

# Grinding the Space: Learning to Classify Against Strategic Agents

Yiling Chen  
Harvard University  
yiling@seas.harvard.edu

Yang Liu  
UC Santa Cruz  
yangliu@ucsc.edu

Chara Podimata  
Harvard University  
podimata@g.harvard.edu

## Abstract

We study the problem of online learning in *strategic* classification settings from the perspective of the learner, who is repeatedly facing myopically rational strategic agents. We model this interplay as a repeated Stackelberg game, where at each timestep the learner deploys a high-dimensional linear classifier first and an agent, after observing the classifier, along with his real feature vector, and according to his underlying utility function, best-responds with a (potentially altered) feature vector. We measure the performance of the learner in terms of *Stackelberg regret* for her 0 – 1 loss function.

Surprisingly, we prove that in strategic settings like the one considered in this paper there exist worst-case scenarios, where *any* sequence of actions providing sublinear external regret might result in linear Stackelberg regret and vice versa. We then provide the GRINDER Algorithm, an *adaptive discretization* algorithm, potentially of independent interest in the online learning community, and prove its data-dependent upper bound on the Stackelberg regret given oracle access, while being computationally efficient. We also provide a *nearly matching* lower bound for the problem of strategic classification. We complement our theoretical analysis with simulation results, which suggest that our algorithm outperforms the benchmarks, *even* given access to approximation oracles. Our results advance the known state-of-the-art results in the growing literature of online learning from revealed preferences, which has so far focused on “smoother” utility and loss functions from the perspective of the agents and the learner respectively.

# 1 Introduction

As Machine Learning (ML) algorithms are getting more and more involved in real-life decision making, the agents that we normally face are neither stochastic, nor adversarial. Rather, they seem to be *strategic*. Think about a college that wishes to deploy an ML algorithm to make the admissions decisions. Student candidates might try to manipulate their test scores in an effort to fool the classifier<sup>1</sup>. Importantly, however, these student candidates *do* want to be admitted; they do not simply want to sabotage the admissions algorithm. And this is precisely what differentiates them from being fully adversarial. Similar situations arise in almost every deployment of ML algorithms in real-life settings. As such, strategic agents present a unique threat, but also a unique opportunity for ML researchers.

Compared to classical adversarial models in ML, strategic models have a unique advantage; namely, if the incentives of the agents are aligned *properly* (i.e., through the use of payments, or through the use of specifically designed mechanisms that are robust to strategic noise), then it is possible for the learner to obtain a virtually *clean* dataset. In the language of game theory, such mechanisms render the respective ML tasks that they are applied to *strategyproof*, and there has been an increasing interest in these types of mechanisms, especially for the tasks of classification [38, 39] and regression [24, 23, 21]. Despite the various encouraging results, strategyproofness remains a very hard desideratum to achieve for dynamic, real-life settings, like the one we consider in this paper. The more relaxed solution concept that has emerged as the appropriate for these dynamic settings is the one of *Stackelberg regret*, where a learner compares her cumulative loss with the cumulative loss of her best-fixed action in hindsight, had she given the agent the opportunity to best-respond to it.

In this work, we focus on the problem of learning an unknown linear classifier, when the data that we use come in an online fashion from strategic agents, who can alter their feature vectors in an effort to *game* the classifier. We model the interplay between the learner and the strategic agents (henceforth referred to simply as “agents”) as a repeated *Stackelberg game* (named after Heinrich Freiherr von Stackelberg, because of his work in [49]), which happens over  $T$  timesteps. In general, at a repeated Stackelberg game, the learner (called the “leader” in the language of Stackelberg games) *commits* to an action, and subsequently, the agent (called the “follower” in the language of Stackelberg games) best-responds to it, i.e., chooses an action that maximizes his underlying utility. Our main technical contribution is the GRINDER algorithm, an adaptive discretization algorithm, which provably achieves nearly optimal performance guarantees with respect to the Stackelberg regret for the problem of strategic classification. We find it useful to formally introduce our model, before discussing our results in more depth.

## 1.1 Model

**Protocol.** In our strategic classification setting, let  $d \in \mathbb{N}$  denote the dimension of the problem and  $\mathcal{A} \subseteq [-1, +1]^{d+1}$  the learner’s action space<sup>2</sup>. Actions  $\alpha \in \mathcal{A}$  correspond to hyperplanes, written in terms of their normal vectors, and without loss of generality, we assume that the  $(d + 1)$ -th coordinate encodes information about the intercept. Let  $h^* : \mathcal{X} \rightarrow \{-1, 1\}$  an *ideal* classifier, not necessarily belonging in  $\mathcal{A}$ , which we assume makes *no* error at identifying the labels of the feature

---

<sup>1</sup>And, indeed, there is ample evidence they would do so [48].

<sup>2</sup>This is without loss of generality, as the normal vector of any hyperplane could be normalized.

vectors that it is given<sup>3</sup>. We use the construction of  $h^*$  solely as a modeling tool, and our results assume no access to it. For a feature vector  $\mathbf{u}$ , we refer to  $h^*(\mathbf{u})$  as its *true* label. Formally, the protocol of the interaction of the learner with the agents (which repeats for all timesteps  $t \in [T]$ ) is given in Protocol 1. We note that we use the notation  $([0, 1]^d, 1)$  to denote the  $(d + 1)$ -dimensional vectors having any value from  $[0, 1]$  in the first  $d$  dimensions, and 1 in the  $(d + 1)$ -th one.

---

**Protocol 1** Learner-Agent Interaction at Round  $t$

---

- 1: The environment adversarially chooses feature vector  $\mathbf{x}_t \in \mathcal{X} \subseteq ([0, 1]^d, 1)$ .
  - 2: The learner chooses action  $\alpha_t \in \mathcal{A}$ , and commits to it.
  - 3: An agent observes  $\alpha_t$  and  $\sigma_t = (\mathbf{x}_t, y_t)$ , where  $y_t = h^*(\mathbf{x}_t)$ .
  - 4: The agent reports feature vector  $\mathbf{z}_t(\alpha_t; \sigma_t) \in \mathcal{X}$  (potentially,  $\mathbf{z}_t(\alpha_t; \sigma_t) \neq \mathbf{x}_t$ ).
  - 5: The learner observes  $(\mathbf{z}_t(\alpha_t; \sigma_t), \hat{y}_t)$ , where  $\hat{y}_t = h^*(\mathbf{z}_t(\alpha_t; \sigma_t))$ , and incurs binary classification loss  $\ell(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) = \mathbb{1}\{\text{sgn}(\hat{y}_t \cdot \langle \alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t) \rangle) = -1\}$ .
- 

**Agents’ Behavior in Step 4.** When the agent reports  $\mathbf{z}_t(\alpha_t; \sigma_t)$ , instead of his true (i.e., assigned by the environment) feature vector  $\mathbf{x}_t$ , he gets *value*  $v_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) \in [0, 1]$ , and incurs cost  $c_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) \geq 0$ . Altogether, the agent’s utility for reporting  $\mathbf{z}_t(\alpha_t; \sigma_t)$  is equal to:  $u_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) = \delta \cdot v_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) - c_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t))$  for a scalar  $\delta \in (0, 1]$ , which is *known* to the learner. To summarize, in our model the learner does *not* know the specific value and cost functions, which can be arbitrary<sup>4</sup>. What she does know is the *general form* of the utility function (i.e., value – cost) and the scalar  $\delta$ . In terms of behavior, the agents are *myopically rational*, i.e., they will report  $\mathbf{r}_t(\alpha_t; \sigma_t) = \arg \max_{\mathbf{z}_t \in \mathcal{X}; \sigma_t} u_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t))$  in Step 4 of Protocol 1<sup>5</sup>. We will alternatively refer to the agents as being *myopically rational and  $\delta$ -bounded*.

Motivated by the spam emails setting, we assume that the agents derive no value if, by manipulating their feature vector from  $\mathbf{x}_t$  to  $\mathbf{z}_t(\alpha_t; \sigma_t)$ , they also change  $y_t$ . Indeed, a spammer ( $y_t = -1$ ) wishes to fool the learner’s classifier, without actually having to change their email to be a non-spam one ( $\hat{y}_t = 1$ ). So since for symmetric losses (e.g.,  $0 - 1$  as we use), agents derive no value when  $\hat{y}_t \neq y_t$ , then by observing  $\hat{y}_t$ , the learner essentially observes  $y_t$ . We, hence, drop the notation  $\hat{y}_t$  and only use  $y_t$  for the rest of the paper. Finally, in such a spam emails setting, the value of the agent might correspond to the value that he has for being labeled as  $+1$  by the learner’s classifier, and the cost might correspond to the effort for altering their true datapoint  $\sigma_t$  to  $\mathbf{z}_t(\alpha_t; \sigma_t)$ .

**Model Comparison with Dong et al. [28].** Prior to our work, Dong et al. [28] studied online learning in strategic classification settings. Their work is orthogonal to ours in one key aspect: they find the appropriate conditions which can guarantee that the best-response of an agent, written as a function of the learner’s action, is concave. As a result, in their model the learner’s loss function becomes *convex* and well known online convex optimization algorithms could be applied (e.g., [29, 19]) in conjunction with the mixture feedback that the learner receives. The foundation of our work, however, is to provide solutions (potentially of independent interest) for a setting where standard online learning algorithms could not be applied. Additionally, our work focuses on binary

---

<sup>3</sup>One could think of  $h^*(\mathbf{x})$  as the label that a human verifier would assign to feature vector  $\mathbf{x}$ .

<sup>4</sup>For example, the agents’ value function could be  $v_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) = \mathbb{1}\{\langle \alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t) \rangle \geq 0\} \cdot \mathbb{1}\{\hat{y}_t = y_t\}$ , while the cost function could be  $c_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) = (\mathbf{x}_t - \mathbf{z}_t(\alpha_t; \sigma_t))^2$ .

<sup>5</sup>When clear from context, we will write  $\mathbf{r}_t(\alpha_t)$  instead of  $\mathbf{r}_t(\alpha_t; \sigma_t)$ , in order to simplify notation.

loss functions for the learner, while being able to accommodate even 0 – 1 value functions, thus answering the open questions posed by Dong et al. [28]. More broadly, our work is situated at the interace of three research areas: learning using data from strategic data sources, Stackelberg games and multi-armed bandits. We defer the extended discussion on the related work to Section 6.

**Solution Concept.** The goal of the learner is to minimize her Stackelberg regret. In the language of strategic classification, we will compare the loss of the learner’s deployed linear classifiers against the loss of the best fixed linear classifier in hindsight, had the learner given the agents the opportunity to best-respond to it.

## 1.2 Our Results and Techniques

But why would one consider the arguably weirder notion of Stackelberg regret in the first place, when there are a lot of works in online learning reducing the standard notion of external regret to other notions? Our first result answers this question; surprisingly, we show that despite our intuition that there is a clear hierarchy between the notions of external and Stackelberg regret, the two are *strongly incompatible* for strategic classification settings of interest; i.e., there exist strategic classification settings against myopically rational  $\delta$ -bounded agents, where *any* algorithm achieving *sublinear* Stackelberg regret will incur *linear* external regret, and *any* algorithm achieving *sublinear* external regret would suffer from *linear* Stackelberg regret.

**Theorem 1.1** (Informal). *There are strategic classification settings against myopically rational  $\delta$ -bounded agents, where the external and the Stackelberg regret are strongly incompatible.*

As a result, for online learning in strategic classification settings, since measuring the external regret of an algorithm would not give us enough valuable information, we should not readily apply standard no-external regret algorithms and expect to achieve good Stackelberg regret guarantees. The situation becomes much harder due to the fact that the agent best-responds to the learner’s actions according to an *unknown* utility function, and in the worst case the best-response-induced loss function of the learner is *not* even Lipschitz, as we see in Example 1.2.

**Example 1.2.** *Take for example the valuation and the cost functions for the agents that we mentioned in Subsection 1.1, i.e., for learner’s action  $\alpha \in \mathcal{A}$ :  $v(\alpha, \mathbf{z}(\alpha; \sigma)) = \delta \mathbb{1}\{\langle \alpha, \mathbf{z}(\alpha; \sigma) \rangle \geq 0\} \cdot \mathbb{1}\{\hat{y} = y\}$  and  $c(\alpha, \mathbf{z}(\alpha; \sigma)) = (\mathbf{x} - \mathbf{z}(\alpha; \sigma))^2$ , and the binary loss for the learner (i.e.,  $\ell(\alpha', \mathbf{z}(\alpha'; \sigma)) = \mathbb{1}\{\text{sgn}(\hat{y} \cdot \langle \alpha', \mathbf{z}(\alpha'; \sigma) \rangle) = -1\}$ ,  $\alpha' \in \mathcal{A}$ ).*

In light of this, one realizes that a novel approach is required for the problem of learning to classify against strategic agents. Our approach is based on the fact that for such settings, despite the fact that the learner’s loss function might not belong to any specific function family, we can still infer the loss of the learner for some actions against the best-responses of the agents for these actions, i.e.,  $\ell(\alpha, \mathbf{r}_t(\alpha))$ . We stress here that inferring the loss  $\ell(\alpha, \mathbf{r}_t(\alpha))$  is a very easy task. However, it can turn out to be rather useless, due to the strong incompatibility between the external and the Stackelberg regret that we proved (Informal Theorem 1.1).

