operator faces the following hitting cost:

$$f_t(u_t) = \mathbf{c}^\top u_t + \gamma \max\{\ell_t - \mathbf{2}^\top u_t, 0\}$$

where $\mathbf{c} \in \mathbb{R}^6_+$ is a vector of fuel costs for each generator, and γ is the cost associated with failing to meet 1 MW of demand. Thus f_t has two parts: the first term gives the fuel cost associated with a commitment decision u_t, and the second term penalizes mismatch between generation and demand. We assume that $\mathbf{c}$ has no two entries identical, i.e., there is a strict ordering of generator costs, and moreover that γ is strictly greater than each entry of $\mathbf{c}$.

In addition to the hitting cost at each time, there is also a switching cost associated with changing generators' commitment statuses between timesteps, given by

$$d(u_t, u_{t-1}) = \beta \|u_t - u_{t-1}\|_{\ell^1}.$$

Such a switching cost is common in the power systems literature on unit commitment, where it is referred to as a cycling or startup/shutdown cost, and reflects fixed costs associated with cycling a generator on/off, including fuel costs for "cold starts" and wear and tear due to thermal stress, e.g., [27, 41]. As a result, the total cost faced by the microgrid operator is $\sum_{t=1}^T f_t(u_t) + d(u_t, u_{t-1})$; so the microgrid operator must balance fuel costs and meeting load while taking care not to cycle the generators more than is necessary. Under the assumption that net demand ℓ_t is drawn randomly from some probability density, then there is some $\alpha > 0$, unknown to the microgrid operator *a priori*, such that f_t is almost surely α-polyhedral with respect to the distance d at each time t. We can thus frame the problem of optimal microgrid operation in our framework of online optimization with switching costs and non-convex hitting costs, where the non-convexity arises from the discrete nature of the dispatch decisions u_t.

In our experiments, we set $\mathbf{c} = (1, 1.2, 1.4, 1.6, 1.8, 2)$, $\gamma = 3$, and $\beta = 8$. We use four days of per-second DNS traffic data from a campus network from [45], which we aggregate into 15 minute periods and suitably normalize. We gather simulated data on the wind speed at an altitude of 100 m off the coast of California at (39.970406, -128.77481) every 15 minutes of the year 2019 from the Wind Integration National Dataset Toolkit [25]. We then convert the wind speed data to generation levels using the power curve for an IEC Class 2 wind turbine [25]. As we have significantly more wind generation data than DNS traffic data, we compute 92 four-day net load trajectories, each

[4] In practice, many dispatchable generation resources and in particular small backup generators are constrained in their output flexibility – i.e., if $u_{i,t} = 1$, then generator i is producing at or near its maximal capacity. As such, it suffices in this example to assume that if $u_{i,t} = 1$, then generator i is producing at exactly 2 MW, though our model formulation can be extended to the case with both discrete commitment variables u_t as well as an auxiliary variable corresponding to generation level.

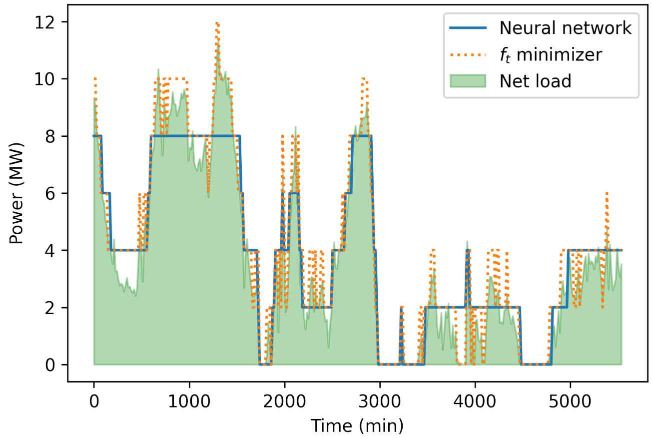

Figure 1: Comparison of the neural network dispatch vs. f_t minimizer dispatch on an example net load trajectory in the test set.

using the same window of DNS traffic data but different windows of wind generation. We separate these into a training set of 70 trajectories and a test set of 22 trajectories.

We train a three-layer neural network for the microgrid operation problem as outlined. We assume that at time t, in addition to the previous dispatch u_{t-1} and the current net load ℓ_t, the microgrid operator also has predictions of the next $W = 10$ net loads $\hat{\ell}_{t+1} \dots, \hat{\ell}_{t+W}$, which are used in conjunction with u_{t-1} and ℓ_t as inputs to the neural network. We train the model under the assumption that predictions are perfect, i.e., $\hat{\ell}_{t+k} = \ell_{t+k}$ for each $k = 1, \dots, W$. However, at test time, we evaluate the impact of perturbing the predictions in order to evaluate the success of AOS (Algorithm 1) at ensuring worst-case robustness even under the presence of distribution shift or increased prediction noise. Note that even though the problem setting is non-convex due to the integer variables u_t, Algorithm 1 is in practice tractable, as the minimizations defining v_t and p_t at each time can be computed using standard tools for mixed-integer linear programming.

To illustrate the behavior of machine-learned algorithms that enable improved performance when using perfect predictions of future net loads, we plot in Figure 1 the aggregate dispatch power planned by the trained neural network alongside the dispatches chosen by the algorithm of [50] that simply chooses the minimizer v_t of f_t at each time t. Since generator commitments are discrete, the microgrid generation can only occur in factors of 2 MW. From the figure, it is evident that the algorithm that follows the minimizers v_t undergoes significant variation in dispatches and hence incurs large costs due to frequent cycling, whereas the machine-learned algorithm, with access to perfect predictions of future demands over the next 10 time periods, has learned to "smooth" its dispatches so as not to incur cycling costs unless necessary.

Next, we evaluate the performance of AOS (Algorithm 1) using the neural network's decision at time t as the prediction $\tilde{x}_t$. We consider the behavior of AOS for various values of the hyperparameter δ, and under two different prediction scenarios: in the first, we assume that predictions are perturbed by zero-mean i.i.d. Gaussian noise with standard deviation σ, representing the case where predictions at test time are noisier than those during training; and in the second, we assume that predictions are perturbed deterministically with a bias of value μ that is uniform across predictions at the current timestep, representing the scenario in which the model giving future net load predictions is biased due to distribution shift between training and test time. In each scenario, we vary the values

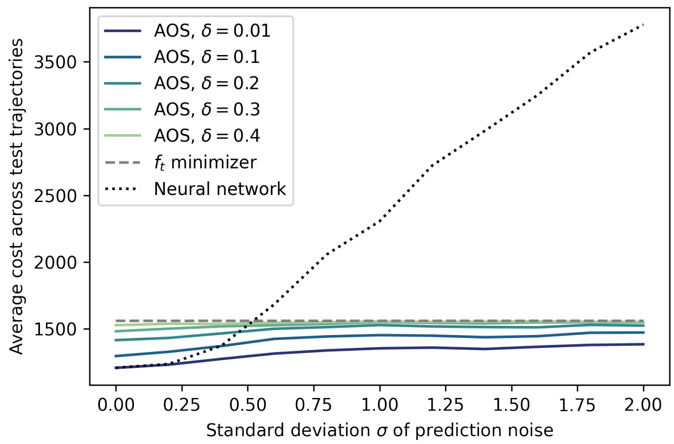

Figure 2: Performance of the AOS algorithm vs. the f_t minimizer and neural network algorithms under different scenarios of prediction noise magnitude and selections of hyperparameter δ.

of σ and μ to evaluate the resulting impact on the neural network performance, and resultingly, on the performance of AOS.

We detail the results in the zero-mean Gaussian noise case in Figure 2. It is clear that the neural network algorithm performs well in the regime of small σ, and improves upon the performance of the f_t minimizer algorithm for all $\sigma \leq 0.5$. Moreover, when δ and σ are both small, the performance of AOS nearly matches that of the neural network, and degrades as δ becomes larger. However, remarkably, the performance of AOS is vastly improved over that of the neural network in the regime of high prediction noise, and even matches or improves upon that of the f_t minimizer algorithm in this regime, even when $\delta = 0.01$. This good performance is likely due both to the adaptive switching of the AOS algorithm as well as the filtering of the neural network decisions $\tilde{x}_t$ to arrive at improved decisions p_t. Together, these two properties allow AOS to exploit each algorithm when it is performing well, enabling AOS to achieve the "best of both worlds" in terms of performance. Thus this example demonstrates that AOS with a small δ can bridge the performance of the neural network algorithm and the f_t minimizer algorithm across noise regimes.

We illustrate the performance of the AOS algorithm with various choices of δ against the neural network and f_t minimizer in the deterministic prediction perturbation scenario in Figure 3. Here, the impact on neural network and AOS performance of perturbation μ being far from zero is more pronounced: for $|\mu| \approx 1$, the neural network algorithm incurs high cost, and as a result AOS incurs cost that is significantly larger than that of the f_t minimizer algorithm when $\delta = 0.01$. However, as δ increases, the performance of AOS dramatically improves, rapidly approaching performance comparable to the f_t minimizer. Moreover, in the $|\mu| \ll 1$ regime, we see once again that the neural network outperforms the f_t minimizer, and AOS can closely match the neural network performance when δ is small. In particular, the choosing $\delta = 0.1$ seems to effectively bridge the good performance of the neural network algorithm in the $|\mu| \ll 1$ setting with the robustness of the f_t minimizer in the large $|\mu|$ setting.

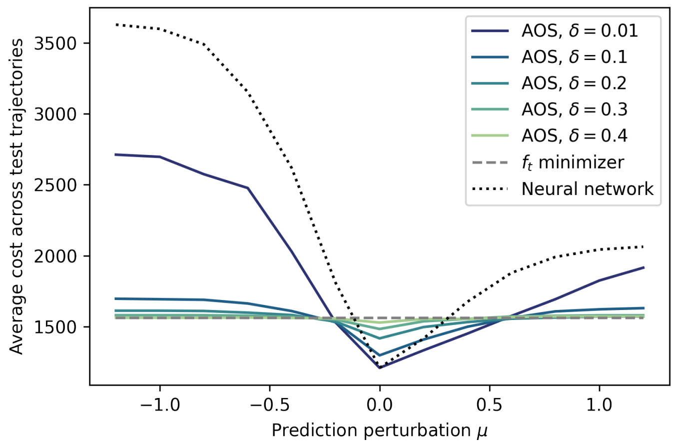

Figure 3: Performance of the AOS algorithm vs. the f_t minimizer and neural network algorithms under different scenarios of prediction perturbation and selections of hyperparameter δ.

B Online Convex Optimization with Bregman Divergence Switching Costs

To highlight the challenges for trade-offs between consistency and robustness in related online optimization settings, we prove a lower bound on the trade-off between consistency and robustness for the case of squared ℓ_2 switching costs. In particular, we consider the following assumptions.

1. The functions f_t are α-strongly convex and β-strongly smooth.
2. The space $M = \mathbb{R}$ and the distance $d(x_{t-1}, x_t) = \frac{1}{2}\|x_{t-1} - x_t\|_2^2$.

Note that the squared ℓ_2 switching costs are an instance of the more general Bregman divergence switching costs [20]. We now state our lower bound.

Lemma B.1. *Let $\mathcal{A}$ be any algorithm for the online convex optimization problem. The next two statements are mutually exclusive.*

1. *$\mathcal{A}$ is γ-robust where $\gamma < \infty$.*
2. *$\mathcal{A}$ is c-consistent where*

$$c < \frac{1 + \sqrt{1 + 4\alpha^{-1}}}{2(1 + \beta^{-1})}. \tag{14}$$

Notice that the consistency bound in (14) is achieved by an online algorithm which ignores predictions in [20] in the case that $\beta = \infty$. Thus, this bound shows the difficulty in extracting value from untrusted predictions in this setting.

Proof. Fix any $\delta > 0$. Let $x_0 = 0$ and $f_t(x) = \alpha x^2/2$ for $1 \le t \le T-1$. The predictions $\tilde{x}_t$ are equal to the hindsight optimal as if $f_T(x) = \beta(x-\delta)^2/2$. We distinguish two cases.

1. Assume there exists $1 \le t \le T-1$ such that $x_t > 0$. Then, let $f_T(x) = \beta x^2/2$. The optimal algorithm achieves a cost of zero by letting $o_t = 0$ for all $1 \le t \le T$. However, the cost of $\mathcal{A}$ is at least $f_t(x_t) = \alpha x_t^2/2 > 0$ and hence the competitive ratio is unbounded. Therefore, $\mathcal{A}$ is not robust.

2. Assume that $x_t = 0$ for $1 \leq t \leq T$. Then, let $\theta_T = (\delta, \beta)$. Goel et al. [20] prove that the cost of the hindsight optimal is at most $\delta^2(-\alpha + \sqrt{\alpha^2 + 4\alpha})/4$ asymptotically as $T \to \infty$. The algorithm $\mathcal{A}$ suffers cost at least

$$\min_{x \in [0,\delta]} \frac{x^2}{2} + \frac{\beta(x-\delta)^2}{2} = \frac{\delta^2}{2(1+\beta^{-1})}, \tag{15}$$

and hence the competitive ratio is at least

$$\frac{2}{(1+\beta^{-1})(-\alpha + \sqrt{\alpha^2 + 4\alpha})} = \frac{1 + \sqrt{1 + 4\alpha^{-1}}}{2(1+\beta^{-1})}. \tag{16}$$

Therefore, since the predictions are perfect, $\mathcal{A}$ is not c-consistent for any c strictly smaller than (16).

□

C Proof of Theorem 3.1

By definition, following the predictions $\tilde{x}_1, \ldots, \tilde{x}_T$ exactly guarantees 1-consistency. However, if the hitting cost functions are steep and predictions are not perfect, naively following the predictions does not yield a competitive ratio with a smooth dependence on the accuracy η. However, it is possible to obtain, by suitably "filtering" the predictions, a sequence of decisions that are at least as good as the predictions and achieve a competitive ratio that is linear in η. We use this subroutine that filters the predictions in both AOS and AOBD.

To this end, we separately present in Algorithm 3 the filtering procedure called *Follow the Prediction* (FtP) that, at each time t, moves to a point p_t constituting a "filtered" form of the prediction $\tilde{x}_t$: p_t is chosen to minimize the sum of hitting cost, switching cost, and the cost to subsequently switch to $\tilde{x}_t$.

Algorithm 3 Follow the Prediction

```
p_0 ← x_0
for t = 1, ..., T do
    Observe f_t and x̃_t
    p_t ← argmin_{p∈M} f_t(p) + d(p, p_{t−1}) + d(p, x̃_t)
end for
```

The FtP algorithm is a refinement of the algorithm by the same name in [5]. As we shall see in the following lemma, FtP is no worse than the original predictions, and its competitive ratio is linear in η.

Lemma C.1. *Let* $\mathrm{CR}(\eta)$ *be the competitive ratio of* FtP *(Algorithm 3). Then,*

$$\mathrm{CR}(\eta) \leq 1 + 2\eta. \tag{17}$$

Moreover, for any fixed $\tilde{x}$*,* FtP *incurs a cost at most the cost of* $\tilde{x}$*.*

Note that Lemma C.1 does not guarantee that FtP is worst-case competitive, a.k.a. robust, and in fact, it is relatively straightforward to construct example settings where η is arbitrarily large and FtP has an unbounded competitive ratio.

Proof of Lemma C.1. We begin with the second claim. Since p_t minimizes line 4 of Algorithm 3 for each t, we have

$$\sum_{t=1}^{T} f_t(p_t) + d(p_t, p_{t-1}) + d(p_t, \tilde{x}_t) \leq \sum_{t=1}^{T} f_t(\tilde{x}_t) + d(\tilde{x}_t, p_{t-1}).$$

The bound follows immediately from moving $\sum_{t=1}^{T-1} d(p_t, \tilde{x}_t)$ to the right-hand side and applying the triangle inequality.

The proof of the competitive ratio bound is a slight modification of the argument in [5]. Define

$$C_t = \sum_{i=1}^{t} f_t(p_t) + d(p_t, p_{t-1}) + f_{t+1}(o_{t+1}) + d(o_{t+1}, p_t) + \sum_{i=t+2}^{T} f_t(o_t) + d(o_t, o_{t-1}).$$

Observe

$$\begin{aligned} C_t &= C_{t-1} + (d(p_t, p_{t-1}) + d(o_{t+1}, p_t) + f_t(p_t)) - (d(o_t, p_{t-1}) + d(o_{t+1}, o_t) + f_t(o_t)) \\ &\leq C_{t-1} + d(p_t, p_{t-1}) - d(o_t, p_{t-1}) + d(p_t, o_t) + f_t(p_t) - f_t(o_t) \end{aligned}$$

by the triangle inequality. Then using $d(p_t, o_t) \leq d(p_t, \tilde{x}_t) + d(o_t, \tilde{x}_t)$ as well as the fact that p_t minimizes the objective of line 4, we obtain

$$\begin{aligned} C_t &\leq C_{t-1} - d(o_t, p_{t-1}) + d(o_t, \tilde{x}_t) - f_t(o_t) + f_t(o_t) + d(o_t, p_{t-1}) + d(o_t, \tilde{x}_t) \\ &= C_{t-1} + 2d(o_t, \tilde{x}_t). \end{aligned}$$

Summing the inequality over t and observing that C_0 is the cost of the optimal trajectory whereas C_T is the cost of FtP yields the competitive ratio bound. □

Recall Theorem 3.1 as stated in the previous section. The theorem summarizes two separate bounds on the competitive ratio of the AOS algorithm (Algorithm 1), which is represented by the two terms in the minimum. We provide a proof of Theorem 3.1 here, where we treat the two bounds separately and prove the following two propositions, which are slightly stronger than Theorem 3.1. Let $\textsc{Alg}(t) := f_t(x_t) + d(x_t, x_{t-1})$ denote the cost incurred by the AOS algorithm at time t.