**Inferring  $\ell(\alpha, \mathbf{r}_t(\alpha))$  Without Observing  $\mathbf{x}_t$ .** Our results are enabled by thinking of the learner’s and the agent’s action spaces as dual ones. Let us focus on the agent’s action space first. Because the utility function is of the form  $u_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) = \delta \cdot v_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t)) - c_t(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t))$ , where  $v_t(\cdot) \in$

$[0, 1]$ , and the agents are individually rational (i.e.,  $u_t(\cdot) \geq 0$ ), they satisfy a nice  $\delta$ -boundedness property, i.e., given feature vector  $\mathbf{x}_t$  the agent can only misreport within the ball of radius  $\delta$  centered around  $\mathbf{x}_t$ , pictorially shown in purple for  $d = 2$  in Figure 1 and henceforth denoted by  $\mathcal{B}_\delta(\mathbf{x}_t)$ . Since the learner observes  $\mathbf{r}_t(\alpha)$  and knows that the agent would misreport in a ball of radius  $\delta$  around  $\mathbf{x}_t$  (which remains unknown to the learner), the learner knows that, in the worst case, the agent's  $\mathbf{x}_t$  is found somewhere within the ball of radius  $\delta$  centered at  $\mathbf{r}_t(\alpha)$ . Note that since  $\mathbf{r}_t(\alpha)$  can be at the boundary of  $\mathcal{B}_\delta(\mathbf{x}_t)$ , the set of all of the agent's possible misreports against any committed action  $\alpha'$  from the learner ( $\mathbf{r}_t(\alpha')$ ) is the augmented  $2\delta$  ball denoted in green in Figure 1. This means that, since  $y_t$  is also observed by the learner, she could infer her loss  $\ell(\alpha', \mathbf{r}_t(\alpha'))$  for *any* action  $\alpha'$  that has  $\mathcal{B}_{2\delta}(\mathbf{r}_t(\alpha))$  *fully* in an open halfspace (for example, actions  $\beta, \gamma$  in Figure 1).

In the learner's action space, where actions  $\alpha, \beta, \gamma$  are just multidimensional *points*, the aforementioned observation has a nice mathematical translation. An action  $\gamma$  has  $\mathcal{B}_{2\delta}(\mathbf{r}_t(\alpha))$  fully in one of its open halfspaces, if its distance from  $\mathbf{r}_t(\alpha)$  is more than  $2\delta$ . Alternatively, for all actions  $\alpha'$  such that:

$$\frac{|\langle \alpha', \mathbf{r}_t(\alpha) \rangle|}{\|\alpha'\|_2} \leq 2\delta \Leftrightarrow |\langle \alpha', \mathbf{r}_t(\alpha) \rangle| \leq 4\sqrt{d}\delta$$

where the last inequality comes from the fact that  $\mathcal{A} \subseteq [-1, 1]^{d+1}$  and that the  $(d + 1)$ -th coordinate corresponds to the intercept, the learner should *not* try to infer the loss  $\ell(\alpha', \mathbf{r}_t(\alpha'))$ . But for all other actions  $\gamma$  in  $\mathcal{A}$ , the learner can compute her loss *precisely*, due to the fact that she always knows that  $\mathbf{r}_t(\gamma)$  lies somewhere within  $\mathcal{B}_{2\delta}(\mathbf{r}_t(\alpha))$ . From that, we derive that the learner can discretize her action space into the following *polytopes*: upper polytopes  $\mathcal{P}_t^u$ , containing actions  $\mathbf{w} \in \mathcal{A}$  such that  $\langle \mathbf{w}, \mathbf{r}_t(\alpha) \rangle \geq 4\sqrt{d}\delta$  and lower polytopes  $\mathcal{P}_t^l$ , containing actions  $\mathbf{w}' \in \mathcal{A}$  such that  $\langle \mathbf{w}', \mathbf{r}_t(\alpha) \rangle \leq -4\sqrt{d}\delta$ . The distinction into the two sets is helpful as one of them will always assign label  $+1$  to the agent's best-response, and the other will always assign label  $-1$ .

Our GRINDER algorithm uses the above as follows. It maintains a sequence of nested polytopes at all timesteps, and adaptively keeps adding new polytopes to the learner's action space (i.e., "grinding" it) according to the responses received from the agents. Our main result is the following Stackelberg regret bound for GRINDER.

**Theorem 1.3** (Informal). GRINDER *incurs Stackelberg regret*:

$$\mathcal{R}(T) \leq \mathcal{O} \left( \sqrt{\max_{t \in [T]} \left\{ 4 \log \left( 4 \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l| T \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m) \right\}} \cdot \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \cdot T \right)$$

where by  $\lambda(A)$  we denote the Lebesgue measure of a measurable space  $A$ , by  $\underline{p}$  the smallest polytope created in the learner's space and by  $\mathcal{P}_{t,\sigma_t}^u, \mathcal{P}_{t,\sigma_t}^l$  the learner's upper and lower polytopes sets induced if the learner knew  $\sigma_t$ .

This upper bound can be relaxed to  $\mathcal{R}(T) \leq \mathcal{O} \left( \sqrt{\log \left( T \cdot \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \cdot \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \cdot T} \right)$ , but it still depends heavily on quantity  $\log(\lambda(\mathcal{A})/\lambda(\underline{p}))$ . This dependence is rather unsatisfactory, as there are

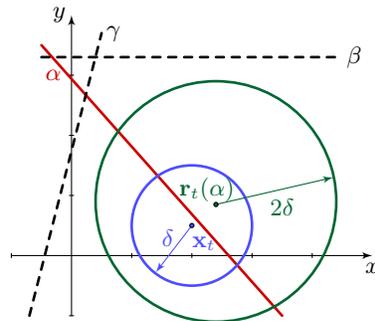


Figure 1: Agent's action space:  $\alpha, \beta, \gamma$  are the learner's actions,  $\mathbf{x}_t$  is the agent's true feature vector and  $\mathbf{r}_t(\alpha)$  his misreport against  $\alpha$ .

sequences of labeled datapoints,  $\{\sigma_t\}_{t=1}^T$ , where  $\lambda(\underline{p})$  becomes so small compared to  $\lambda(\mathcal{A})$ , that the upper bound on the Stackelberg regret of GRINDER becomes vacuous. That said, we prove that this dependence is (up to an extent) unavoidable.

**Theorem 1.4** (Informal). *The Stackelberg regret of learning a linear classifier against myopically rational  $\delta$ -bounded strategic agents is lower bounded by  $\Omega\left(\sqrt{T \log\left(\frac{\lambda(\mathcal{A})}{\lambda(\underline{p})}\right)}\right)$ .*

As a result, the GRINDER algorithm’s regret bound is nearly matching the lower bound. Before we move on to the formal presentation of our results, we clarify the following: GRINDER assumes access to an *in-oracle*, i.e., an oracle that computes the probability that each action  $\alpha \in \mathcal{A}$  will be updated by another action  $\alpha' \in \mathcal{A}$ . To relax this assumption, we provide extensive simulation results showing that our regret guarantees carry over for approximation oracles as well.

### 1.3 Notation Summary

For clarity purposes, in Table 1 we list the notation used in our model.

Variable	Description
$d \in \mathbb{N}$	dimension of the problem
$\mathcal{A} \subseteq [-1, 1]^{d+1}$	learner’s action space
$\alpha_t \in \mathcal{A}$	learner’s committed action for timestep $t$
$\mathcal{X} \subseteq ([0, 1]^d, 1)$	agent’s feature vector space
$h^* : \mathcal{X} \rightarrow \{-1, 1\}$	ideal classifier (not necessarily belonging in $\mathcal{A}$ )
$\mathbf{x}_t \in \mathcal{X}$	agent’s feature vector, <i>as chosen by the environment</i>
$\sigma_t = (\mathbf{x}_t, y_t), y_t = h^*(\mathbf{x}_t)$	agent’s labeled datapoint, <i>as chosen by the environment</i>
$\mathbf{z}_t(\alpha_t; \sigma_t) \in \mathcal{X}$	agent’s <i>reported</i> feature vector
$\hat{y}_t \in \{-1, 1\}$	$\mathbf{z}_t(\alpha_t; \sigma_t)$ ’s label, according to $h^*$
$\ell(\alpha_t, \mathbf{z}_t(\alpha_t; \sigma_t))$	learner’s loss for $\alpha_t$ against report $\mathbf{z}_t(\alpha_t; \sigma_t)$
$u_t(\alpha_t, \mathbf{z}_t(\alpha_t, \sigma_t))$	agent’s utility for reporting $\mathbf{z}_t(\alpha_t, \sigma_t)$ , when learner commits to $\alpha_t$
$\mathbf{r}_t(\alpha_t; \sigma_t)$	agent’s reported feature vector, when he is myopically rational <sup>6</sup>
$R(T)$	learner’s <i>external</i> regret after $T$ timesteps
$\mathcal{R}(T)$	learner’s <i>Stackelberg</i> regret after $T$ timesteps
$\lambda(A)$	Lebesgue measure of measurable space $A$

Table 1: Model Notation Summary

## 2 Preliminaries

In this section, we study the relationship between two of the most common notions of regret used to evaluate online learning in strategic settings: external and Stackelberg regret. The difference between these two is that the benchmark of the loss of the best fixed action in hindsight changes according to the model of the agent’s response. For completeness, we include the definitions of the external and the Stackelberg regret in the context of repeated Stackelberg games, and then we

proceed with the statement of our results. For what follows, let  $\{\alpha_t\}_{t=1}^T$  be the sequence of actions chosen by the learner in a repeated Stackelberg game and  $\mathcal{A}$  the allowable action set. The proofs of this section can be found in Appendix A.1.

**Definition 2.1** (External Regret).  $R(T) = \sum_{t=1}^T \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \min_{\alpha_E^* \in \mathcal{A}} \sum_{t=1}^T \ell(\alpha_E^*, \mathbf{r}_t(\alpha_E^*))$ .

The external regret is the most standard regret notion in online learning. In repeated Stackelberg games, it compares the cumulative loss incurred from  $\{\alpha_t\}_{t \in [T]}$  to the cumulative loss incurred from the best fixed action in hindsight, had you *not* given the opportunity to the agents to best-respond.

**Definition 2.2** (Stackelberg Regret).  $\mathcal{R}(T) = \sum_{t=1}^T \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \min_{\alpha^* \in \mathcal{A}} \sum_{t=1}^T \ell(\alpha^*, \mathbf{r}_t(\alpha^*))$ .

Stackelberg regret [6, 28] is the regret notion that has been previously analyzed in the contexts of strategic classification and Stackelberg Security Games. It compares the cumulative loss incurred from  $\{\alpha_t\}_{t \in [T]}$  to the cumulative loss incurred from the best fixed action in hindsight, had you given the opportunity to the agents to best-respond.

**Theorem 2.3** (Incompatibility of External and Stackelberg Regret). *There exists a repeated strategic classification setting between a learner and a myopically rational  $\delta$ -bounded agent, such that every action sequence with sublinear external regret incurs linear Stackelberg regret, and every action sequence with sublinear Stackelberg regret incurs linear external regret.*

Despite these worst-case incompatibility results, we show in Appendix A.2 that there are some families of repeated Stackelberg games, which we term *Pure Stackelberg Games*, where there does exist a hierarchy between these regret notions. The challenge, however, rests on the fact that most meaningful repeated Stackelberg games (e.g., strategic classification, Stackelberg Security Games) lie in the space *between* the worst-case incompatibility instances and Pure Stackelberg Games. In fact, as we explain in Appendix A.3, unless some assumptions are imposed upon the response functions of the agents, the learner’s loss function is *not even* Lipschitz.

### 3 The GRINDER Algorithm

In this section, we present and analyze the GRINDER Algorithm.

**Warm-Up: Discrete and Countable Action Set.** For the ease of exposition, we begin with a thought experiment: assume that the learner’s action space,  $\mathcal{A}$ , was a countable action set. Then, combining known tools from the literature in online learning with feedback graphs [1], we observe that the learner could at each time step build a strongly observable feedback graph as follows. As per the definition of feedback graphs for online learning, each action  $i \in \mathcal{A}$  corresponds to one node in the graph, and there exists a directed edge  $(i, j)$ ,  $i, j \in \mathcal{A}$  when by playing action  $i$ , the learner can get information about the loss incurred by action  $j$  as well. So, in our case, for each action  $\alpha \in \mathcal{A}$ , all actions  $\alpha' \in \mathcal{A}$  with distance more than  $2\delta$  from  $\mathbf{r}_t(\alpha)$  will be part of the *out* neighborhood of  $\alpha$ ,  $N^{out}(\alpha)$ . In an ideal scenario with perfect knowledge of  $\mathbf{x}_t$ , the learner would want to place in  $N^{out}(\alpha)$  only the actions whose distance from  $\mathbf{x}_t$  is at least  $\delta$ , but in the absence of knowledge of  $\mathbf{x}_t$ , updating actions whose distance from  $\mathbf{r}_t(\alpha)$  is at least  $2\delta$  is a *conservative*<sup>7</sup> estimate of the

<sup>7</sup>This is crucial in the worst-case, as any  $\mathcal{B}_\varepsilon(\mathbf{x}'_t)$  for  $\varepsilon < 2\delta$  and  $\mathbf{x}'_t \neq \mathbf{x}_t$  might create constant bias, thus resulting in linear Stackelberg regret.

$N^{out}(\alpha)$ . More often than not, such a countable action set is not provided to the learner. Further, the function  $\ell(\alpha, \mathbf{r}_t(\alpha))$  is *not* even Lipschitz – therefore no standard discretization technique would work, and no black-box external regret minimizing algorithm could be safely applied. GRINDER provides a solution to these problems by adaptively discretizing the learner’s action space, according to the agent’s responses.

**Algorithm Overview.** At a high level, at each round  $t$  GRINDER maintains a sequence of nested *polytopes*  $\mathcal{P}_t$  of the form  $[x, y]^{d+1}$  for  $-1 \leq x \leq y \leq 1$ . We start by setting  $\mathcal{P}_1 = [-1, 1]^{d+1}$ . At round  $t$ , GRINDER chooses an action for the learner randomly according to a two-stage sampling process. First, the learner draws a polytope  $p \in \mathcal{P}_t$  with probability  $\pi_t(p)$  and then, she draws an action  $\alpha_t \sim \text{Unif}(p)$  uniformly over the polytope  $p$ . We denote the resulting distribution by  $\mathcal{D}_t$ , and by  $\Pr_{\mathcal{D}_t}$  and  $f_{\mathcal{D}_t}(\alpha)$  the associated probability and probability density function<sup>8</sup> respectively.