Proposition C.2. *(Consistency)*

$$\sum_{i=1}^{T} \textsc{Alg}(i) \leq (1 + 2\delta) \sum_{i=1}^{T} \textsc{Adv}(i) \tag{18}$$

Proposition C.3. *(Robustness)*

$$\sum_{i=1}^{T} \textsc{Alg}(i) \leq \left(\frac{4U(\infty) + 4}{\delta} + 2U(\infty) + 5 \right) \sum_{i=1}^{T} \textsc{Rob}(i), \tag{19}$$

where $U(\infty) := \sup_{t \geq 1} U(t)$ *and*

$$\begin{aligned} U(t) := \min_{U,y} \quad & U \\ \text{s.t.} \quad & \delta \sum_{i=s}^{t} (1 + \alpha(i - s + 1)) y_i + \alpha \sum_{i=s}^{t-1} y_i - 2 \sum_{i=s+1}^{t} y_i \geq 1 + \alpha(t - s + 1) \quad \textit{for } 1 \leq s \leq t, \\ & U \geq 2y_s + \delta \sum_{i=s}^{t} y_i - 1, \; y_s \geq 0 \quad \textit{for } 1 \leq s \leq t. \end{aligned} \tag{20}$$

Moreover, assume that $\frac{2}{\alpha\delta} \in \mathbb{N}$*. Then,*

$$\sum_{i=1}^{T} \textsc{Alg}(i) \leq \left(\frac{4\tilde{U}+4}{\delta} + 2\tilde{U} + 5\right) \sum_{i=1}^{T} \textsc{Rob}(i), \tag{21}$$

where

$$\tilde{U} = \alpha \left(\frac{2}{\alpha+\delta(1+\alpha)}\right)^{2/\alpha\delta} + \frac{2}{(2-\alpha-\delta(1+\alpha))^2} \left(\alpha \left(\frac{2}{\alpha+\delta(1+\alpha)}\right)^{2/\alpha\delta} - \frac{2-\alpha}{\delta} + 1\right). \tag{22}$$

We note that the assumption that $2/(\alpha\delta) \in \mathbb{N}$ in the statement of Proposition C.3 is without loss of generality, but it prevents rounding symbols from appearing in the notation.

Proof of Theorem 3.1. Note that Proposition C.2 and Lemma C.1 imply that

$$\sum_{i=1}^{T} \textsc{Alg}(i) \leq (1+2\delta) \sum_{i=1}^{T} \textsc{Adv}(i) \leq (1+2\delta)(1+2\eta) \sum_{i=1}^{T} \textsc{Opt}(i), \tag{23}$$

which proves that the competitive ratio of the AOS algorithm is at most $(1+2\delta)(1+2\eta)$. Also, note that Proposition C.3 and the competitive ratio of $\max\{1, 2/\alpha\}$ of the greedy algorithm [50] imply that

$$\sum_{i=1}^{T} \textsc{Alg}(i) \leq \left(\frac{4\tilde{U}+4}{\delta} + 2\tilde{U} + 5\right) \sum_{i=1}^{T} \textsc{Rob}(i) \leq \left(\frac{4\tilde{U}+4}{\delta} + 2\tilde{U} + 5\right) \max\left\{1, \frac{2}{\alpha}\right\} \sum_{i=1}^{T} \textsc{Opt}(i), \tag{24}$$

which completes the proof of the theorem. □

We detail the proofs of Proposition C.2 and C.3 in the next two sections, respectively.

C.1 Proof of Consistency Bound

Recall that the AOS algorithm works in cycles or *stages*. A stage $k \in \mathbb{N}$ starts at time T_k when the algorithm switches to the advice and ends at the time $T_{k+1} - 1$ when the algorithm again switches back to the advice after having switched to the greedy algorithm in between. Also, recall that the time when the algorithm switches to the greedy algorithm in the k-th stage is denoted by M_k. In the rest of the proofs, whenever we refer to T_k (and M_k), we will implicitly assume that k is such that $T_k \leq T$ (and $M_k \leq T$, resp.) Intuitively, the lemma below proves that the AOS algorithm is $(1+2\delta)$-consistent within each stage.

Lemma C.4. *For all* $k \in \mathbb{N}$ *and* $T_k \leq t \leq T_{k+1} - 1$*,*

$$\sum_{i=T_k}^{t} \textsc{Alg}(i) - d(v_{T_k-1}, p_{T_k-1}) \leq (1+2\delta) \sum_{i=T_k}^{t} \textsc{Adv}(i) - d(x_t, p_t). \tag{25}$$

Lemma C.4 readily implies the consistency bound stated in Proposition C.2.

Proof of Proposition C.2. Let $K \in \mathbb{N}$ be the largest integer such that $T \geq T_K$. Then,

$$\begin{aligned}\sum_{i=1}^{T} \textsc{Alg}(i) &\leq \sum_{k=1}^{K-1} \left(\sum_{i=T_k}^{T_{k+1}-1} \textsc{Alg}(i) + d(v_{T_{k+1}-1}, p_{T_{k+1}-1}) - d(v_{T_k-1}, p_{T_k-1}) \right) \\ &\quad + \sum_{i=T_K}^{T} \textsc{Alg}(i) + d(x_T, p_T) - d(v_{T_K-1}, p_{T_K-1}) \leq (1+2\delta) \sum_{i=1}^{T} \textsc{Adv}(i),\end{aligned} \tag{26}$$

where the first inequality follows from the telescoping sum and the second inequality follows using Lemma C.4. □

We now provide the proof of Lemma C.4. The proof is rather straightforward and follows along similar lines as the intuition introduced before. The proof depends on the conditions in lines 4 and 9 in the AOS algorithm.

Proof of Lemma C.4. Fix any $k \in \mathbb{N}$ and $T_k \leq t \leq M_k - 1$. Then,

$$\begin{aligned}\sum_{i=T_k}^{t} \textsc{Alg}(i) - d(v_{T_k-1}, p_{T_k-1}) &= f_{T_k}(p_{T_k}) + d(v_{T_k-1}, p_{T_k}) + \sum_{i=T_k+1}^{t} \textsc{Adv}(i) - d(v_{T_k-1}, p_{T_k-1}) \\ &\leq \sum_{i=T_k}^{t} \textsc{Adv}(i) - d(p_t, p_t),\end{aligned} \tag{27}$$

where the inequality follows by the triangle inequality. Fix any $k \in \mathbb{N}$ and $M_k \leq t \leq T_{k+1} - 1$. Then,

$$\begin{aligned}&\sum_{i=T_k}^{t} \textsc{Alg}(i) - d(v_{T_k-1}, p_{T_k-1}) \\ &\quad = f_{T_k}(p_{T_k}) + d(v_{T_k-1}, p_{T_k}) + \sum_{i=T_k+1}^{M_k-1} \textsc{Adv}(i) + \sum_{i=M_k}^{t} \textsc{Alg}(i) - d(v_{T_k-1}, p_{T_k-1}) \\ &\quad \leq \sum_{i=T_k}^{M_k-1} \textsc{Adv}(i) + f_{M_k}(v_{M_k}) + d(p_{M_k-1}, v_{M_k}) + \sum_{i=M_k+1}^{t} \textsc{Rob}(i) \\ &\quad \leq \sum_{i=T_k}^{M_k-1} \textsc{Adv}(i) + d(p_{M_k-1}, v_{M_k-1}) + \sum_{i=M_k}^{t} \textsc{Rob}(i) \\ &\quad \leq (1+\delta) \sum_{i=T_k}^{M_k} \textsc{Adv}(i) + \sum_{i=M_k+1}^{t} \textsc{Rob}(i) - d(v_{M_k}, p_{M_k}) \leq (1+2\delta) \sum_{i=T_k}^{M_k} \textsc{Adv}(i) - d(v_t, p_t),\end{aligned} \tag{28}$$

where the first and second inequality follow by the triangle inequality, the third inequality follows by line 4 of Algorithm 1, and the fourth inequality follows by line 9 of Algorithm 1. □

C.2 Proof of Robustness Bound

The proof of the robustness bound is more involved. For example, even the fact that the AOS algorithm eventually switches to the greedy algorithm and does not follow the advice forever, is not obvious. The reason that the AOS algorithm eventually switches to the greedy algorithm and is robust itself critically depends on the globally α-polyhedral assumption. Intuitively, to see why the AOS algorithm will switch eventually, consider the following example. Assume that the minimizer is static and the advice moves a distance of 1 away from the minimizer in each round. Then, the accumulated switching cost is equal to t at time t. However, the accumulated hitting cost is at least $\alpha \sum_{i=1}^{t} i = \alpha t(t+1)/2$ at time t. Hence, the accumulated hitting cost increases *quadratically*, while the accumulated switching cost increases *linearly*. In general, the accumulated hitting cost increases *at a faster rate* than the accumulated switching cost. As the distance to the minimizer is upper bounded by the accumulated switching cost, this means that there must be a t for which

$$d(p_{t-1}, v_{t-1}) + d(v_t, p_t) = t - 1 + t < \frac{\alpha\delta(t-1)t}{2} \leq \sum_{i=1}^{t-1} \textsc{Adv}(i), \tag{29}$$

which is sufficient to invalidate the condition in line 4, and hence, the AOS algorithm will switch to the greedy algorithm. In general, not only the time until a switch is bounded, one can even bound the cost of the advice until this time is reached, as shown by the next lemma.