After the learner observes  $\mathbf{r}_t(\alpha_t)$ , the refinement of her action space is done as follows. She computes two hyperplanes:  $\beta_t^u(\alpha_t)$  and  $\beta_t^l(\alpha_t)$ ,<sup>9</sup> such that  $\beta_t^u(\alpha_t) : \forall \mathbf{w} \in \mathcal{A}, \langle \mathbf{w}, \mathbf{r}_t(\alpha_t) \rangle = 4\sqrt{d}\delta$  and  $\beta_t^l(\alpha_t) : \forall \mathbf{w} \in \mathcal{A}, \langle \mathbf{w}, \mathbf{r}_t(\alpha_t) \rangle = -4\sqrt{d}\delta$ , as we explained in Section 1.2. We call hyperplanes  $\beta_t^u(\alpha_t)$  and  $\beta_t^l(\alpha_t)$  the *boundary* hyperplanes, since they define the polytope boundaries of the learner’s action space.

They also split the learner’s action space into three *regions*; one for which  $\forall \mathbf{w} : \langle \mathbf{w}, \mathbf{r}_t(\alpha_t) \rangle \geq 4\sqrt{d}\delta$ , one for which  $\forall \mathbf{w} : \langle \mathbf{w}, \mathbf{r}_t(\alpha_t) \rangle \leq -4\sqrt{d}\delta$  and one “in the middle”, where  $\forall \mathbf{w} : -4\sqrt{d}\delta \leq \langle \mathbf{w}, \mathbf{r}_t(\alpha_t) \rangle \leq 4\sqrt{d}\delta$ . Figure 2 shows a depiction of the initial discretization of  $[-1, 1]^2$ .

Let  $H^+(\beta), H^-(\beta)$  denote the closed positive and negative halfspaces defined by hyperplane  $\beta$  for intercept  $4\sqrt{d}\delta$  and  $-4\sqrt{d}\delta$  respectively, i.e.,  $\alpha \in H^+(\beta)$  if  $\alpha \in \mathcal{A}, \langle \beta, \alpha \rangle \geq 4\sqrt{d}\delta$  and similarly,  $\alpha \in H^-(\beta)$  if  $\alpha \in \mathcal{A}, \langle \beta, \alpha \rangle \leq -4\sqrt{d}\delta$ . Slightly abusing notation, we say that  $p \in H^+(\beta)$  if  $\forall \alpha \in p$  it holds that  $\alpha \in H^+(\beta)$ .

**Definition 3.1** (Upper & Lower Polytopes Set). *We define the sets  $\mathcal{P}_t^u(\alpha_t) = \{p \in \mathcal{P}_t, p \in H^+(\beta_t^u(\alpha_t))\}$  and  $\mathcal{P}_t^l(\alpha_t) = \{p \in \mathcal{P}_t, p \in H^-(\beta_t^l(\alpha_t))\}$ , as action  $\alpha_t$ ’s upper and lower polytopes set respectively.*

The usefulness of sets  $\mathcal{P}_t^u(\alpha_t)$  and  $\mathcal{P}_t^l(\alpha_t)$  is that we will be able to safely update the loss for actions contained in  $\mathcal{P}_t^u(\alpha_t) \cup \mathcal{P}_t^l(\alpha_t)$ : for all the actions contained in them, due to the agent’s  $\delta$ -boundedness, and since  $\alpha \in \mathcal{P}_t^u(\alpha_t) \cup \mathcal{P}_t^l(\alpha_t) : |\langle \alpha, \mathbf{r}_t(\alpha_t) \rangle| \geq 4\sqrt{d}\delta$ , the loss of the learner boils down to:  $\ell(\alpha, \mathbf{r}_t(\alpha)) = \mathbb{1}\{y_t = -1\} \mathbb{1}\{\alpha \in \mathcal{P}_t^u(\alpha_t)\} + \mathbb{1}\{y_t = 1\} \mathbb{1}\{\alpha \in \mathcal{P}_t^l(\alpha_t)\}$ , without requiring direct observation of  $\mathbf{x}_t$ . Despite the fact that our algorithm does not require the computation of  $\mathcal{P}_t^u(\alpha), \mathcal{P}_t^l(\alpha), \forall \alpha \neq \alpha_t \in \mathcal{A}$ , we do require access to what we call an *In-Oracle*, defined formally below. While access to such an omnipotent oracle might seem far-fetched, we provide simulations in Section 5, where we relax this requirement to an easily-computable approximate in-oracle, with positive results.

<sup>8</sup>Since actions are not distinct, but are instead drawn from a continuous probability distribution, we will use the probability density function in order to construct the unbiased estimates of the actions found at distance more than  $2\delta$  from  $\mathbf{r}_t(\alpha_t)$ .

<sup>9</sup>The subscript  $t$  is used to denote that the definition of these hyperplanes depends not only on the learner’s action  $\alpha_t$ , but also, on the agent’s report,  $\mathbf{r}_t(\alpha_t)$ .

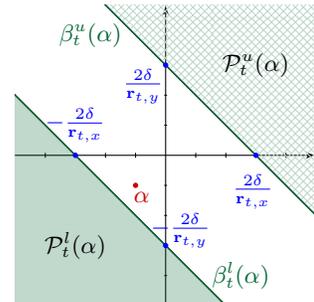


Figure 2: Polytope partitioning for  $d = 2$  in the learner’s action space.

**Definition 3.2** (In-Oracle). We define the In-Oracle as a black-box algorithm, which takes as input an action (resp. a polytope) and returns the total in-probability for this action (resp. polytope):

$$\frac{in}{Pr_{\mathcal{D}_t}}[\alpha] = \int_{\mathcal{A}} Pr_{\mathcal{D}_t} \left[ \left\{ \alpha \in H^+ (\beta_t^u (\alpha')) \right\} \cup \left\{ \alpha \in H^- (\beta_t^l (\alpha')) \right\} \cup \{ \alpha' = \alpha \} \right] d\alpha'$$

and

$$\frac{in}{Pr_{\mathcal{D}_t}}[p] = \int_{\mathcal{A}} Pr_{\mathcal{D}_t} \left[ \left\{ p \subseteq H^+ (\beta_t^u (\alpha')) \right\} \cup \left\{ p \subseteq H^- (\beta_t^l (\alpha')) \right\} \right] d\alpha'$$

The in-probability will be needed in defining an unbiased loss via importance re-weighting. We are now ready to formally define the GRINDER Algorithm. In what follows, we refer to  $dq_t(\alpha)$  as the probability density function at point  $\alpha$ , and to  $q_t(B)$  as the cumulative distribution function for a polytope  $B$ .

---

**Algorithm 2** GRINDER Algorithm for Strategic Classification

---

- 1: Let  $dq_1(\alpha) = 1/\lambda(\mathcal{A}), \forall \alpha \in \mathcal{A}, \mathcal{P} = [-1, 1]^{d+1}, w_1(p) = \lambda(p), p \in \mathcal{P}$  and  $\eta, \gamma$  to be specified.
  - 2: **for**  $t = 1, \dots, T$  **do**
  - 3:   Compute  $\forall p \in \mathcal{P}_t : \pi_t(p) = (1 - \gamma)q_t(p) + \gamma \frac{\lambda(p)}{\lambda(\mathcal{A})}$ .
  - 4:   Select polytope  $p_t \sim \pi_t$  and then, draw action  $\alpha_t \sim \text{Unif}(p_t)$  and commit to it.
  - 5:   Observe the agent's response  $(\mathbf{r}_t(\alpha_t), y_t)$ .
  - 6:   Define set of new polytopes  $\mathcal{P}_t = \mathcal{P}_{t+1}^u(\alpha_t) \cup \mathcal{P}_{t+1}^m(\alpha_t) \cup \mathcal{P}_{t+1}^l(\alpha_t)$ , where:
    - $\mathcal{P}_{t+1}^u(\alpha_t) = \{p' | p' \neq \emptyset, p' = p \cap H^+(\beta_t^u(\alpha_t)), p \in \mathcal{P}_t\}$
    - $\mathcal{P}_{t+1}^l(\alpha_t) = \{p' | p' \neq \emptyset, p' = p \cap H^-(\beta_t^l(\alpha_t)), p \in \mathcal{P}_t \setminus \mathcal{P}_{t+1}^u(\alpha_t)\}$
    - $\mathcal{P}_{t+1}^m(\alpha_t) = \{p' | p' \neq \emptyset, p \in \mathcal{P}_t \setminus (\mathcal{P}_{t+1}^u(\alpha_t) \cup \mathcal{P}_{t+1}^l(\alpha_t))\}$
  - 7:   Compute  $\hat{\ell}(\alpha_t, \mathbf{r}_t(\alpha_t)) = \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) / Pr_{\mathcal{D}_t}^{in}[\alpha_t]$ .
  - 8:   **for**  $p \in \mathcal{P}_t$  **do**
  - 9:     **if**  $p \subseteq H^+(\beta_t^u(\alpha_t))$  or  $p \subseteq H^-(\beta_t^l(\alpha_t))$  **then**
  - 10:       Compute  $\hat{\ell}(p, \mathbf{r}_t(p)) = \ell(p, \mathbf{r}_t(p)) / Pr_{\mathcal{D}_t}^{in}[p]$ .
  - 11:    $w_{t+1}(p) = \lambda(p) \exp\left(-\eta \sum_{\tau=1}^t \hat{\ell}(p, \mathbf{r}_t(p))\right), q_{t+1}(p) = w_{t+1}(p) / \sum_{p' \in \mathcal{P}_{t+1}} w_{t+1}(p')$ .
- 

**Weights Update in Step 11.** The polytopes are defined in such a way that at each timestep the estimated loss within each polytope is *constant*. Not only that, but also, if a polytope has *not* been further “grinded” by the algorithm, then the estimated loss that was used to update the polytope has been the same within the actions of the polytope for each time step! This observation explains the way the weights of the polytopes are updated. Essentially, the learner updates the  $dq_t(\alpha)$  and  $dw_t(\alpha)$  of all the actions in the space  $\mathcal{A}$  as follows

$$dw_{t+1}(\alpha) = \exp\left(-\eta \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))\right) dw_t(\alpha)$$

and

$$dq_{t+1}(\alpha) = \frac{dw_{t+1}(\alpha)}{\int_{\alpha' \in \mathcal{A}} \exp\left(-\eta \hat{\ell}(\alpha', \mathbf{r}_t(\alpha'))\right) dw_t(\alpha')}$$

In other words, instead of updating the *probability* of each point, we directly update the *probability density function* induced at this point. Using the above, and since within each polytope the estimated loss of *all the included actions* is the same, we can update the weight of the polytope directly. More formally, for all the polytopes *apart from the point-polytopes*, it holds that:

$$\begin{aligned} w_{t+1}(p) &= \int_{\alpha \in p} w_t(\alpha) \exp\left(-\eta \sum_{\tau=1}^{t-1} \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))\right) d\alpha = \exp\left(-\eta \sum_{\tau=1}^{t-1} \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))\right) \int_{\alpha \in p} w_t(\alpha) d\alpha \\ &= \lambda(p) w_t(p) \exp\left(-\eta \sum_{\tau=1}^{t-1} \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))\right) \end{aligned}$$

For polytopes including only one action their weight gets updated as  $p = \alpha : w_{t+1}(p) = 0$ , but  $dw_{t+1}(p) = \exp(-\eta \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))) dw_t(\alpha)$  and  $dq_{t+1}(\alpha) = \frac{dw_{t+1}(\alpha)}{\int_{\alpha' \in \mathcal{A}} \exp(-\eta \hat{\ell}(\alpha', \mathbf{r}_t(\alpha'))) dw_t(\alpha')}$ . We are now ready to state our main result.

**Theorem 3.3.** *Given a finite time horizon  $T$ , the Stackelberg Regret induced by the GRINDER Algorithm bounds as:*

$$\mathcal{R}(T) \leq \mathcal{O} \left( \sqrt{\max_{t \in [T]} \left\{ 4 \log \left( 4 \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l| T \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m) \right\}} \cdot \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \cdot T \right)$$

where by  $|S|$  we denote the cardinality of a set  $S$ , by  $\underline{p}$  the polytope with the smallest Lebesgue measure after  $t$  timesteps and by  $\mathcal{P}_{t,\sigma_t}^u, \mathcal{P}_{t,\sigma_t}^l$  the polytope sets including actions in signed distance more than  $2\delta$  away from  $\mathbf{x}_t$  and  $\mathcal{P}_{t,\sigma_t}^m = \mathcal{P}_t \setminus \mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l$ .

The full proof can be found in Appendix B, but below we include a sketch. We also note that the algorithm could be turned into one that does not assume knowledge of  $T$  or  $\lambda(\underline{p})$  by using the standard *doubling trick*.

*Proof Sketch.* We first show that the loss estimator  $\hat{\ell}(\alpha, \mathbf{r}_t(\alpha))$  is an *unbiased* estimator of the true loss for each action  $\ell(\alpha, \mathbf{r}_t(\alpha))$ . Subsequently, we show that its second moment for an action  $\alpha$  is upper bounded by the term  $1/\text{Pr}^{in}[\alpha]$ . For the purposes of the analysis, we then define three families of polytope sets as follows. Polytopes including actions that are in *signed distance* more than  $2\delta$  away from  $\mathbf{x}_t$  belong in the *upper  $\sigma_t$ -induced* polytope set,  $\mathcal{P}_{t,\sigma_t}^u$ . Polytopes including actions that are in *signed distance* less than  $-2\delta$  from  $\mathbf{x}_t$  belong in the *lower  $\sigma_t$ -induced* polytope set,  $\mathcal{P}_{t,\sigma_t}^l$ . The rest of the polytopes belong in the *middle  $\sigma_t$ -induced* polytope set,  $\mathcal{P}_{t,\sigma_t}^m$ . Actions that belong in  $\mathcal{P}_{t,\sigma_t}^m$  are actions for which the agent could potentially misreport and fool the learner, so each of these actions can safely update only itself. This creates the dependency of our regret to the Lebesgue measure of  $\mathcal{P}_{t,\sigma_t}^m$ . The next argument makes a novel connection with a graph theoretic lemma, used by the literature in online learning with feedback graphs. Observe that each of the actions in  $\mathcal{P}_{t,\sigma_t}^u, \mathcal{P}_{t,\sigma_t}^l$  gets updated with probability 1 by any other action in the sets  $\mathcal{P}_{t,\sigma_t}^u, \mathcal{P}_{t,\sigma_t}^l$ . This is because for any of the actions in  $\mathcal{P}_{t,\sigma_t}^u, \mathcal{P}_{t,\sigma_t}^l$ , the agent could not have possibly misreported<sup>10</sup>. So, for all actions  $\alpha \in \mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l$  we have that:  $\text{Pr}^{in}[\alpha] \geq \sum_{p \in \mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l} \pi_t(p)$ . As a result, we can instead think about the set of *polytopes* that belong in  $\mathcal{P}_{t,\sigma_t}^u$  and  $\mathcal{P}_{t,\sigma_t}^l$  as forming a fully connected

<sup>10</sup>Since they are outside of his  $\delta$ -bounded region, and the agent is myopically rational.

feedback graph. The latter, coupled with the fact that our exploration term makes sure that each polytope  $p$  is chosen with probability at least  $\lambda(p)/\lambda(\mathcal{A})$  gives us the result. We note here that an effort of a straightforward application of the graph theoretic lemma on  $\mathcal{A}$ , rather than  $\mathcal{P}_t$ , gives vacuous regret upper bounds, due to the logarithmic dependence in the number of nodes of the feedback graph, which is infinite for the case of  $\mathcal{A}$ .  $\blacktriangle$

In fact, using the fact that  $\lambda(\mathcal{A})/\lambda(\underline{p})$  corresponds to an *upper* bound on the number of polytopes that generally exist (i.e., it is an upper bound also for  $|\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l|$ ), we get the following corollary for the Stackelberg regret.

**Corollary 3.4.** *The Stackelberg Regret induced by the GRINDER Algorithm bounds as:*

$$\mathcal{R}(T) \leq \mathcal{O} \left( \sqrt{\log \left( T \cdot \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \cdot \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \cdot T} \right)$$

Finally, we remark here that all the *data-dependent* quantities of the regret's upper bound crucially depend on  $\delta$ . We have chosen to omit this from the notation for clarity of exposition.

## 4 Lower Bound

The Stackelberg regret bound presented in Section 3 has an inverse dependence in the Lebesgue measure of the smallest polytope, which is rather unsatisfactory; indeed, if this polytope becomes sufficiently small, the upper bound becomes vacuous. In this section, we show that to an extent, this dependence is unavoidable, by presenting a lower bound on the Stackelberg regret of learning a linear classifier against myopically rational  $\delta$ -bounded strategic agents. The proofs of this section can be found in Appendix C.

**Theorem 4.1.** *For any strategy and any  $\delta$ , there exists a sequence of  $\{(\mathbf{x}_t, y_t)\}_{t=1}^T$  such that  $\underline{p}$  is the smallest polytope induced by  $\delta$  and*

$$\mathbb{E} \left[ \sum_{t \in [T]} \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) \right] - \min_{\alpha^* \in \mathcal{A}} \mathbb{E} \left[ \sum_{t \in [T]} \ell(\alpha^*, \mathbf{r}_t(\alpha^*)) \right] \geq \frac{1}{9\sqrt{2}} \sqrt{T \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right)} \quad (1)$$

In the above, the expectation is taken with respect to the randomness of the environment. The lower bound on the expected regret also gives us a worst-case lower bound on Stackelberg Regret  $\mathcal{R}(T)$ :  $\mathcal{R}(T) \geq \frac{1}{9\sqrt{2}} \sqrt{T \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right)}$ .

*Proof Sketch.* Fix a  $\delta > 0$ , and assume that the agents that the learner will encounter  $\forall t \in [T]$  are going to be *truthful* (i.e.,  $\mathbf{r}_t(\alpha) = \mathbf{x}_t, \forall t \in [T], \forall \alpha \in \mathcal{A}$ ). Faithful to our model, however, the learner can only observe  $\mathbf{r}_t(\alpha)$ , without knowing its equivalence to  $\mathbf{x}_t$ . Observe that since settings where agents decide to be truthful are only a *subset* of the different behavioral models for which our GRINDER algorithm applies, the lower bound will hold for the more general behavioral models, as well. We prove the theorem in two steps.

In the first step (Lemma C.1) we show a more relaxed lower bound of order  $\Omega(\sqrt{T})$ . To prove this, we fix a particular feature vector  $\mathbf{x}$  for the agent, and two different adversarial environments

(call them  $U$  and  $L$ ) choosing the label of  $\mathbf{x}$  according to different Bernoulli probability distributions; one of them favors label  $y_t = +1$ , while the other favors label  $y_t = -1$ . Our  $\Omega(\sqrt{T})$  lower bound corresponds to the regret accrued by the learner in order to distinguish between the two. A useful property for the second step is that we choose the adversarial environments in a way that allows us to reason about the polytope holding the best-fixed action in hindsight; for adversarial environment  $U$  all actions in distance  $2\delta$  that favor label  $+1$  are optimal, while for  $L$  the opposite is true.

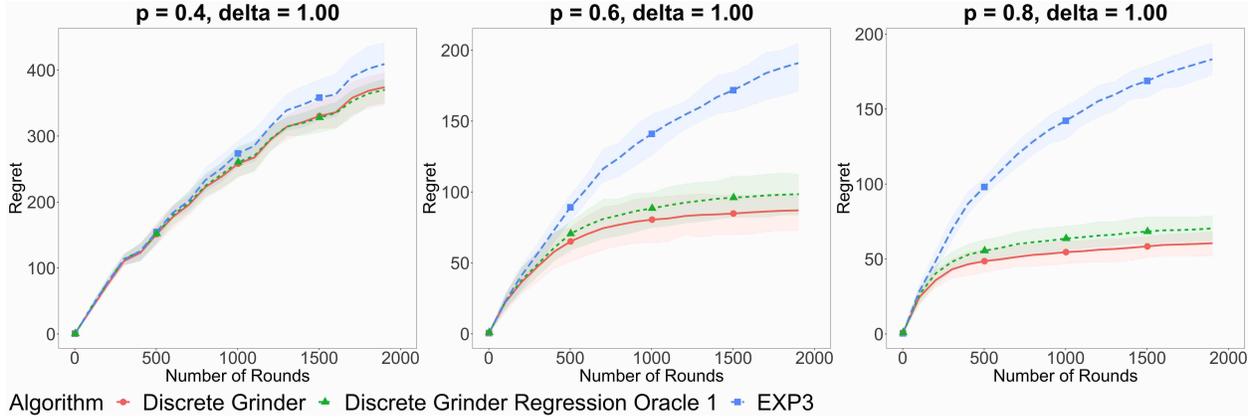
For the second step, we separate the  $T$  timesteps of our finite horizon into  $\Phi = \log(\lambda(\mathcal{A})/\lambda(p))$  phases of  $T/\Phi$  consecutive timesteps each. By dividing the time horizon into  $\Phi$  phases our goal is to show a setting where *any* learner incurs regret of at least  $\sqrt{T/162\Phi}$  at each phase by Lemma C.1. In order to achieve that, we need to make sure (through the construction of the adversarial environments that we encounter at each phase and the choice of  $\mathbf{x}_t$ 's) that there always exists a fixed action that would be optimal for all previous phases. The rest of the proof is the construction of such an instance.  $\blacktriangle$

## 5 Simulations

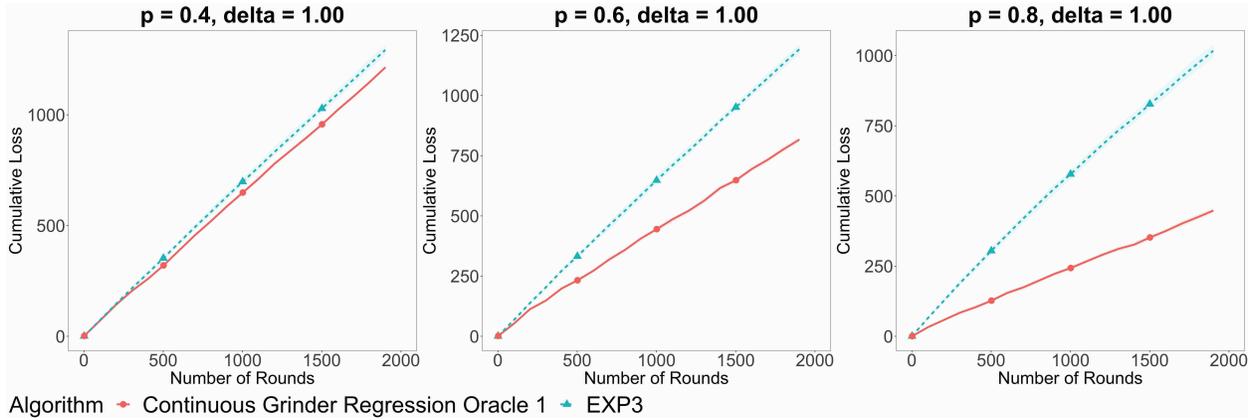
In this section, we present our experimental results. Since all real, currently available datasets for classification are *not* collected taking the strategic considerations of the agents into account, they *cannot* be used for evaluating our algorithm. Indeed, one cannot be certain whether the currently reported feature vectors stem from an altered original feature vector or not. In order to evaluate empirically GRINDER's performance against other algorithms, one needs to know the original feature vectors  $\mathbf{x}_t$ . In Appendix D, we include extra simulations, along with an extended discussion both on the setups, and the findings.

**Setup.** We fix the dimension  $d = 2$ , but the code can be generalized to higher dimensions as well. To make our model more realistic, and drawing intuition from Dong et al. [28], we assume that we do not just face strategic agents (called *spammers* with  $y_t = -1, \mathbf{x}_t \sim \text{Unif}[0.0, 0.6]^2$ ), but also, with probability  $p$ , agents that do not wish to try and fool us (called *non-spammers* with  $y_t = 1, \mathbf{x}_t \sim \text{Unif}[0.4, 1]^2$ ). A careful reader will notice that in this case, our theoretical analysis remains unchanged, except for the part that computes the variance of the estimator. We also tested 3 different values for  $\delta : 0.35, 0.7, 1$ . Figure 3 shows our results for  $\delta = 1$  and we present the results for  $\delta = 0.35, 0.7$  in Appendix D. In the first set of simulations, we assumed a *predefined* action space of size 100 that both GRINDER and EXP3 use. In the second one, GRINDER is run on a fully continuous interval, and EXP3 on a fixed discretization of that interval. Each instance was run for  $T = 2000$  timesteps, with 30 repetitions for each timestep. The solid plots correspond to the *empirical mean* of the regret in the predefined action space simulations, and to the *cumulative loss* in the continuous space ones, while the opaque bands correspond to the 10th and the 90th percentile.

Unless otherwise specified, GRINDER is run using an approximation oracle, which uses past spammer data to train a logistic regression model for each  $\Pr^{in}[\alpha]$ . For the case of the predefined action set, we can compute  $\Pr^{in}[\alpha]$  precisely, but this is no longer the case in the continuous action set. We present the results for the predefined action set as a way to measure experimentally the performance of our approximation oracle compared to the precise one. In Appendix D, we explain thoroughly how the logistic regression is run, and we also present simulation results for another approximation oracle. In summary, we see that both with the use of a precise oracle and with the



(a) GRINDER vs. EXP3 for a predefined action set.



(b) GRINDER vs. EXP3 for a continuous action set.

Figure 3: Regret Performance of GRINDER vs EXP3 for  $\delta = 1$ ; in all cases, GRINDER converges faster.

use of an approximation one for every  $\delta$ , and for every  $p$  our algorithm performs better than EXP3. Interestingly, the approximation oracle follows very closely the performance of the precise one.

For the continuous action space, we implemented *agnostic* versions of both EXP3, and GRINDER. Since it is not generally possible to compute the best-fixed action in hindsight from the infinity of actions that  $[-1, 1]^3$  contains, we resort to depicting the *cumulative losses* of the two algorithms. We use the logistic regression oracle for each of the polytopes again using only spammers' data. As we see from the results in Figure 3b, for the more realistic cases of  $p = 0.6, 0.8$ , GRINDER's regret clearly outperforms that of EXP3. Furthermore, we have just run GRINDER with a relatively simple logistic regression oracle; for more advanced, better oracles, the results would be strengthened. The latter is also the reason that we see that in some cases with a majority of attackers ( $p = 0.4$ ), we perform comparably to EXP3, which, however, is bound to incur linear regret in the worst case, as we have reasoned about in Section 2.

## 6 Further Related Work

In this section we discuss the connections that our work has with the literature around three main areas: learning using data from strategic data sources, Stackelberg games and multi-armed bandits. We also outline some connections with the literature on learning halfspaces with malicious or bounded noise.

**Learning using data from strategic sources.** Recently, there has been an increasing body of work studying machine learning algorithms that are trained on data controlled by *strategic* agents. In this context, the two most studied machine learning tasks for which such algorithms are provided are *classification*, and *linear regression*. For linear regression, most of the works focus on the *offline* setting where the agents hold as their manipulable information their *dependent variables*, and they have either privacy concerns [23, 20], or they wish to make the algorithm predict correctly on their input [40, 24, 21]. There is also a parallel line of work focused on the problem of competing algorithms in the context of strategic linear regression [9, 10]. Both stand in contrast to our model, which is inherently dynamic and thus, online. The situation for the task of classification is similar; most of the existing works are written from the perspective of an *offline* setting [36, 15, 38, 39, 30], with the exception of [28]. Contrary to our setting, in the models used by Meir et al. [38, 39] the *feature vectors* of the agents are assumed to be publicly verifiable, while similarly to our setting in [36, 15, 30, 28] the *feature vectors* are the only manipulable piece of information that the agents possess. Our work, which assumes only knowledge of the general family of utility functions for the agents, differs from [36, 15, 30], as they consider a learner who knows either fully the strategic agents' utility function, or some prior distribution. A byproduct of this is that our solution concept is the one of Stackelberg regret, contrary to the notions of Nash and Stackelberg Equilibria considered in [36, 15, 30].

Another line of research falling under the general umbrella of learning using data from strategic sources that our work is related to, is the research on *learning from revealed preferences* (see e.g., [8, 50, 5, 2, 41, 31, 42, 12]). Abstracting away from the specificities of each model, most of these works are centered around a buyer-seller problem with (potentially) combinatorial preferences, where buyers respond to the sellers' offers by revealing their preferences, rather than their actual value for a choice they made. Our revealed preferences model is centered around the problem of classification, and crucially, presents a lot less structure in the agents' and the learner's utility and loss functions respectively than what normally is assumed in the learning from revealed preferences literature. Finally, there has been recent interest in learning against no-regret learners [13, 25]. Our setting differs from these works in that the agents that we consider are *myopically* rational.