Lemma C.5. *For all $k \in \mathbb{N}$ and $1 \leq t \leq M_k - T_k - 1$,*

$$\sum_{i=T_k+1}^{T_k+t} \textsc{Adv}(i) \leq U(t) \left(\sum_{i=T_k+1}^{T_k+t} \textsc{Rob}(i) + d(p_{T_k}, v_{T_k}) \right), \tag{30}$$

where $U(t)$ is as defined in Proposition C.3.

Proof of Lemma C.5. The idea of the proof is to write the inequality as an optimization problem, where the adversary aims to invalidate the inequality by maximizing the left-hand side and minimizing the right-hand side simultaneously. We reduce the constraints in the optimization problem to a set of necessary, linear constraints, and therefore construct a linear program that represents the inequality. The objective value of the linear program is equal to the objective value of its dual, which results in the expression for $U(t)$ in (20).

Fix any $k \in \mathbb{N}$ and $1 \leq t \leq M_k - T_k - 1$. Let

$$\begin{aligned}
\Delta_s^{\parallel} &:= d(p_{T_k+s}, v_{T_k+s}) - d(p_{T_k+s-1}, v_{T_k+s-1}) \quad && \text{for } 2 \leq s \leq t, \\
\Delta_1^{\parallel} &:= d(p_{T_k+1}, v_{T_k+1}), \\
\Delta_s^{\perp} &:= d(p_{T_k+s}, p_{T_k+s-1}) - \left|\Delta_s^{\parallel}\right| && \text{for } 1 \leq s \leq t, \\
\sigma_s &:= \frac{f_{T_k+s}(p_{T_k+s})}{d(p_{T_k+s}, v_{T_k+s})} && \text{for } 1 \leq s \leq t, \\
\gamma_s &:= \textsc{Rob}(T_k+s) && \text{for } 2 \leq s \leq t, \\
\gamma_1 &:= \textsc{Rob}(T_k+1) + d(p_{T_k}, v_{T_k}).
\end{aligned} \tag{31}$$

Note that the problem may be rewritten only in terms of the definitions introduced above. In fact,

$$\begin{aligned}
&\sum_{i=T_k+1}^{T_k+t} \textsc{Adv}(i) - U(t) \left(\sum_{i=T_k+1}^{T_k+t} \textsc{Rob}(i) + d(p_{T_k}, v_{T_k}) \right) \leq 0 \\
\iff &\sum_{i=1}^{t} \left(\Delta_i^{\perp} + \left|\Delta_i^{\parallel}\right| + \sigma_i \sum_{j=1}^{i} \Delta_j^{\parallel} \right) - U(t) \sum_{i=1}^{t} \gamma_i \leq 0.
\end{aligned} \tag{32}$$

We identify a set of necessary constraints on the quantities introduced above. Note that

$$
\begin{aligned}
&\sum_{i=T_k}^{T_k+s} \textsc{Adv}(i) + 2\textsc{Rob}(T_k+s) + 2d(p_{T_k+s-1}, v_{T_k+s-1}) \\
&\qquad\qquad \geq \sum_{i=T_k}^{T_k+s-1} \textsc{Adv}(i) + \textsc{Rob}(T_k+s) + d(p_{T_k+s-1}, v_{T_k+s-1}) + \alpha d(p_{T_k+s}, v_{T_k+s}) \\
&\qquad\qquad\quad + d(p_{T_k+s}, p_{T_k+s-1}) + d(v_{T_k+s}, v_{T_k+s-1}) + d(p_{T_k+s-1}, v_{T_k+s-1}) \\
&\qquad\qquad \geq \sum_{i=T_k}^{T_k+s-1} \textsc{Adv}(i) + \textsc{Rob}(T_k+s) + d(p_{T_k+s-1}, v_{T_k+s-1}) + d(v_{T_k+s}, p_{T_k+s}) \\
&\qquad\qquad\quad + \alpha d(p_{T_k+s}, v_{T_k+s}) \\
&\qquad\qquad \geq (1+\delta) \sum_{i=T_k}^{T_k+s} \textsc{Adv}(i) + \alpha d(p_{T_k+s}, v_{T_k+s}),
\end{aligned}
\tag{33}
$$

for all $1 \leq s \leq t$, where the second inequality follows by the triangle inequality and the third inequality follows from line 4 of Algorithm 1. Therefore,

$$
2\gamma_s + 2\sum_{i=1}^{s-1} \Delta_i^{\|} \geq \delta \sum_{i=1}^{s} \left(\Delta_i^{\perp} + \left|\Delta_i^{\|}\right| + \sigma_i \sum_{j=1}^{i} \Delta_j^{\|} \right) + \alpha \sum_{i=1}^{s} \Delta_i^{\|} \text{ for } 1 \leq s \leq t. \tag{34}
$$

The next constraints follow directly from the definitions:

$$
\sum_{i=1}^{s} \Delta_i^{\|} \geq 0,\ \Delta_s^{\perp} \geq -\gamma_s,\ \sigma_s \geq \alpha,\ \gamma_s \geq 0 \text{ for } 1 \leq s \leq t, \tag{35}
$$

where the second constraint follows by the triangle inequality. Now,

$$
\begin{aligned}
&\sum_{i=T_k+1}^{T_k+t} \textsc{Adv}(i) - U(t)\left(\sum_{i=T_k+1}^{T_k+t} \textsc{Rob}(i) + d(p_{T_k}, v_{T_k}) \right) \\
&\qquad\qquad \leq \sup_{\Delta^{\|}, \Delta^{\perp}, \sigma, \gamma} \quad \sum_{i=1}^{t} \left(\Delta_i^{\perp} + \left|\Delta_i^{\|}\right| + \sigma_s \sum_{j=1}^{i} \Delta_j^{\|} \right) - U(t) \sum_{i=1}^{t} \gamma_i \\
&\qquad\qquad\qquad\qquad \text{s.t.} \quad (34) \text{ and } (35).
\end{aligned}
\tag{36}
$$

We claim that (36) is equivalent to a linear problem. More formally, we will prove the following:

Claim C.6. *Fix any $\lambda = (\Delta^{\|}, \Delta^{\perp}, \sigma, \gamma)$, which satisfies (34) and (35). Then, there exists $\hat{\lambda} = (\hat{\Delta}^{\|}, \hat{\Delta}^{\perp}, \alpha, \hat{\gamma})$, satisfying (34) and (35), such that $\hat{\Delta}_s^{\|} \geq 0$ for $1 \leq s \leq t-1$, $\hat{\Delta}_t^{\|} \leq 0$, and the objective values of λ and $\hat{\lambda}$ are equal.*

We provide the proof of Claim C.6 below. Note that Claim C.6 implies that

$$
\begin{aligned}
(36) \ = \max_{\Delta^{\|},\Delta^{\perp},\gamma} \quad & \sum_{i=1}^{t-1}\left(\Delta_i^{\perp}+\Delta_i^{\|}+\alpha\sum_{j=1}^{i}\Delta_j^{\|}\right)+\left(\Delta_t^{\perp}+\Delta_t^{\|}+\alpha\left(\sum_{i=1}^{t-1}\Delta_i^{\|}-\Delta_t^{\|}\right)\right)-U(t)\sum_{i=1}^{t}\gamma_i \\
\text{s.t.} \quad & 2\gamma_s+2\sum_{i=1}^{s-1}\Delta_i^{\|}\ge\delta\sum_{i=1}^{s}\left(\Delta_i^{\perp}+\Delta_i^{\|}+\alpha\sum_{j=1}^{i}\Delta_j^{\|}\right)+\alpha\sum_{i=1}^{s}\Delta_i^{\|} \text{ for } 1\le s\le t-1, \\
& 2\gamma_t+2\sum_{i=1}^{t-1}\Delta_i^{\|}\ge\delta\sum_{i=1}^{t-1}\left(\Delta_i^{\perp}+\Delta_i^{\|}+\alpha\sum_{j=1}^{i}\Delta_j^{\|}\right)+\delta\left(\Delta_t^{\perp}+\Delta_t^{\|}+\alpha\left(\sum_{i=1}^{t-1}\Delta_i^{\|}-\Delta_t^{\|}\right)\right), \\
& \Delta_s^{\|}\ge 0,\ \ \Delta_s^{\perp}\ge-\gamma_s,\ \gamma_s\ge 0 \text{ for } 1\le s\le t.
\end{aligned}
\tag{37}
$$