**Stackelberg Games.** Our work (and generally, the learning from revealed preferences problem) is also related to the Stackelberg games literature, which has mostly dealt with Stackelberg Security Games (SSGs). It is nearly impossible for us to survey the vast literature in SSGs, but we refer the interested reader to [47, 45] for overviews. There has also been work in learning theoretic problems related to SSGs ([35, 37, 11, 6]). Out of these, the one most related to our work is the work of Balcan et al. [6], who provide information theoretic sublinear Stackelberg regret<sup>11</sup> algorithms for the learner. In SSGs the utilities of the agents and the learner have nice linearity properties, which are not present in the setting of strategic classification that we consider.

---

<sup>11</sup>Even though the formal definition of Stackelberg regret was introduced by Dong et al. [28].

**Multi-Armed Bandits.** Finally, our work is related to the vast literature in online learning with partial feedback (see [18, 46, 34] for excellent overviews of the field). Contrary to the above, however, our strategic classification setting presents a unique form of feedback, which depends on the best-response of the agents to the learner’s committed actions. For online learning Lipschitz functions in stochastic environments Kleinberg et al. [32] and Bubeck et al. [17] provided adaptive discretization algorithms which take advantage of the specifics of each problem instance, and performed “better” in “good” instances. Our GRINDER algorithm, presents an adaptive discretization algorithm for a problem that is not fully stochastic, as has been the case in the works of Kleinberg et al. [32], Bubeck et al. [17]. Additionally, albeit completely different technically, the naming of our algorithm is an homage to the *Zooming* algorithm of Kleinberg et al. [32]. Further, our work is related to the literature in online learning with feedback graphs [1, 22]. In fact, as we explain in Section 3, if the learner were to be given a predefined set of actions instead of a continuous interval, one could almost straightforwardly apply the feedback graph results and guarantee sublinear Stackelberg regret. However, in our work, we are dealing directly with the infinite set of actions inside a continuous interval, which makes the application of previous results impossible.

Finally, connecting learning from strategic sources and online learning for multi-armed bandits, Braverman et al. [14] studied a model where each arm is controlled by a strategic entity who chooses the amount of the reward that it will pass to the learner. In our model, the arms are not explicitly controlled by anybody.

**Learning Halfspaces with Malicious or Bounded Noise.** Distantly related to our work is the problem of learning halfspaces with malicious or bounded noise (e.g., [33, 4, 27, 26]) in the offline setting. Our work has significant differences from these settings; we do not make *any* distributional assumptions about  $\{\mathbf{x}_t\}_{t=1}^T$ , our agents *cannot* alter their  $y_t$ ’s, and ultimately, they have *agency* about their misreport. What does seem to be a connection between our paper and this line of works can be seen in the lower bound; it is well known that if certain margin assumptions are not satisfied, then it is not possible to learn the linear separator in the offline setting. Our lower bound seems to provide the same interpretation, but for the online, strategic setting; indeed, one can easily create a sequence of  $\{\mathbf{x}_t\}_{t=1}^T$  with almost zero margin, such that  $\lambda(p) \rightarrow 0$ .

## 7 Discussion and Open Questions

In this paper, we have studied online learning against  $\delta$ -bounded, strategic agents in classification settings, for which we provided the GRINDER algorithm. We complemented our theoretical analysis with simulations, showing first the benefits of our algorithm with a precise oracle, and second, that there are approximation oracles that perform comparably well.

There is a number of interesting questions stemming from our work. On a technical level, an interesting research direction is to provide both theoretical results regarding the Stackelberg regret experienced by the learner for the case that she uses *approximation* oracles (e.g., similar to the ones used in the simulations) and experimental ones for different variants of the agents’ best-response oracles. In many natural settings (mostly the ones that are considered in the fairness community, as mentioned by Bechavod et al. [7]), the learner can only observe the labels of *some* of the agents. This partial observability imposes an extra challenge, and cannot be handled by our framework currently. Moreover, our current analysis is based on the fact that the learner knows  $\delta$  a priori, but no studies exist currently about an estimation of  $\delta$ . We believe that this can only be achieved

through experiments with human subjects in as realistic conditions as possible. Finally, we do believe that the agents' real feature vectors are neither fully stochastic, nor fully adversarial. This model formulation is similar to the one considered in the recent line of works in the online learning literature on the best-of-both-worlds [16, 44, 3, 43] regret bounds. We believe it would be very interesting to provide theoretical bounds for the Stackelberg regret in such strategic classification settings.

## References

- [1] Noga Alon, Nicolo Cesa-Bianchi, Ofer Dekel, and Tomer Koren. Online learning with feedback graphs: Beyond bandits. In *JMLR WORKSHOP AND CONFERENCE PROCEEDINGS*, volume 40. Microtome Publishing, 2015.
- [2] Kareem Amin, Rachel Cummings, Lili Dworkin, Michael Kearns, and Aaron Roth. Online learning and profit maximization from revealed preferences. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [3] Peter Auer and Chao-Kai Chiang. An algorithm with nearly optimal pseudo-regret for both stochastic and adversarial bandits. In *Conference on Learning Theory*, pages 116–120, 2016.
- [4] Pranjal Awasthi, Maria-Florina Balcan, Nika Haghtalab, and Ruth Urner. Efficient learning of linear separators under bounded noise. In *Conference on Learning Theory*, pages 167–190, 2015.
- [5] Maria-Florina Balcan, Amit Daniely, Ruta Mehta, Ruth Urner, and Vijay V Vazirani. Learning economic parameters from revealed preferences. In *International Conference on Web and Internet Economics*, pages 338–353. Springer, 2014.
- [6] Maria-Florina Balcan, Avrim Blum, Nika Haghtalab, and Ariel D Procaccia. Commitment without regrets: Online learning in stackelberg security games. In *Proceedings of the sixteenth ACM conference on economics and computation*, pages 61–78. ACM, 2015.
- [7] Yahav Bechavod, Katrina Ligett, Aaron Roth, Bo Waggoner, and Zhiwei Steven Wu. Equal opportunity in online classification with partial feedback. *arXiv preprint arXiv:1902.02242*, 2019.
- [8] Eyal Beigman and Rakesh Vohra. Learning from revealed preference. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, pages 36–42. ACM, 2006.
- [9] Omer Ben-Porat and Moshe Tennenholtz. Best response regression. In *Advances in Neural Information Processing Systems*, pages 1499–1508, 2017.
- [10] Omer Ben-Porat and Moshe Tennenholtz. Competing prediction algorithms. *arXiv preprint arXiv:1806.01703*, 2018.
- [11] Avrim Blum, Nika Haghtalab, and Ariel D Procaccia. Learning optimal commitment to overcome insecurity. In *Advances in Neural Information Processing Systems*, pages 1826–1834, 2014.

- [12] Avrim Blum, Yishay Mansour, and Jamie Morgenstern. Learning what’s going on: Reconstructing preferences and priorities from opaque transactions. *ACM Transactions on Economics and Computation (TEAC)*, 6(3-4):13, 2018.
- [13] Mark Braverman, Jieming Mao, Jon Schneider, and Matt Weinberg. Selling to a no-regret buyer. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 523–538. ACM, 2018.
- [14] Mark Braverman, Jieming Mao, Jon Schneider, and S. Matthew Weinberg. Multi-armed bandit problems with strategic arms. In Alina Beygelzimer and Daniel Hsu, editors, *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 383–416, Phoenix, USA, 25–28 Jun 2019. PMLR.
- [15] Michael Brückner and Tobias Scheffer. Stackelberg games for adversarial prediction problems. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 547–555. ACM, 2011.
- [16] Sébastien Bubeck and Aleksandrs Slivkins. The best of both worlds: stochastic and adversarial bandits. In *Conference on Learning Theory*, pages 42–1, 2012.
- [17] Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. X-armed bandits. *Journal of Machine Learning Research*, 12(May):1655–1695, 2011.
- [18] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.
- [19] Sébastien Bubeck, Yin Tat Lee, and Ronen Eldan. Kernel-based methods for bandit convex optimization. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 72–85. ACM, 2017.
- [20] Yang Cai, Constantinos Daskalakis, and Christos Papadimitriou. Optimum statistical estimation with strategic data sources. In *Conference on Learning Theory*, pages 280–296, 2015.
- [21] Yiling Chen, Chara Podimata, Ariel D Procaccia, and Nisarg Shah. Strategyproof linear regression in high dimensions. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 9–26. ACM, 2018.
- [22] Alon Cohen, Tamir Hazan, and Tomer Koren. Online learning with feedback graphs without the graphs. In *International Conference on Machine Learning*, pages 811–819, 2016.
- [23] Rachel Cummings, Stratis Ioannidis, and Katrina Ligett. Truthful linear regression. In *Conference on Learning Theory*, pages 448–483, 2015.
- [24] Ofer Dekel, Felix Fischer, and Ariel D Procaccia. Incentive compatible regression learning. *Journal of Computer and System Sciences*, 76(8):759–777, 2010.
- [25] Yuan Deng, Jon Schneider, and Balusubramanian Sivan. Strategizing against no-regret learners. *arXiv preprint arXiv:1909.13861*, 2019.

- [26] Ilias Diakonikolas, Themis Gouleakis, and Christos Tzamos. Distribution-independent pac learning of halfspaces with massart noise. *arXiv preprint arXiv:1906.10075*, 2019.
- [27] Ilias Diakonikolas, Daniel M Kane, and Pasin Manurangsi. Nearly tight bounds for robust proper learning of halfspaces with a margin. *arXiv preprint arXiv:1908.11335*, 2019.
- [28] Jinshuo Dong, Aaron Roth, Zachary Schutzman, Bo Waggoner, and Zhiwei Steven Wu. Strategic classification from revealed preferences. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 55–70. ACM, 2018.
- [29] Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 385–394. Society for Industrial and Applied Mathematics, 2005.
- [30] Moritz Hardt, Nimrod Megiddo, Christos Papadimitriou, and Mary Wootters. Strategic classification. In *Proceedings of the 2016 ACM conference on innovations in theoretical computer science*, pages 111–122. ACM, 2016.
- [31] Shahin Jabbari, Ryan M Rogers, Aaron Roth, and Steven Z Wu. Learning from rational behavior: Predicting solutions to unknown linear programs. In *Advances in Neural Information Processing Systems*, pages 1570–1578, 2016.
- [32] Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 681–690. ACM, 2008.
- [33] Adam R Klivans, Philip M Long, and Rocco A Servedio. Learning halfspaces with malicious noise. *Journal of Machine Learning Research*, 10(Dec):2715–2740, 2009.
- [34] Tor Lattimore and Csaba Szepesvári. Bandit algorithms. <https://tor-lattimore.com/downloads/book/book.pdf>, 2019.
- [35] Joshua Letchford, Vincent Conitzer, and Kamesh Munagala. Learning and approximating the optimal strategy to commit to. In *International Symposium on Algorithmic Game Theory*, pages 250–262. Springer, 2009.
- [36] Wei Liu and Sanjay Chawla. A game theoretical model for adversarial learning. In *2009 IEEE International Conference on Data Mining Workshops*, pages 25–30. IEEE, 2009.
- [37] Janusz Marecki, Gerry Tesauro, and Richard Segal. Playing repeated stackelberg games with unknown opponents. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 821–828. International Foundation for Autonomous Agents and Multiagent Systems, 2012.
- [38] Reshef Meir, Shaull Almagor, Assaf Michaely, and Jeffrey S Rosenschein. Tight bounds for strategyproof classification. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 319–326. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

- [39] Reshef Meir, Ariel D Procaccia, and Jeffrey S Rosenschein. Algorithms for strategyproof classification. *Artificial Intelligence*, 186:123–156, 2012.
- [40] Javier Perote and Juan Perote-Pena. Strategy-proof estimators for simple regression. *Mathematical Social Sciences*, 47(2):153–176, 2004.
- [41] Aaron Roth, Jonathan Ullman, and Zhiwei Steven Wu. Watch and learn: Optimizing from revealed preferences feedback. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 949–962. ACM, 2016.
- [42] Aaron Roth, Aleksandrs Slivkins, Jonathan Ullman, and Zhiwei Steven Wu. Multidimensional dynamic pricing for welfare maximization. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 519–536. ACM, 2017.
- [43] Yevgeny Seldin and Gábor Lugosi. An improved parametrization and analysis of the  $\text{exp3++}$  algorithm for stochastic and adversarial bandits. *arXiv preprint arXiv:1702.06103*, 2017.
- [44] Yevgeny Seldin and Aleksandrs Slivkins. One practical algorithm for both stochastic and adversarial bandits. In *ICML*, pages 1287–1295, 2014.
- [45] Arunesh Sinha, Fei Fang, Bo An, Christopher Kiekintveld, and Milind Tambe. Stackelberg security games: Looking beyond a decade of success. In *IJCAI*, pages 5494–5501, 2018.
- [46] Aleksandrs Slivkins. Introduction to multi-armed bandits. *arXiv preprint arXiv:1904.07272*, 2019.
- [47] Milind Tambe. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge university press, 2011.
- [48] The New York Times. The college admissions scandal. <https://www.nytimes.com/news-event/college-admissions-scandal>, 2019. [Online; accessed 30-April-2019].
- [49] Heinrich Von Stackelberg. *Marktform und gleichgewicht*. J. springer, 1934.
- [50] Morteza Zadimoghaddam and Aaron Roth. Efficiently learning from revealed preference. In *International Workshop on Internet and Network Economics*, pages 114–127. Springer, 2012.

## A Supplementary Material for Section 2

### A.1 Missing Proofs from Section 2

*Proof of Theorem 2.3.* Let an action space  $\mathcal{A} = \{\alpha^1, \alpha^2\}$  such that  $\alpha^1 = (1, 1, -1)$  and  $\alpha^2 = (0.5, -1, 0.25)$ , and let  $\delta = 0.1$ . The environment draws feature vectors  $\mathbf{x}^1 = (0.4, 0.5)$ ,  $\mathbf{x}^2 = (0.6, 0.6)$ ,  $\mathbf{x}^3 = (0.65, 0.3)$ ,  $\mathbf{x}^4 = (0.8, 1)$  with probabilities 0.05, 0.15, 0.05, 0.75 respectively, and with labels  $-1, -1, +1, +1$ . For clarity, Figure 4 provides a pictorial depiction of the example, along with the best responses of the agents for each action. We first analyze the loss incurred by each action, according to the best-responses of the agents and the probabilities with which the feature vectors are chosen by the environment.

- $(\mathbb{E}[\ell(\alpha^1, \mathbf{r}_t(\alpha^1))])$  When the learner plays  $\alpha^1$  against agent's responses  $\mathbf{r}_t(\alpha^1)$ , she makes a mistake in her prediction every time that the environment drew  $\mathbf{x}_1$  or  $\mathbf{x}_2$  for the agent, i.e.,  $\mathbb{E}[\ell(\alpha^1, \mathbf{r}_t(\alpha^1))] = 0.2$ .
- $(\mathbb{E}[\ell(\alpha^2, \mathbf{r}_t(\alpha^2))])$  When the learner plays  $\alpha^2$  against agent's responses  $\mathbf{r}_t(\alpha^2)$ , she makes a mistake in her prediction every time that the environment drew  $\mathbf{x}_1$  or  $\mathbf{x}_2$  or  $\mathbf{x}_3$  for the agent, i.e.,  $\mathbb{E}[\ell(\alpha^2, \mathbf{r}_t(\alpha^2))] = 0.25$ .
- $(\mathbb{E}[\ell(\alpha^1, \mathbf{r}_t(\alpha^2))])$  When the learner plays  $\alpha^1$  against agent's responses  $\mathbf{r}_t(\alpha^2)$ , she makes a mistake in her prediction every time that the environment drew  $\mathbf{x}_2$  or  $\mathbf{x}_4$  for the agent, i.e.,  $\mathbb{E}[\ell(\alpha^1, \mathbf{r}_t(\alpha^2))] = 0.9$ .
- $(\mathbb{E}[\ell(\alpha^2, \mathbf{r}_t(\alpha^1))])$  When the learner plays  $\alpha^2$  against agent's responses  $\mathbf{r}_t(\alpha^1)$ , she makes a mistake in her prediction every time that the environment drew  $\mathbf{x}_3$  for the agent, i.e.,  $\mathbb{E}[\ell(\alpha^2, \mathbf{r}_t(\alpha^1))] = 0.05$ .

We now prove that *any* sequence with sublinear Stackelberg regret will have linear external regret. Observe that for the Stackelberg regret, the best fixed action in hindsight is action  $\alpha^1$ , with cumulative loss  $0.2T$ . Therefore, any action sequence that yields sublinear Stackelberg regret must have cumulative loss  $0.2T + o(T)$ , meaning that action  $\alpha^2$  is played at most  $o(T)$  times, while action  $\alpha^1$  is played at least  $T - o(T)$  times. Given this, we proceed by identifying the best fixed action for the *external* regret in any action such sequence  $\{\alpha_t\}_{t=1}^T$ . For that, we compute the loss that any of the actions in  $\mathcal{A}$  would incur, had they been the fixed action for sequence  $\{\alpha_t\}_{t=1}^T$ .

Assume that action  $\alpha^1$  was the fixed action in hindsight for the sequence  $\{\alpha_t\}_{t=1}^T$ . Then, the cumulative loss incurred by playing  $\alpha^1$  constantly throughout the  $T$  rounds, denoted by  $\sum_{t=1}^T \ell(\alpha^1, \mathbf{r}_t(\alpha_t))$  is:

$$\underbrace{0.2(T - o(T))}_{\text{loss incurred when playing } \alpha^1 \text{ against } \mathbf{r}_t(\alpha^1)} + \underbrace{0.9o(T)}_{\text{loss incurred when playing } \alpha^1 \text{ against } \mathbf{r}_t(\alpha^2)}$$

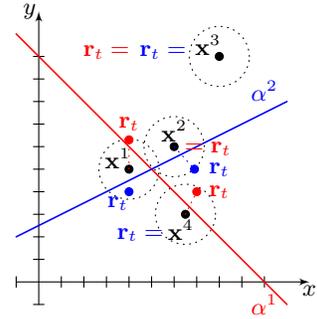


Figure 4: Incompatibility between external-Stackelberg regret example. Black dots denote true feature vectors. Dotted circles correspond to the  $\delta$ -bounded interval inside which agents can misreport. Blue dots correspond to misreports against action  $\alpha_2$  and red dots to misreports against action  $\alpha_1$ .

Assume that action  $\alpha^2$  was the fixed action in hindsight for the aforementioned action sequence. Then, the cumulative loss incurred by playing  $\alpha^2$ , denoted by  $\sum_{t=1}^T \ell(\alpha^2, \mathbf{r}_t(\alpha_t))$  is equal to

$$\underbrace{0.05(T - o(T))}_{\text{loss incurred when playing } \alpha^2 \text{ against } \mathbf{r}_t(\alpha^1)} + \underbrace{0.25o(T)}_{\text{loss incurred when playing } \alpha^2 \text{ against } \mathbf{r}_t(\alpha^2)}$$

Hence, we have that the best fixed action in hindsight for the external regret for the sequence  $\{\alpha_t\}_{t=1}^T$  is action  $\alpha^2$ . This means, however, that for the sequence  $\{\alpha_t\}_{t=1}^T$ , which guaranteed sublinear Stackelberg regret, the external regret is *linear* in  $T$ :

$$R(T) \geq 0.2T - 0.05T \geq 0.15T$$

Moving forward, we prove that *any* sequence with sublinear external regret will have linear Stackelberg regret. Since we previously proved that any action sequence  $\{\alpha_t\}_{t=1}^T$  with sublinear Stackelberg regret plays at least  $T - o(T)$  times action  $\alpha^1$  and this resulted in having linear external regret, we only need to consider sequences where action  $\alpha^2$  is played  $T - o(T)$  times, while action  $\alpha^1$  is played for  $o(T)$  times. For any such action sequence, it suffices to show that the external regret yielded will be sublinear, since for any such sequence the Stackelberg regret will be linear:

$$\mathcal{R}(T) = 0.2o(T) + 0.25 \cdot (T - o(T)) - 0.2T \geq 0.05T$$

Similarly to the analysis above, we distinguish the following cases. Assume that action  $\alpha^1$  was the fixed action in hindsight for  $\{\alpha_t\}_{t=1}^T$ . Then, the cumulative loss incurred by playing  $\alpha^1$  is  $\sum_{t=1}^T \ell(\alpha^1, \mathbf{r}_t(\alpha_t)) = 0.2o(T) + 0.9(T - o(T))$ . Assume that action  $\alpha^2$  was the fixed action in hindsight for the aforementioned action sequence. Then, the cumulative loss incurred by playing  $\alpha^2$  is  $\sum_{t=1}^T \ell(\alpha^2, \mathbf{r}_t(\alpha_t)) = 0.05o(T) + 0.25(T - o(T))$ . As a result, the best fixed action in hindsight for the Stackelberg regret would be action  $\alpha^2$ , yielding external regret  $o(T)$ , i.e., sublinear. This concludes our proof.  $\blacktriangle$

## A.2 Purely Adversarial and Cooperative Stackelberg Games

Despite the worst-case incompatibility results that we have shown for the notions of external and Stackelberg regret, there are families of repeated games for which there is a clear *hierarchy* between the two. In this subsection, we will study two of the most important ones; the family of *Purely Adversarial*, and the family of *Purely Cooperative Stackelberg Games*.

**Definition A.1** (Purely Adversarial Stackelberg Game (PASGs)). *We will call a Stackelberg Game, where a learner commits to an action  $\alpha \in \mathcal{A}$  and an agent best-responds to it  $\mathbf{r}(\alpha)$ , Purely Adversarial, if for all actions  $\alpha' \in \mathcal{A}$  for the loss of the learner it holds that:  $\ell(\alpha, \mathbf{r}(\alpha)) \geq \ell(\alpha, \mathbf{r}(\alpha'))$ , i.e., the agent inflicts the higher loss to the learner, when best-responding to the action to which she committed.*

**Definition A.2** (Purely Cooperative Stackelberg Game (PCSGs)). *We will call a Stackelberg Game, where a learner commits to an action  $\alpha \in \mathcal{A}$  and an agent best-responds to it  $\mathbf{r}(\alpha)$ , Purely Cooperative if for all actions  $\alpha' \in \mathcal{A}$  for the loss of the learner it holds that:  $\ell(\alpha, \mathbf{r}(\alpha)) \leq \ell(\alpha, \mathbf{r}(\alpha'))$ , i.e., the agent inflicts the lowest loss to the learner, when best-responding to the action to which she committed.*

We remark here that despite their similarities, PASGs and PCSGs are *not* equivalent to zero-sum games; in fact, it is easy to see that every zero-sum game is either a PASG or a PCSG, but the converse is *not* true (see e.g., the example loss matrix given in Table 2 where the first coordinate of tuple  $(i, j)$  corresponds to the loss of the learner, and the second to the loss of the agent). Next, we outline the hierarchy between external and Stackelberg regret in repeated PASGs and PCSGs.

	$\mathbf{r}_t(\alpha)$	$\mathbf{r}_t(\alpha')$
$\alpha$	(7, -1)	(6, -3)
$\alpha'$	(6, -3)	(7, -1)

Table 2: Example of a PASG that is not zero-sum.

**Lemma A.3.** *In repeated PASGs, Stackelberg regret is upper bounded by external regret, i.e.,  $\mathcal{R}(T) \leq R(T)$ . In other words, any no-Stackelberg regret sequence of actions is also a no-external regret one.*

*Proof.* Let  $\tilde{\alpha} = \arg \min_{\alpha \in \mathcal{A}} \sum_{t=1}^T \ell(\alpha, \mathbf{r}_t(\alpha_t))$  and  $\alpha^* = \arg \min_{\alpha \in \mathcal{A}} \sum_{t=1}^T \ell(\alpha, \mathbf{r}_t(\alpha))$ . Then:

$$\begin{aligned}
R(T) &= \sum_{t=1}^T \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \sum_{t=1}^T \ell(\tilde{\alpha}, \mathbf{r}_t(\alpha_t)) && \text{(definition of external regret)} \\
&\geq \sum_{t=1}^T \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \sum_{t=1}^T \ell(\alpha^*, \mathbf{r}_t(\alpha_t)) && \text{(definition of } \tilde{\alpha}) \\
&\geq \sum_{t=1}^T \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \sum_{t=1}^T \ell(\alpha^*, \mathbf{r}_t(\alpha^*)) && (\ell(\alpha^*, \mathbf{r}_t(\alpha_t)) \leq \ell(\alpha^*, \mathbf{r}_t(\alpha^*))) \\
&= \mathcal{R}(T)
\end{aligned}$$

▲

On the other hand, for PCSGs it holds that:

**Lemma A.4.** *In repeated PCSGs, Stackelberg regret is lower bounded by external regret, i.e.,  $\mathcal{R}(T) \geq R(T)$ . In other words, any no-Stackelberg regret sequence of actions is also a no-external regret one.*

*Proof.* Let  $\tilde{\alpha} = \arg \min_{\alpha \in \mathcal{A}} \sum_{t=1}^T \ell(\alpha, \mathbf{r}_t(\alpha_t))$  and  $\alpha^* = \arg \min_{\alpha \in \mathcal{A}} \sum_{t=1}^T \ell(\alpha, \mathbf{r}_t(\alpha))$ . Then:

$$\begin{aligned}
R(T) &= \sum_{t=1}^T \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \sum_{t=1}^T \ell(\tilde{\alpha}, \mathbf{r}_t(\alpha_t)) && \text{(definition of external regret)} \\
&\leq \sum_{t=1}^T \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \sum_{t=1}^T \ell(\tilde{\alpha}, \mathbf{r}_t(\tilde{\alpha})) && \text{(definition of PCSGs)} \\
&\leq \sum_{t=1}^T \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \sum_{t=1}^T \ell(\alpha^*, \mathbf{r}_t(\alpha^*)) && \text{(definition of } \alpha^*) \\
&= \mathcal{R}(T)
\end{aligned}$$

▲

### A.3 The Function $\ell(\alpha, \mathbf{r}_t(\alpha))$

In this subsection we will show that even in the case where the learner's loss function is Lipschitz with respect to both its first and second argument, this does not guarantee that we will have a Lipschitz loss function, if we give the agent the opportunity to best-respond. The following Lemma formalizes an argument first made by Balcan et al. [6].

**Lemma A.5.** *Let  $\ell(x, y)$  denote the learner's loss function in a Stackelberg game, such that  $\ell$  is  $L_1$ -Lipschitz with respect to the first argument, and  $L_2$ -Lipschitz with respect to the second. Then, for the learner's loss between any two actions  $\alpha, \alpha' \in \mathcal{A}$  it holds that:*

$$|\ell(\alpha, \mathbf{r}_t(\alpha)) - \ell(\alpha', \mathbf{r}_t(\alpha'))| \leq \max \{L_1 \cdot \|\alpha' - \alpha\|, L_2 \cdot \|\mathbf{r}_t(\alpha) - \mathbf{r}_t(\alpha')\|\}$$

*Proof.* We distinguish the set of actions  $\mathcal{A}$  into pairs  $(\alpha, \alpha')$  with the following properties:

1. For pair  $(\alpha, \alpha')$  it holds that:  $\ell(\alpha, \mathbf{r}_t(\alpha)) \geq \ell(\alpha, \mathbf{r}_t(\alpha'))$  and  $\ell(\alpha', \mathbf{r}_t(\alpha')) \geq \ell(\alpha', \mathbf{r}_t(\alpha))$ . Observe that, given that  $\ell$  is  $L_1$ -Lipschitz in its first argument, we have that:

$$\ell(\alpha', \mathbf{r}_t(\alpha')) - \ell(\alpha, \mathbf{r}_t(\alpha)) \geq \ell(\alpha', \mathbf{r}_t(\alpha)) - \ell(\alpha, \mathbf{r}_t(\alpha)) \geq -L_1 \|\alpha' - \alpha\|$$

and

$$\ell(\alpha', \mathbf{r}_t(\alpha')) - \ell(\alpha, \mathbf{r}_t(\alpha)) \leq \ell(\alpha', \mathbf{r}_t(\alpha')) - \ell(\alpha, \mathbf{r}_t(\alpha')) \leq L_1 \|\alpha' - \alpha\|$$

Therefore, for such pairs of actions function  $\ell(\alpha, \mathbf{r}_t(\alpha))$  is  $L_1$ -Lipschitz with respect to  $\alpha$ .