Observe that the linear program in (37) is conic. Hence, the objective value of (37) is either zero or unbounded. Then, by strong duality, the objective value is equal to zero if and only if there exists a solution y to the dual, i.e.,

$$
\begin{aligned}
\delta\sum_{i=s}^{t}(1+\alpha(i-s+1))y_i+\alpha\sum_{i=s}^{t-1}y_i-2\sum_{i=s+1}^{t}y_i\ge 1+\alpha(t-s+1) \quad & \text{for } 1\le s\le t, \\
U(t)\ge 2y_s+\delta\sum_{i=s}^{t}y_i-1,\ y_s\ge 0 \quad & \text{for } 1\le s\le t.
\end{aligned}
\tag{38}
$$

Therefore, by the definition of $U(t)$, the objective value in (37) must be zero, which completes the proof of the lemma. □

Proof of Claim C.6. Fix any $\lambda=(\Delta^{\|},\Delta^{\perp},\sigma,\gamma)$, which satisfies (34) and (35). Then, let $\lambda'=(\Delta^{\|},\Delta^{\perp\prime},\alpha,\gamma)$, where

$$
\Delta_s^{\perp\prime}:=\Delta_s^{\perp}+(\sigma_s-\alpha)\sum_{i=1}^{s}\Delta_i^{\|} \text{ for } 1\le s\le t. \tag{39}
$$

Note that λ' satisfies (34) and (35) and the objective values of λ and λ' are equal. Let $1\le l\le t-1$ be the smallest integer such that $\Delta_l^{\|}<0$. We will reason inductively on the value of l. If such an l does not exist, then let $l=t-1$ by convention and set $\lambda''=\lambda$. If $l\le t-1$, then let

$\lambda'' = (\Delta^{\|''}, \Delta^{\perp''}, \alpha, \gamma'')$, where

$$\Delta_s^{\|''} := \begin{cases} 0 & \text{for } s = 1, \\ \Delta_{s-1}^{\|} & \text{for } 2 \le s \le l, \\ \Delta_l^{\|} + \Delta_{l+1}^{\|} & \text{for } s = l+1, \\ \Delta_s^{\|} & \text{for } l+2 \le s \le t, \end{cases}$$

$$\Delta_s^{\perp''} := \begin{cases} 0 & \text{for } s = 1, \\ {\Delta_{s-1}^{\perp}}' & \text{for } 2 \le s \le l, \\ {\Delta_l^{\perp}}' + {\Delta_{l+1}^{\perp}}' + \left|\Delta_l^{\|}\right| + \left|\Delta_{l+1}^{\|}\right| - \left|{\Delta_{l+1}^{\|}}''\right| + \alpha \sum_{i=1}^{l} \Delta_i^{\|} & \text{for } s = l+1, \\ {\Delta_s^{\perp}}' & \text{for } l+2 \le s \le t, \end{cases} \tag{40}$$

$$\gamma_s'' := \begin{cases} 0 & \text{for } s = 1, \\ \gamma_{s-1} & \text{for } 2 \le s \le l, \\ \gamma_l + \gamma_{l+1} & \text{for } s = l+1, \\ \gamma_s & \text{for } l+2 \le s \le t. \end{cases}$$

Note that λ'' satisfies (34) and (35) and the objective values of λ' and λ'' are equal. Moreover, for any $1 \le s \le l$, we know that $\Delta_s^{\|''} \ge 0$. Therefore, we iteratively construct λ'' according to the procedure stated above until $\Delta_s^{\|''} \ge 0$ for all $1 \le s \le t-1$. For the sake of notation, let $\lambda'' = (\Delta^{\|''}, \Delta^{\perp''}, \alpha, \gamma'')$ be the solution obtained in this way. If $\Delta_t^{\|''} \le 0$, then set $\hat{\lambda} = \lambda''$ to complete the proof of the claim. Otherwise, let $\hat{\lambda} = (\hat{\Delta}^{\|}, \hat{\Delta}^{\perp}, \alpha, \gamma'')$, where

$$\begin{aligned} \hat{\Delta}_s^{\|} &:= \begin{cases} \Delta_s^{\|''} & \text{for } 1 \le s \le t-1, \\ 0 & \text{for } s = t, \end{cases} \\ \hat{\Delta}_s^{\perp} &:= \begin{cases} \Delta_s^{\perp''} & \text{for } 1 \le s \le t-1, \\ \Delta_t^{\perp''} + \Delta_t^{\|''} + \alpha \Delta_t^{\|''} & \text{for } s = t. \end{cases} \end{aligned} \tag{41}$$

Note that $\hat{\lambda}$ satisfies (34) and (35), $\hat{\Delta}_t^{\|} \le 0$ and the objective values of λ'' and $\hat{\lambda}$ are equal. This concludes the proof of the claim. □

Lemma C.4 is the most crucial step in the proof of Proposition C.3. The remainder of the proof of Proposition C.3 largely builds on Lemma C.4. The second statement in Proposition C.3 follows by identifying one particular feasible solution to the linear optimization problem in (20).

Proof of Proposition C.3. Fix any $k \in \mathbb{N}$. Then,

$$\begin{aligned} \sum_{i=M_k+1}^{T_{k+1}} (2\textsc{Rob}(i) + \textsc{Adv}(i)) &\ge \sum_{i=M_k+1}^{T_{k+1}} (\textsc{Rob}(i) + d(v_i, v_{i-1}) + d(p_i, p_{i-1})) \\ &\ge \sum_{i=M_k+1}^{T_{k+1}} \textsc{Rob}(i) + d(v_{T_{k+1}}, p_{T_{k+1}}) - d(v_{M_k}, p_{M_k}) \\ &\ge (1+\delta) \sum_{i=M_k+1}^{T_{k+1}} \textsc{Adv}(i) + \delta \sum_{i=T_k}^{T_{k+1}} \textsc{Adv}(i), \end{aligned} \tag{42}$$

where the second inequality follows by the triangle inequality and the third inequality follows by line 9. Therefore, since $M_k \geq T_k + 1$,

$$\sum_{i=T_k}^{T_{k+1}} \textsc{Adv}(i) \leq \frac{2}{\delta} \sum_{i=T_k+1}^{T_{k+1}} \textsc{Rob}(i). \tag{43}$$

Let $K \in \mathbb{N}$ be the largest integer such that $T \geq T_K$. Then,

$$\begin{aligned}
\sum_{i=1}^{T_K} \textsc{Alg}(i) &= \sum_{k=1}^{K-1} \left(\sum_{i=T_k}^{T_{k+1}-1} \textsc{Alg}(i) + d(v_{T_{k+1}-1}, p_{T_{k+1}-1}) - d(v_{T_k-1}, p_{T_k-1}) \right) \\
&\quad + \textsc{Alg}(T_K) + d(p_{T_K}, p_{T_K}) - d(v_{T_K-1}, p_{T_K-1}) \\
&\leq (1+2\delta) \sum_{i=1}^{T_K} \textsc{Adv}(i) \\
&\leq (1+2\delta) \sum_{k=1}^{K-1} \sum_{i=T_k}^{T_{k+1}} \textsc{Adv}(i) \\
&\leq \left(\frac{2}{\delta} + 4 \right) \sum_{i=1}^{T_K} \textsc{Rob}(i),
\end{aligned} \tag{44}$$

where the first inequality follows by Lemma C.4 and the third inequality follows by (43). Also, if

$T \geq M_K$, then

$$\begin{aligned}
\sum_{i=T_K+1}^{T} \textsc{Alg}(i) &= \sum_{i=T_K+1}^{M_K-1} \textsc{Adv}(i) + f_{M_K}(v_{M_K}) + d(p_{M_K-1}, v_{M_K}) + \sum_{i=M_K+1}^{T} \textsc{Rob}(i) \\
&\leq \sum_{i=T_K+1}^{M_K-1} \textsc{Adv}(i) + d(p_{M_K-1}, v_{M_K-1}) + \sum_{i=M_K}^{T} \textsc{Rob}(i) \\
&\leq \sum_{i=T_K+1}^{M_K-1} \textsc{Adv}(i) + \sum_{i=1}^{M_K-1} (d(p_i, p_{i-1}) + d(v_i, v_{i-1})) + \sum_{i=M_K}^{T} \textsc{Rob}(i) \\
&\leq 2 \sum_{i=T_K+1}^{M_K-1} \textsc{Adv}(i) + \sum_{i=1}^{T_K} \textsc{Adv}(i) + \sum_{i=1}^{T} \textsc{Rob}(i) \\
&\leq 2U(\infty) \left(\sum_{i=T_K+1}^{M_K-1} \textsc{Rob}(i) + d(p_{T_K}, v_{T_K}) \right) + \sum_{i=1}^{T_K} \textsc{Adv}(i) + \sum_{i=1}^{T} \textsc{Rob}(i) \qquad (45) \\
&\leq 2U(\infty) \left(\sum_{i=T_K+1}^{M_K-1} \textsc{Rob}(i) + \sum_{i=1}^{T_K} (d(p_i, p_{i-1}) + d(v_i, v_{i-1})) \right) \\
&+ \sum_{i=1}^{T_K} \textsc{Adv}(i) + \sum_{i=1}^{T} \textsc{Rob}(i) \\
&\leq (2U(\infty) + 1) \sum_{i=1}^{T_K} \textsc{Adv}(i) + (2U(\infty) + 1) \sum_{i=1}^{T} \textsc{Rob}(i) \\
&\leq \left(\frac{4U(\infty) + 2}{\delta} + 2U(\infty) + 1 \right) \sum_{i=1}^{T} \textsc{Rob}(i),
\end{aligned}$$

where the first and second inequalities follow by the triangle inequality, the fourth inequality follows by Lemma C.5, the fifth inequality follows again by the triangle inequality and the seventh inequality follows by (43). If instead $T \leq M_K - 1$, then

$$\begin{aligned}
\sum_{i=T_K+1}^{T} \textsc{Alg}(i) &= \sum_{i=T_K+1}^{T} \textsc{Adv}(i) \\
&\leq U(\infty) \left(\sum_{i=T_K+1}^{T} \textsc{Rob}(i) + d(p_{T_K}, v_{T_K}) \right) \\
&\leq U(\infty) \left(\sum_{i=T_K+1}^{T} \textsc{Rob}(i) + \sum_{i=1}^{T_K} (d(p_i, p_{i-1}) + d(v_i, v_{i-1})) \right) \qquad (46) \\
&\leq U(\infty) \sum_{i=1}^{T_K} \textsc{Adv}(i) + U(\infty) \sum_{i=1}^{T} \textsc{Rob}(i) \\
&\leq \left(\frac{2U(\infty)}{\delta} + 1 \right) \sum_{i=1}^{T} \textsc{Rob}(i),
\end{aligned}$$

where the first inequality follows by Lemma C.5, the second inequality follows by the triangle inequality and the fourth inequality follows by (43). The proof of the first statement is completed by adding (44) and either (45) or (46).

To see the proof of the second statement, assume that $\frac{2}{\alpha\delta} \in \mathbb{N}$. Fix any $t \geq 1$. Note that

$$\begin{aligned} U &= \alpha \left(\frac{2}{\alpha + \delta(1+\alpha)} \right)^{2/\alpha\delta} + \frac{2}{(2-\alpha-\delta(1+\alpha))^2} \left(\alpha \left(\frac{2}{\alpha+\delta(1+\alpha)} \right)^{2/\alpha\delta} - \frac{2-\alpha}{\delta} + 1 \right), \\ y_{t-s} &= \left(\frac{2}{\alpha+\delta(1+\alpha)} \right)^{s} \left(\frac{2-(s-1)^{+}\alpha\delta}{2\delta} \right)^{+} \quad \text{for } 0 \leq s \leq t-1, \end{aligned} \tag{47}$$

is feasible for the optimization problem in (20). Therefore, $U(t) \leq \tilde{U}$ for all $t \geq 1$. This completes the proof of the proposition. □

D Proof of Theorem 3.2

In this section, we will prove Theorem 3.2 by proving the next, stronger proposition.

Proposition D.1. *Fix any $\delta > 0$. Let $\mathcal{A}$ be any deterministic algorithm for the non-convex optimization problem in* (4)*. If there exists $0 < \varepsilon < \delta$ such that $\mathcal{A}$ is $(1+\varepsilon)$-consistent, then $\mathcal{A}$ is at least $\sup_{t\geq 1} L(t)$-robust, where*

$$\begin{aligned} L(t) := \max_{\Delta} \quad & \sum_{i=1}^{t} \left(\Delta_i + \alpha \left(1 + \sum_{j=1}^{i} \Delta_j \right) \right) \\ \textit{s.t.} \quad 2 \left(1 + \sum_{i=1}^{s-1} \Delta_i \right) &\geq \delta \sum_{i=1}^{s} \left(\Delta_i + \alpha \left(1 + \sum_{j=1}^{i} \Delta_j \right) \right) + \alpha \left(1 + \sum_{i=1}^{s} \Delta_i \right) \quad \textit{for } 1 \leq s \leq t, \\ \Delta_s &\geq 0 \quad \textit{for } 1 \leq s \leq t. \end{aligned} \tag{48}$$

Moreover, assume that $\frac{2-\alpha(1-\delta^2)}{\alpha\delta(1+\delta)} \in \mathbb{N}$, then

$$\sup_{t\geq 1} L(t) \geq \frac{\alpha\delta}{4} \left(\frac{2}{\alpha+\delta(1+\alpha)} \right)^{\frac{2-\alpha(1-\delta^2)}{\alpha\delta(1+\delta)}} - \mathcal{O}(1), \tag{49}$$

where the $\mathcal{O}$-notation holds in the limit $\alpha, \delta \to 0$.

Note that Theorem 3.2 follows directly from Proposition D.1. Also, note that the assumption that $\frac{2-\alpha(1-\delta^2)}{\alpha\delta(1+\delta)} \in \mathbb{N}$ is again without loss of generality, but prevents rounding symbols from appearing in the notation. The proof of Proposition D.1 depends on the following idea: If Δ in (48) somehow represents the movement of the advice in one dimension, then the constraints are sufficient to let any algorithm follow the advice exactly, otherwise the algorithm would violate the assumption of $(1+\delta)$-consistency. Then, we want the advice trajectory which has the maximum cost given these constraints. This results in the maximization problem in (48). It turns out that a worst-case solution to (48) and hence a worst-case instance is when the advice moves away exponentially fast from the minimizer.

Proof of Proposition D.1. Fix any $\delta > 0$, and let $\mathcal{A}$ be any deterministic algorithm. Assume, for the sake of contradiction, that there exists $\varepsilon < \delta$ such that $\mathcal{A}$ is $(1+\varepsilon)$-consistent and L-robust, where $L < \sup_{t \geq 1} L(t)$. Then, let $t \geq 1$ be such that $L(t) > L$ and Δ be an optimal solution to (48). We will construct an instance in the metric space $M = \mathbb{R}$, where $d(x,y) = |x-y|$ and $x_0 = v_0 = p_0 = 0$. Suppose $v_s = -1$, $p_s = \sum_{i=1}^{s} \Delta_i$ and $f_s(x) = \alpha|x - v_s| + \infty \cdot \mathbf{1}_{x \notin \{v_s, p_s\}}$ for $1 \leq s \leq t$. Note that $x_s \in \{v_s, p_s\}$ for all $1 \leq s \leq t$; otherwise, the algorithm would incur an infinite cost, while the algorithm following the minimizer has a finite cost and the proof of the proposition follows trivially. Even more strongly, we claim that $x_s = p_s$ for all $1 \leq s \leq t$. Assuming the claim to be true, observe that

$$\begin{aligned}\sum_{i=1}^{t} \textsc{Alg}(i) = \sum_{i=1}^{t} \textsc{Adv}(i) = \sum_{i=1}^{t} \left(\Delta_i + \alpha \left(1 + \sum_{j=1}^{i} \Delta_j \right) \right) \\ = L(t) = L(t) \sum_{i=1}^{t} \textsc{Rob}(i) \geq L(t) \sum_{i=1}^{t} \textsc{Opt}(i) > L \sum_{i=1}^{t} \textsc{Opt}(i),\end{aligned} \tag{50}$$

which violates L-robustness. This is a contradiction and the proof of the proposition follows. We now provide the proof of the claim that $x_s = p_s$ for all $1 \leq s \leq t$. For the sake of contradiction, let $1 \leq l \leq t$ be the smallest integer such that $x_l = v_l$. As the adversary, we modify the instance for $s \geq l+1$ such that $v_s = p_s = p_l$, $f_s(x) = \infty \cdot \mathbf{1}_{x \notin \{p_l\}}$ for $l+1 \leq s \leq t+1$. Note that $x_s = p_l$ for $l+1 \leq s \leq t+1$; otherwise, the algorithm would incur an infinite cost, while the algorithm following the minimizers has a finite cost and the proof of the proposition follows trivially. Then,

$$\begin{aligned}\sum_{i=1}^{t+1} \textsc{Alg}(i) &= \sum_{i=1}^{l-1} \textsc{Adv}(i) + |v_l - p_{l-1}| + |p_l - v_l| = \sum_{i=1}^{l-1} \textsc{Adv}(i) + \left(1 + \sum_{i=1}^{l-1} \Delta_i \right) + \left(1 + \sum_{i=1}^{l} \Delta_i \right) \\ &\geq \sum_{i=1}^{l-1} \textsc{Adv}(i) + \delta \sum_{i=1}^{l} \left(\Delta_i + \alpha \left(1 + \sum_{j=1}^{i} \Delta_j \right) \right) + \Delta_l + \alpha \left(1 + \sum_{i=1}^{l} \Delta_i \right) \\ &= \sum_{i=1}^{l-1} \textsc{Adv}(i) + \delta \sum_{i=1}^{l} \textsc{Adv}(i) + \textsc{Adv}(l) = (1+\delta) \sum_{i=1}^{t+1} \textsc{Adv}(i),\end{aligned} \tag{51}$$

where the inequality follows by the first constraint in (48). This violates the assumption that there exists $\varepsilon < \delta$ such that $\mathcal{A}$ is $(1+\varepsilon)$-consistent. Now, to see the second statement (49) in the proposition, note that