2. For pair  $(\alpha, \alpha')$  it holds that:  $\ell(\alpha, \mathbf{r}_t(\alpha)) \leq \ell(\alpha, \mathbf{r}_t(\alpha'))$  and  $\ell(\alpha', \mathbf{r}_t(\alpha')) \leq \ell(\alpha', \mathbf{r}_t(\alpha))$ . Similarly to Case 1, it is easy to see that on these pairs of actions, function  $\ell(\alpha, \mathbf{r}_t(\alpha))$  is again  $L_1$ -Lipschitz with respect to  $\alpha$ .

3. For pair  $(\alpha, \alpha')$  it holds that

$$\ell(\alpha, \mathbf{r}_t(\alpha)) \geq \ell(\alpha, \mathbf{r}_t(\alpha')) \tag{2}$$

and

$$\ell(\alpha', \mathbf{r}_t(\alpha')) \leq \ell(\alpha', \mathbf{r}_t(\alpha)) \tag{3}$$

From Equations (2) and (3) we have that

$$\ell(\alpha', \mathbf{r}_t(\alpha')) - \ell(\alpha, \mathbf{r}_t(\alpha)) \leq L_1 \|\alpha' - \alpha\| \tag{4}$$

We further distinguish the following cases:

- (a)  $\ell(\alpha, \mathbf{r}_t(\alpha)) = \ell(\alpha', \mathbf{r}_t(\alpha'))$ . Clearly,  $|\ell(\alpha, \mathbf{r}_t(\alpha)) - \ell(\alpha', \mathbf{r}_t(\alpha'))| \leq L_1 \cdot \|\alpha' - \alpha\|$  holds.
- (b)  $\ell(\alpha, \mathbf{r}_t(\alpha)) \leq \ell(\alpha', \mathbf{r}_t(\alpha'))$ . From Equation (4), we get:  $|\ell(\alpha, \mathbf{r}_t(\alpha)) - \ell(\alpha', \mathbf{r}_t(\alpha'))| \leq L_1 \cdot \|\alpha' - \alpha\|$ .
- (c)  $\ell(\alpha, \mathbf{r}_t(\alpha)) \geq \ell(\alpha', \mathbf{r}_t(\alpha'))$  Observe now that if  $\ell(\alpha, \mathbf{r}_t(\alpha)) \geq \ell(\alpha', \mathbf{r}_t(\alpha))$ , then from Equation (3) the latter is lower bounded by  $\ell(\alpha', \mathbf{r}_t(\alpha'))$ , which leads to a contradiction. Hence, it has to be the case that  $\ell(\alpha, \mathbf{r}_t(\alpha)) \leq \ell(\alpha', \mathbf{r}_t(\alpha))$ . The latter, combined with the assumption that  $\ell$  is  $L_2$  - Lipschitz with respect to its second argument, implies that  $\ell(\alpha', \mathbf{r}_t(\alpha')) - \ell(\alpha, \mathbf{r}_t(\alpha)) \geq -L_2 \cdot \|\mathbf{r}_t(\alpha') - \mathbf{r}_t(\alpha)\|$ .

4. For the pair  $(\alpha, \alpha')$  it holds that  $\ell(\alpha, \mathbf{r}_t(\alpha)) \leq \ell(\alpha, \mathbf{r}_t(\alpha'))$  and  $\ell(\alpha', \mathbf{r}_t(\alpha')) \geq \ell(\alpha', \mathbf{r}_t(\alpha))$ . The case is analogous to Case 3. ▲

To summarize, in PASGs and PCSGs the loss function written in terms of the action of the agent is *Lipschitz*, i.e.,  $|\ell(\alpha, \mathbf{r}_t(\alpha)) - \ell(\alpha', \mathbf{r}_t(\alpha'))| \leq L_1 \cdot \|\alpha' - \alpha\|$ . However, in General Stackelberg Games one can only guarantee that

$$|\ell(\alpha, \mathbf{r}_t(\alpha)) - \ell(\alpha', \mathbf{r}_t(\alpha'))| \leq \max \{L_1 \cdot \|\alpha' - \alpha\|, L_2 \cdot \|\mathbf{r}_t(\alpha') - \mathbf{r}_t(\alpha)\|\} \quad (5)$$

Despite the fact that the latter means that  $\ell(\alpha, \mathbf{r}_t(\alpha))$  is not generally Lipschitz, there are some Stackelberg settings where  $\|\mathbf{r}_t(\alpha') - \mathbf{r}_t(\alpha)\|$  can be upper bounded by  $\|\alpha' - \alpha\|$  multiplied by a constant. For example, from well known results in convex optimization (for completeness see Lemma A.6), we can see that this is exactly the case in settings where the agent's utility function,  $u_t(\alpha, r)$  is *strongly concave* in  $r$ , and *quasilinear*<sup>12</sup> in  $\alpha$ .

**Lemma A.6** (Closeness of Maxima of Strongly Concave Functions (folklore)). *Let functions  $f : \mathcal{X} \mapsto \mathbb{R}, g : \mathcal{X} \mapsto \mathbb{R}$  be two multidimensional,  $1/\eta_c$ -strongly concave functions with respect to some norm  $\|\cdot\|$ . Let  $h(\mathbf{x}) = f(\mathbf{x}) - g(\mathbf{x}), \mathbf{x} \in \mathcal{X}$  be  $L_{f,g}$ -Lipschitz<sup>13</sup> with respect to the same norm  $\|\cdot\|$ . Then, for the maxima of the two functions:  $\mu_f = \arg \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$  and  $\mu_g = \arg \max_{\mathbf{x} \in \mathcal{X}} g(\mathbf{x})$  it holds that:*

$$\|\mu_f - \mu_g\| \leq L_{f,g} \cdot \eta_c \quad (6)$$

*Proof.* First, we take the Taylor expansion of  $f$  around its maximum,  $\mu_f$  and use the strong concavity condition:

$$\begin{aligned} f(\mathbf{x}) &\leq f(\mu_f) + \langle \nabla f(\mu_f), \mathbf{x} - \mu_f \rangle - \frac{1}{2\eta} \|\mu_f - \mathbf{x}\|^2 && \text{(strong concavity)} \\ &= f(\mu_f) - \frac{1}{2\eta} \|\mu_f - \mathbf{x}\|^2 && (\nabla f(\mu_f) = 0, \text{ since } \mu_f \text{ is the maximum}) \end{aligned}$$

Similarly, by taking the Taylor expansion of  $g$  around its maximum and using the strong concavity condition:

$$g(\mathbf{x}) \leq g(\mu_g) - \frac{1}{2\eta} \|\mu_g - \mathbf{x}\|^2 \quad (7)$$

Using the  $L_{f,g}$ -Lipschitzness of  $h(\mathbf{x})$  we get:

$$\begin{aligned} L_{f,g} \cdot \|\mu_g - \mu_f\| &\geq |h(\mu_g) - h(\mu_f)| \geq h(\mu_g) - h(\mu_f) \\ &\geq f(\mu_g) - f(\mu_f) + g(\mu_f) - g(\mu_g) \\ &\geq \frac{1}{2\eta} \|\mu_f - \mu_g\|^2 + \frac{1}{2\eta} \|\mu_f - \mu_g\|^2 && \text{(from Taylor expansion)} \\ &\geq \frac{1}{\eta} \|\mu_f - \mu_g\|^2 \end{aligned}$$

Dividing both sides with  $\|\mu_g - \mu_f\|$  concludes the proof. ▲

An example of such a utility function in the context of strategic classification (similar to the family of utility functions used in [28]) is presented below.

<sup>12</sup>Quasilinearity in  $\alpha$  establishes that  $L_{f,g}$  which is used by Lemma A.6 will be linear in  $\|\alpha' - \alpha\|$ .

<sup>13</sup>We use the subscript  $f, g$  in the Lipschitzness constant to denote the fact that it depends on the two functions  $f$  and  $g$ .

**Example.** Let  $u_t(\alpha, \mathbf{z}(\alpha; \mathbf{x})) = \langle \alpha, \mathbf{z}(\alpha; \mathbf{x}) \rangle - (\mathbf{x} - \mathbf{z}(\alpha; \mathbf{x}))^2$ . Then, we would like to compute an upper bound on the difference between  $\|\mathbf{r}(\alpha) - \mathbf{r}(\alpha')\|$ , where  $\mathbf{r}(\alpha) = \arg \max_{\mathbf{z} \in \mathcal{X}; \mathbf{x}} u_t(\alpha, \mathbf{z})$  and  $\mathbf{r}(\alpha') = \arg \max_{\mathbf{z} \in \mathcal{X}; \mathbf{x}} u_t(\alpha', \mathbf{z})$ . Following Lemma A.6 we can define functions  $f(\mathbf{z}) = u_t(\alpha, \mathbf{z})$  and  $g(\mathbf{z}) = u_t(\alpha', \mathbf{z})$ . Now, observe that function  $h(\mathbf{z}) = f(\mathbf{z}) - g(\mathbf{z})$  is indeed  $\|\alpha - \alpha'\|$ -Lipschitz (i.e., the Lipschitzness constant depends on the specific actions):

$$|f(\mathbf{y}) - g(\mathbf{y}) - f(\mathbf{z}) + g(\mathbf{z})| = |\langle \alpha - \alpha', \mathbf{y} - \mathbf{z} \rangle| \leq \|\alpha - \alpha'\| \cdot \|\mathbf{y} - \mathbf{z}\|$$

where the last inequality comes from the Cauchy-Schwartz inequality. Furthermore, observe that both  $f(\cdot)$  and  $g(\cdot)$  are  $\frac{1}{2}$ -strongly concave. Therefore, from Lemma A.6 we get that:

$$\|\mathbf{r}(\alpha) - \mathbf{r}(\alpha')\| \leq \frac{\|\alpha - \alpha'\|}{2}$$

## B Supplementary Material for Section 3

The proof of Theorem 3.3 follows from a sequence of lemmas and claims presented below. By convention, we call a single point a *point-polytope*, and we denote the set of all point-polytopes by  $\overline{\mathcal{P}}$ . Our proof uses a lot of notation. For easier reference, we summarize the notation used in our analysis in Table 3.

Variable	Description
$\mathcal{P}_t$	set of active polytopes at timestep $t$
$\overline{\mathcal{P}}_t$	set of active point-polytopes at timestep $t$
$\mathcal{D}_t$	induced distribution from 2-step sampling process
$\Pr_{\mathcal{D}_t}, f_{\mathcal{D}_t}$	cdf and pdf of $\mathcal{D}_t$
$\beta_t^u(\alpha_t) : \langle \mathbf{r}_t(\alpha_t), \mathbf{w} \rangle = 4\sqrt{d}\delta$	upper boundary hyperplane
$\beta_t^l(\alpha_t) : \langle \mathbf{r}_t(\alpha_t), \mathbf{w} \rangle = -4\sqrt{d}\delta$	lower boundary hyperplane
$H^+(\beta_t^u(\alpha))$	$\alpha' \in H^+(\beta_t^u(\alpha))$ , if $\langle \mathbf{r}_t(\alpha), \alpha' \rangle \geq 4\sqrt{d}\delta$
$H^-(\beta_t^l(\alpha))$	$\alpha' \in H^-(\beta_t^l(\alpha))$ , if $\langle \mathbf{r}_t(\alpha), \alpha' \rangle \leq -4\sqrt{d}\delta$
$\mathcal{P}_t^u(\alpha)$	upper polytopes set ( $p \in \mathcal{P}_t : p \in H^+(\beta_t^u(\alpha))$ )
$\mathcal{P}_t^l(\alpha)$	lower polytopes set ( $p \in \mathcal{P}_t : p \in H^-(\beta_t^l(\alpha))$ )
$\mathcal{P}_t^m(\alpha)$	middle polytopes set ( $p \in \mathcal{P}_t : p \notin \mathcal{P}_t \setminus (\mathcal{P}_t^u \cup \mathcal{P}_t^l)$ )
$\Pr_{\mathcal{D}_t}^{in}[\alpha], \Pr_{\mathcal{D}_t}^{in}[p]$	in-probability for $\alpha$ and $p$ (see Definition 3.2)
$\underline{p}$	polytope ( $\notin \overline{\mathcal{P}}_t$ ) with smallest Lebesgue measure at $T$

Table 3: GRINDER Notation Summary

**Claim B.1.** *The two-stage sampling probability distribution  $\mathcal{D}_t$  is equivalent to a one-stage probability distribution of drawing directly an action from density  $d\pi_t(\cdot)$ .*

*Proof.* The one-stage probability distribution that draws an action from  $\pi_t$  is equivalent to choosing an action  $\alpha \in \mathcal{A}$  from probability density function:  $d\pi_t(\alpha) = (1 - \gamma)dq_t(\alpha) + \frac{\gamma}{\lambda(\mathcal{A})}$ . The two-stage probability is:  $d\pi_{\mathcal{D}_t}(\alpha) = \frac{1}{\lambda(p)} \left( (1 - \gamma)q_t(p) + \frac{\gamma\lambda(p)}{\lambda(\mathcal{A})} \right)$ . Since  $q_t(p) = \lambda(p)dq_t(\alpha), \forall \alpha \in p$ , we get the result.  $\blacktriangle$

Moving forward we first analyze the first and the second moment of the loss  $\hat{\ell}(\alpha, \mathbf{r}_t(\alpha))$  for each action  $\alpha$ , based on the induced probability distribution  $\mathcal{D}_t$ , assuming oracle access to  $\text{Pr}_{\mathcal{D}_t}^{\text{in}}[\alpha]$ .