$$\Delta_s = \frac{2 - \alpha(1-\delta^2) - s\alpha\delta(1+\delta)}{2} \left(\frac{2}{\alpha + \delta(1+\alpha)} \right)^s \text{ for } 1 \leq s \leq t, \tag{52}$$

is feasible for the optimization problem in (48) for $t = \frac{2-\alpha(1-\delta^2)}{\alpha\delta(1+\delta)}$. Also, with this definition,

$$\begin{aligned}\sum_{i=1}^{t} \left(\Delta_i + \alpha \left(1 + \sum_{j=1}^{i} \Delta_j \right) \right) = \frac{\alpha\delta(1+\delta)(2+3\alpha-\delta(1+\alpha))}{(2-\alpha-\delta(1+\alpha))^3} \left(\frac{2}{\alpha+\delta(1+\alpha)} \right)^{\frac{2-\alpha(1-\delta^2)}{\alpha\delta(1+\delta)}} \\ + \frac{2\alpha^2(1+\delta)(2-4\delta+\delta^2-\alpha(1+\delta)) + 2\alpha(2-\delta)(2-\delta^2)}{(2-\alpha-\delta(1+\alpha))^3} - \frac{2(2-\delta)^2(2+\delta)}{(1+\delta)(2-\alpha-\delta(1+\alpha))^3}.\end{aligned} \tag{53}$$

This completes the proof of the proposition. ☐

E Proof of Theorem 3.4

We start by formally stating the definitions of memoryless and scale- and rotation-invariance.

Definition E.1. *An algorithm $\mathcal{A}$ for online optimization with switching costs and predictions is* ***memoryless*** *if its decision x_t at time t depends only on x_{t-1}, $\tilde{x}_t$, and f_t.*

Definition E.2. *A memoryless algorithm $\mathcal{A}$ for online optimization with switching costs and predictions is* ***scale-invariant*** *if, when M is a real normed vector space, the behavior of $\mathcal{A}$ is invariant under scaling of M. That is, if x_t is the decision $\mathcal{A}$ makes at time t when faced with x_{t-1}, $\tilde{x}_t$, and f_t, then $\mathcal{A}$ is scale-invariant if, for all $\lambda > 0$, its decision is λx_t when faced with previous decision λx_{t-1}, prediction $\lambda \tilde{x}_t$, and hitting cost $f_t(\lambda \cdot)$.*

We say $\mathcal{A}$ is ***rotation-invariant*** *if, when M is a real inner product space, the behavior of $\mathcal{A}$ is invariant under rotation of M. That is, if for all rotation operators U, $\mathcal{A}$ makes decision Ux_t when its previous decision was Ux_{t-1}, advice is Up_t, and the hitting cost is $f_t(U\cdot)$.*

Proof. We prove Theorem 3.4 in the metric space $M = \mathbb{R}^2$, where $d(x, y) = \|x - y\|_2$ is the Euclidian distance. Let $r_1 \leq r_2 < \sqrt{2}/2$ be two arbitrary numbers to be decided later. Let $x_0 = (0, r_2)$ and $f_1(x, y) = \alpha|x - v_t| + Ly$. We let $L \to \infty$ such that any algorithm must move onto the x-axis to achieve a bounded cost (i.e. $x_{1,2} = 0$). Also, let $v_1 = -\frac{r_2^2 - r_1^2}{2r_1}$ and $\tilde{x}_1 = \sqrt{1 - r_2^2}$. We distinguish two cases.

(i) Assume that $x_{1,1} > r_1$. Then, note that $\|x_1 - v_1\|_2 \geq \frac{r_1^2 + r_2^2}{2r_1} = \|x_0 - v_1\|_2$. In other words, the online algorithm moves further away from the minimizer. At time $t = 2$, we rotate the function f_1 around v_1 and position $\tilde{x}_2$ such that the setup at the next time step is exactly equivalent (up to rotation and scaling) to the setup at time $t = 1$. As $\mathcal{A}$ is memoryless and scale- and rotation invariant, $\mathcal{A}$ will move exactly equivalent as at time $t = 1$ and therefore again move further away from the minimizer. We repeat this setup infinitely often. At each time step the algorithm $\mathcal{A}$ incurs a cost of at least $\sqrt{r_1^2 + r_2^2}$. In contrast, the optimal algorithm moves to v_1 at time $t = 1$ and incurs a one-time moving cost of $\frac{r_1^2 + r_2^2}{2r_1}$. The competitive ratio of $\mathcal{A}$ is therefore unbounded, regardless of the values of r_1 and r_2.

(ii) Assume that $x_{1,1} \leq r_1$. At time $t = 2$, we rotate the function f_1 around $\tilde{x}_1$ and position v_2 such that the setup at the next time step is exactly equivalent (up to rotation and scaling) to the setup at time $t = 1$. As $\mathcal{A}$ is memoryless and scale- and rotation invariant, $\mathcal{A}$ will move exactly equivalent as at time $t = 1$. We repeat this setup infinitely often. At time $t = 1$, the algorithm incurs a cost of $\sqrt{x_{1,1}^2 + r_2^2} + \alpha \left| x_{1,1} + \frac{r_2^2 - r_1^2}{2r_1} \right|$. Moreover, the distance to $\tilde{x}_1$ changes from $\|x_0 - \tilde{x}_1\|_2 = 1$ to $\|x_1 - \tilde{x}_1\|_2 = \sqrt{1 - r_2^2} - x_{1,1} > 0$. Hence, at time $t = 2$ a setup is presented where each distance is scaled by a factor of $\sqrt{1 - r_2^2} - x_{1,1}$. If $\sqrt{1 - r_2^2} - x_{1,1} \neq 1$, then the cost of the algorithm from time $t = 1$ to T is

$$\sum_{t=1}^{T} \left(\sqrt{1 - r_2^2} - x_{1,1} \right)^{t-1} \left(\sqrt{x_{1,1}^2 + r_2^2} + \alpha \left| x_{1,1} + \frac{r_2^2 - r_1^2}{2r_1} \right| \right)$$
$$= \frac{\left(1 - \left(\sqrt{1 - r_2^2} - x_{1,1}\right)^T\right) \left(\sqrt{x_{1,1}^2 + r_2^2} + \alpha \left| x_{1,1} + \frac{r_2^2 - r_1^2}{2r_1} \right| \right)}{1 - \left(\sqrt{1 - r_2^2} - x_{1,1}\right)}. \quad (54)$$

The cost of the optimal algorithm from time $t = 1$ to T is

$$1 + \sum_{t=1}^{T} \left(\sqrt{1 - r_2^2} - x_{1,1}\right)^{t-1} \alpha \left(\frac{r_1^2 + r_2^2}{2r_1} + \sqrt{1 - r_2^2}\right)$$
$$= 1 + \frac{\left(1 - \left(\sqrt{1 - r_2^2} - x_{1,1}\right)^T\right) \alpha \left(\frac{r_1^2 + r_2^2}{2r_1} + \sqrt{1 - r_2^2}\right)}{1 - \left(\sqrt{1 - r_2^2} - x_{1,1}\right)}. \quad (55)$$

Let $r_1 = \alpha$ and $r_2 = \sqrt{2\alpha}$. We distinguish two more cases. If $\sqrt{1 - r_2^2} - x_{1,1} \geq 1$, then, as $T \to \infty$, the competitive ratio of $\mathcal{A}$ is at least

$$\mathrm{CR} \geq \frac{\sqrt{x_{1,1}^2 + r_2^2} + \alpha \left| x_{1,1} + \frac{r_2^2 - r_1^2}{2r_1} \right|}{\alpha \left(\frac{r_1^2 + r_2^2}{2r_1} + \sqrt{1 - r_2^2}\right)} \geq \frac{\sqrt{2 - 2\sqrt{1 - 2\alpha}}}{\alpha \left(\frac{\alpha^2 + 2\alpha}{2\alpha} + \sqrt{1 - 2\alpha}\right)} = \frac{1}{\sqrt{2\alpha}} - o\left(\frac{1}{\sqrt{\alpha}}\right). \quad (56)$$

If $\sqrt{1 - r_2^2} - x_{1,1} < 1$, then, as $T \to \infty$, the competitive ratio of $\mathcal{A}$ is at least

$$\mathrm{CR} \geq \frac{\sqrt{x_{1,1}^2 + r_2^2} + \alpha \left(x_{1,1} + \frac{r_2^2 - r_1^2}{2r_1}\right)}{1 - \left(\sqrt{1 - r_2^2} - x_{1,1}\right) + \alpha \left(\frac{r_1^2 + r_2^2}{2r_1} + \sqrt{1 - r_2^2}\right)}$$
$$\geq \frac{\sqrt{\alpha^2 + 2\alpha} + \alpha \left(\alpha + \frac{2\alpha - \alpha^2}{2\alpha}\right)}{1 - \left(\sqrt{1 - 2\alpha} - \alpha\right) + \alpha \left(\frac{\alpha^2 + 2\alpha}{2\alpha} + \sqrt{1 - 2\alpha}\right)} = \frac{1}{\sqrt{8\alpha}} - o\left(\frac{1}{\sqrt{\alpha}}\right). \quad (57)$$

□

F Proof of Theorem 3.5

Let $y \in \mathbb{R}^T$ be an arbitrary solution and define the potential function $\phi(y_t, x_t) := c|y_t - x_t|$ for $c > 0$. Note that if we prove that

$$f_t(x_t) + |x_t - x_{t-1}| + \phi(y_t, x_t) - \phi(y_{t-1}, x_{t-1}) \leq \mathrm{CR}\left(f_t(y_t) + |y_t - y_{t-1}|\right), \quad (58)$$

for all $1 \leq t \leq T$, then, if we sum over t, we obtain

$$\begin{aligned} \sum_{i=1}^{T} \left(f_t(x_t) + |x_t - x_{t-1}|\right) &\leq \mathrm{CR} \sum_{i=1}^{T} \left(f_t(y_t) + |y_t - y_{t-1}|\right) - \phi(y_T, x_T) \\ &\leq \mathrm{CR} \sum_{i=1}^{T} \left(f_t(y_t) + |y_t - y_{t-1}|\right), \end{aligned} \quad (59)$$

which proves that x is CR-competitive *with respect to* y. We will apply this technique to $y = o$ and $y = p$ separately to find the competitive ratio with respect to the hindsight optimal and the advice, respectively. Applying the triangle equality to $\phi(y_t, x_t) - \phi(y_{t-1}, x_{t-1})$ yields

$$\begin{aligned} \phi(y_t, x_t) - \phi(y_{t-1}, x_{t-1}) &= c\left(|y_t - x_t| - |y_{t-1} - x_{t-1}|\right) \\ &\leq c\left(|y_t - y_{t-1}| + |y_t - x_t| - |y_t - x_{t-1}|\right) \\ &\leq \mathrm{CR} \cdot |y_t - y_{t-1}| + c\left(|y_t - x_t| - |y_t - x_{t-1}|\right), \end{aligned} \quad (60)$$

where we assume that $\mathrm{CR} \geq c$. This means it is sufficient to prove that

$$f_t(x_t) + |x_t - x_{t-1}| + c\left(|y_t - x_t| - |y_t - x_{t-1}|\right) \leq \mathrm{CR} \cdot f_t(y_t). \tag{61}$$

We verify equation (61) in the case that $y = o$ first. Let $c = 1 + \underline{\beta}^{-1}$ and $\mathrm{CR} = 1 + (2 + \underline{\beta}^{-1})\bar{\beta}$ as in the theorem. We distinguish two cases.

1. Assume that $f_t(x_t) \leq f_t(o_t)$. Note that in any case

$$|x_t - x_{t-1}| \leq |x(\bar{\lambda}) - x_{t-1}| \leq \bar{\beta} f_t(x(\bar{\lambda})) \leq \bar{\beta} f_t(x_t), \tag{62}$$

   where the last inequality follows by the convexity of f_t and hence, by applying the triangle inequality,

$$\begin{aligned} f_t(x_t) + |x_t - x_{t-1}| + c\left(|o_t - x_t| - |o_t - x_{t-1}|\right) \\ &\leq f_t(x_t) + (1+c)|x_t - x_{t-1}| \\ &\leq \left(1 + (1+c)\bar{\beta}\right) f_t(x_t) \leq \mathrm{CR} \cdot f_t(o_t), \end{aligned} \tag{63}$$

   which verifies equation (61).

2. Assume that $f_t(x_t) > f_t(o_o)$. Note that in this case x_t did not reach v_t which means that $\underline{\lambda} < 1$ and $|x(\underline{\lambda}) - x_{t-1}| = \underline{\beta} f_t(x(\underline{\lambda}))$. Thus,

$$f_t(x_t) \leq f_t(x(\underline{\lambda})) = \frac{|x(\underline{\lambda}) - x_{t-1}|}{\underline{\beta}} \leq \frac{|x_t - x_{t-1}|}{\underline{\beta}}. \tag{64}$$

   Moreover, since $f_t(x_t) > f_t(o_t)$, x_t must have moved closer to o_t during its entire move and thus

$$c\left(|o_t - x_t| - |o_t - x_{t-1}|\right) = -c|x_t - x_{t-1}|. \tag{65}$$

   Therefore,

$$f_t(x_t) + |x_t - x_{t-1}| + c\left(|o_t - x_t| - |o_t - x_{t-1}|\right) \leq \left(1 + \underline{\beta}^{-1} - c\right)|x_t - x_{t-1}| \leq 0, \tag{66}$$

   which verifies equation (61).

We now continue to verify equation (61) in the case that $y = p$. Let $c = 1 + \bar{\beta}^{-1}$ and $\mathrm{CR} = 1 + (2 + \bar{\beta}^{-1})\underline{\beta}$ as in the theorem. We distinguish three cases.

1. Assume that $f_t(x_t) \leq f_t(p_t)$ and $x_t \neq p_t$. Either p_t is on the opposite side of v_t as x_t or p_t is on the same side of v_t as x_t. If p_t is on the same side, then it must be that $x_t = x(\underline{\lambda})$, since by decreasing λ, the point $x(\lambda)$ only moves closer to p_t. Then, by applying the triangle inequality,

$$\begin{aligned} f_t(x_t) + |x_t - x_{t-1}| + c\left(|p_t - x_t| - |p_t - x_{t-1}|\right) &\leq f_t(x_t) + (1+c)|x_t - x_{t-1}| \\ &\leq \left(1 + (1+c)\underline{\beta}\right) f_t(x_t) \leq \mathrm{CR} \cdot f_t(p_t), \end{aligned} \tag{67}$$

   which verifies equation (61). If p_t is on the opposite side of v_t, then x_t must have moved closer to p_t during its entire move and thus

$$c\left(|p_t - x_t| - |p_t - x_{t-1}|\right) = -c|x_t - x_{t-1}|. \tag{68}$$

   Therefore,

$$f_t(x_t) + |x_t - x_{t-1}| + c\left(|p_t - x_t| - |p_t - x_{t-1}|\right) \leq f_t(x_t) \leq \mathrm{CR} \cdot f_t(p_t), \tag{69}$$

   which verifies equation (61).

2. Assume that $x_t = p_t$. Then,

$$f_t(x_t) + |x_t - x_{t-1}| + c\left(|p_t - x_t| - |p_t - x_{t-1}|\right) \leq f_t(p_t) \leq \mathrm{CR} \cdot f_t(p_t) \tag{70}$$

which verifies equation (61).

3. Assume that $f_t(x_t) > f_t(p_t)$. Then it must be that $x_t = x(\bar{\lambda})$, since by increasing λ, the point $x(\lambda)$ only moves closer to p_t. Moreover, x_t did not reach v_t which means that $\bar{\lambda} < 1$ and thus

$$f_t(x_t) = f_t(x(\bar{\lambda})) = \frac{|x(\bar{\lambda}) - x_{t-1}|}{\bar{\beta}} = \frac{|x_t - x_{t-1}|}{\bar{\beta}}. \tag{71}$$

Also, since $f_t(x_t) > f_t(p_t)$, x_t must have moved closer to o_t during its entire move and thus

$$c\left(|p_t - x_t| - |p_t - x_{t-1}|\right) = -c|x_t - x_{t-1}|. \tag{72}$$

Therefore,

$$f_t(x_t) + |x_t - x_{t-1}| + c\left(|p_t - x_t| - |p_t - x_{t-1}|\right) = \left(1 + \bar{\beta}^{-1} - c\right)|x_t - x_{t-1}| \leq 0, \tag{73}$$

which verifies equation (61).

This completes the proof of the theorem by applying Lemma C.1. □

G Proof of Theorem 3.6

Let $\mathcal{A}$ be any deterministic algorithm for the convex, one-dimensional optimization problem and fix any $0 < \delta < 1/2$. Let x denote the decisions of $\mathcal{A}$. We construct an instance. Let $\tilde{x}_0 = x_0 = 0$, $f_1(x) = 2\delta|x - 1|$, the advice $\tilde{x}_1 = 1$ and $T = 2$. We distinguish two cases.

1. Assume that $x_1 \geq \frac{1}{2}$. Let $f_2(x) = |x|$ and the advice $\tilde{x}_2 = 0$. Then, the optimal solution incurs a cost of 2δ with $o_1 = o_2 = 0$. The algorithm $\mathcal{A}$ has a cost of at least 1 and hence the competitive ratio is at least $1/(2\delta)$.

2. Assume that $x_1 < \frac{1}{2}$. Let $f_2(x) = |x - 1|$ and the advice $\tilde{x}_2 = 1$. Then, the optimal solution incurs a cost of 1 with $o_1 = o_2 = 1$. However, the algorithm $\mathcal{A}$ has a cost of at least $1 + 2\delta(1 - x_1) > 1 + \delta$ even though the predictions are perfect. Hence, this case does not satisfy the assumption that $\mathcal{A}$ is $(1 + \delta)$-consistent.

□