**Lemma B.2** (First Moment). *The estimated loss  $\hat{\ell}(\alpha, \mathbf{r}_t(\alpha))$  is an unbiased estimate of the true loss  $\ell(\alpha, \mathbf{r}_t(\alpha))$ , when actions are drawn from the induced probability distribution,  $\mathcal{D}_t$ .*

*Proof.* For all the actions  $\alpha \in \mathcal{A}$ , given Claim B.1, it holds that:

$$\mathbb{E}_{\alpha_t \sim \mathcal{D}_t} \left[ \hat{\ell}(\alpha, \mathbf{r}_t(\alpha)) \right] = \int_{\mathcal{A}} f_{\mathcal{D}_t}(\alpha') \frac{\ell(\alpha, \mathbf{r}_t(\alpha)) \mathbb{1} \{ \alpha \in N^{\text{out}}(\alpha') \}}{\text{Pr}_{\mathcal{D}_t}^{\text{in}}[\alpha]} d\alpha' = \ell(\alpha, \mathbf{r}_t(\alpha))$$

▲

**Lemma B.3** (Second Moment). *For the second moment of the estimated loss  $\hat{\ell}(\alpha, \mathbf{r}_t(\alpha))$  with respect to the induced probability distribution  $\mathcal{D}_t$  it holds that:*

$$\mathbb{E}_{\alpha_t \sim \mathcal{D}_t} \left[ \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))^2 \right] = \frac{\ell(\alpha, \mathbf{r}_t(\alpha))^2}{\text{Pr}_{\mathcal{D}_t}^{\text{in}}[\alpha]} \leq \frac{1}{\text{Pr}_{\mathcal{D}_t}^{\text{in}}[\alpha]}$$

*Proof.* For all the actions  $\alpha \in \mathcal{A}$ , given Claim B.1, it holds that:

$$\mathbb{E}_{\alpha_t \sim \mathcal{D}_t} \left[ \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))^2 \right] = \int_{\mathcal{A}} f_{\mathcal{D}_t}(\alpha') \frac{\ell(\alpha, \mathbf{r}_t(\alpha))^2 \mathbb{1} \{ \alpha \in N^{\text{out}}(\alpha') \}}{\text{Pr}_{\mathcal{D}_t}^{\text{in}}[\alpha]^2} d\alpha' = \frac{\ell(\alpha, \mathbf{r}_t(\alpha))^2}{\text{Pr}_{\mathcal{D}_t}^{\text{in}}[\alpha]} \leq \frac{1}{\text{Pr}_{\mathcal{D}_t}^{\text{in}}[\alpha]}$$

▲

A technical lemma follows, which will be used in our proof for the Stackelberg regret, in order to bound the variance of the estimated losses computed by the GRINDER Algorithm. Before we proceed to it, we find it useful to define the upper, lower, and middle  $\sigma_t$ -induced polytope sets. These will only be used in the analysis of our algorithm.

**Definition B.4.** *In what follows, we denote by  $\text{sdist}(\alpha, \mathbf{x}_t)$  the signed distance of point  $\mathbf{x}_t$  from hyperplane  $\alpha$ .*

1. We define the middle  $\sigma_t$ -induced polytope set, denoted by  $\mathcal{P}_{t, \sigma_t}^m$ , as the set containing the following polytopes:  $p \in \mathcal{P}_{t, \sigma_t}^m : |\text{sdist}(\alpha, \mathbf{x}_t)| \leq 2\delta, \forall \alpha \in p$ .
2. We define the upper  $\sigma_t$ -induced polytope set, denoted by  $\mathcal{P}_{t, \sigma_t}^u$ , as the set containing the following polytopes:  $p \in \mathcal{P}_{t, \sigma_t}^u : \text{sdist}(\alpha, \mathbf{x}_t) \geq 2\delta, \forall \alpha \in p$ .
3. We define the lower  $\sigma_t$ -induced polytope set, denoted by  $\mathcal{P}_{t, \sigma_t}^l$ , as the set containing the following polytopes:  $p \in \mathcal{P}_{t, \sigma_t}^l : \text{sdist}(\alpha, \mathbf{x}_t) \leq -2\delta, \forall \alpha \in p$ .

**Lemma B.5.**

$$\mathbb{E}_{\alpha_t \sim \mathcal{D}_t} \left[ \frac{1}{\text{Pr}_{\mathcal{D}_t}^{\text{in}}[\alpha_t]} \right] \leq 4 \log \left( \frac{4\lambda(\mathcal{A}) \cdot |\mathcal{P}_{t, \sigma_t}^u \cup \mathcal{P}_{t, \sigma_t}^l|}{\gamma \lambda(\underline{p})} \right) + \lambda(\mathcal{P}_{t, \sigma_t}^m)$$

*Proof.* We first analyze the term:  $\mathbb{E}_{\alpha \sim \mathcal{D}_t} \left[ \frac{1}{\Pr_{\mathcal{D}_t}^{in}[\alpha_t]} \right]$  as follows:

$$\begin{aligned} \mathbb{E}_{\alpha_t \sim \mathcal{D}_t} \left[ \frac{1}{\Pr_{\mathcal{D}_t}^{in}[\alpha_t]} \right] &= \int_{\mathcal{A}} \frac{f_{\mathcal{D}_t}(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[\alpha]} d\alpha \\ &= \int_{\bigcup (\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l)} \frac{f_{\mathcal{D}_t}(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[\alpha]} d\alpha + \int_{\bigcup \mathcal{P}_{t,\sigma_t}^m} \frac{f_{\mathcal{D}_t}(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[\alpha]} d\alpha \end{aligned} \quad (8)$$

where by  $\bigcup \mathcal{P}_{t,\sigma_t}^m$  we denote the integral over all *actions* that belong in some polytope from the set  $\mathcal{P}_{t,\sigma_t}^m$ . In the RHS of Equation (8), the term  $\int_{\bigcup \mathcal{P}_{t,\sigma_t}^m} \frac{f_{\mathcal{D}_t}(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[\alpha]} d\alpha$  is relatively easier to analyze. Due to the conservative estimates of the true middle space (i.e., the actions such that  $\text{sdist}(\alpha, \mathbf{x}_t) \leq \delta$ ), the set of polytopes  $\mathcal{P}_{t,\sigma_t}^m$  contains *all* the actions that actually belong in the  $\sigma_t$ -induced middle space, plus some other actions for which the agent could not have misreported, due to their  $\delta$ -boundedness. Now, for all the actions that actually belong in the  $\sigma_t$ -induced middle space, it holds that they only get information (i.e., get updated) when they are chosen by the algorithm, while for the rest of the actions that have ended up in our middle space, they could have been updated by other actions as well. Thus, it holds that:

$$\forall \alpha \in \bigcup \mathcal{P}_{t,\sigma_t}^m : \Pr_{\mathcal{D}_t}^{in}[\alpha] \geq f_{\mathcal{D}_t}(\alpha)$$

As a result:

$$\int_{\bigcup \mathcal{P}_{t,\sigma_t}^m} \frac{f_{\mathcal{D}_t}(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[\alpha]} d\alpha \leq \int_{\bigcup \mathcal{P}_{t,\sigma_t}^m} \frac{f_{\mathcal{D}_t}(\alpha)}{f_{\mathcal{D}_t}(\alpha)} d\alpha = \lambda(\mathcal{P}_{t,\sigma_t}^m) \quad (9)$$

Moving forward, we turn our attention to term  $\int_{\alpha \in \bigcup (\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l)} \frac{f_{\mathcal{D}_t}(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[\alpha]} d\alpha$ . Assume now that an action  $\alpha$  belongs in a polytope  $p_\alpha$ . Then, there are (weakly) more actions that can potentially update action  $\alpha$ , than the whole polytope in which it belongs,  $p_\alpha$ ; indeed, in order to update the polytope, one must make sure that every action within it is updateable. As a result,  $\Pr_{\mathcal{D}_t}^{in}[\alpha] \geq \Pr_{\mathcal{D}_t}^{in}[p_\alpha]$ . Using this in Equation (8) we get that the first term of the RHS of the variance is upper bounded by:

$$\sum_{p \in \mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l} \int_p \frac{f_{\mathcal{D}_t}(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[p]} d\alpha \quad (10)$$

Further, let  $\Pr_{\mathcal{D}_t}^{in}[p]_{u,l}$  be the part of  $\Pr_{\mathcal{D}_t}^{in}[p]$  that depends only in the updates that stem from actions in either the upper or the lower polytopes sets. As such:  $\Pr_{\mathcal{D}_t}^{in}[p]_{u,l} \leq \Pr_{\mathcal{D}_t}^{in}[p]$  and the term in Equation (10) can be upper bounded by:

$$\sum_{p \in \mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l} \frac{1}{\Pr_{\mathcal{D}_t}^{in}[p]_{u,l}} \int_p f_{\mathcal{D}_t}(\alpha) d\alpha \quad (11)$$

where we have also used the fact that we gain oracle access to quantity  $\Pr_{\mathcal{D}_t}^{in}[p]_{u,l}$  and therefore, we treat it as a constant in the integral. Observe now that the term  $\int_p f_{\mathcal{D}_t}(\alpha) d\alpha$  corresponds to the total probability that the action  $\alpha_t$ , which is chosen from the induced probability distribution  $\mathcal{D}_t$ ,

belongs to polytope  $p$ , i.e., it is equal to  $\pi_t(p)$ . Hence, the term in Equation (11) can be rewritten as:

$$\sum_{p \in \mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l} \frac{\pi_t(p)}{\Pr_{\mathcal{D}_t}^{in}[p]_{u,l}} \quad (12)$$

As we have explained before,  $\pi_t(p) = 0$ , for  $p \in \bar{\mathcal{P}}_t$  and as a result, we can disregard point-polytopes from our consideration for the rest of this proof.

We will now upper bound this term by using a graph-theoretic lemma of Alon et al. [1]. Observe now that all the actions within the  $\sigma_t$ -induced upper and the lower polytopes set form a Feedback Graph as follows: each node corresponds to a polytope from one of the sets  $\mathcal{P}_{t,\sigma_t}^u, \mathcal{P}_{t,\sigma_t}^l$ . So the total number of nodes is at most  $|\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l|$ , where by  $|S|$  we denote the cardinality of a set  $S$ . Each edge  $(i, j)$  will correspond to information passing from node  $i$  to node  $j$ , i.e., the directed edge  $(i, j)$  exists when polytope  $j$  gets updated by just observing the loss for action from the polytope  $i$ . However, for each action belonging in a polytope among the  $\sigma_t$ -induced upper and lower polytopes sets, we know that the agent could not possibly misreport (due to myopic individual rationality), and as a result, all the actions within the upper and the lower polytopes sets would be updated! As a result, the independence number of this feedback graph is  $\alpha^G = 1$ . Using the fact that each polytope  $p$  is chosen with probability at least  $\pi_t(p) \geq \gamma \lambda(p) / \lambda(\mathcal{A}) \geq \gamma \lambda(\underline{p}_t) / \lambda(\mathcal{A})$ , where by  $\lambda(\underline{p}_t)$  we denote the Lebesgue measure of the smallest polytope at timestep  $t$ , the variance is upper bounded by:

$$\mathbb{E}_{\alpha_t \sim \mathcal{D}_t} \left[ \frac{1}{\Pr_{\mathcal{D}_t}^{in}[\alpha_t]} \right] \leq 4 \log \left( \frac{4\lambda(\mathcal{A}) \cdot |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l|}{\lambda(\underline{p}_t) \cdot \gamma} \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m)$$

which concludes our proof. ▲

Let  $\underline{p} = \arg \min_{p \in \mathcal{P}_T \setminus \bar{\mathcal{P}}_T} \lambda(p)$  be the polytope with the smallest Lebesgue measure (excluding point-polytopes) in the finest partition of the learner's action space. Since  $\lambda(\mathcal{A}) / \lambda(\underline{p})$  corresponds to an *upper* bound in the number of polytopes that generally exist, we get the following corollary.

**Corollary B.6.**

$$\mathbb{E}_{\alpha_t \sim \mathcal{D}_t} \left[ \frac{1}{\Pr_{\mathcal{D}_t}^{in}[\alpha_t]} \right] \leq 8 \log \left( \frac{2\lambda(\mathcal{A})}{\gamma \lambda(\underline{p})} \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m)$$

where  $\underline{p} = \arg \min_{p \in \mathcal{P}_T \setminus \bar{\mathcal{P}}_T} \lambda(p)$ .

We remark here that the added benefit of expressing the first term of the previous proof in terms of *polytopes* rather than *actions* is that if we followed the graph-theoretic upper bound derived by Alon et al. [1], we would have a logarithmic term *in the number of nodes* (i.e., the number of actions), which are infinite.

**Lemma B.7** (Second Order Regret Bound). *Let  $q_1, \dots, q_T$  be the probability distribution over the polytopes defined by Algorithm 2 for the estimated losses  $\hat{\ell}(\alpha, \mathbf{r}_t(\alpha)), t \in [T]$ . Then, the second order regret bound induced by GRINDER is:*

$$\sum_{t=1}^T \sum_{p \in \mathcal{P}_{t+1}} q_t(p) \hat{\ell}(p, \mathbf{r}_t(p)) - \sum_{t=1}^T \hat{\ell}(\alpha^*, \mathbf{r}_t(\alpha^*)) \leq \frac{\eta}{2} \sum_{t=1}^T \sum_{p \in \mathcal{P}_{t+1}} q_t(p) \hat{\ell}(p, \mathbf{r}_t(p))^2 + \frac{1}{\eta} \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \quad (13)$$

where  $\underline{p} = \arg \min_{p \in \mathcal{P}_T \setminus \bar{\mathcal{P}}_T} \lambda(p)$ .

*Proof.* Let  $W_t = \sum_{p \in \mathcal{P}_t} w_t(p)$ . We will upper and lower bound the quantity  $Q = \sum_{t=1}^T \log(W_{t+1}/W_t)$ . First, we focus on the lower bound:

$$Q = \sum_{t=1}^T \log \left( \frac{W_{t+1}}{W_t} \right) = \log \left( \frac{W_T}{W_1} \right) \quad (14)$$

Observe now that in  $t = 1$  there only exists one polytope (the whole  $[-1, 1]^{d+1}$  space), with a total weight of  $\lambda(\mathcal{A})$  and a probability of 1. In other words, all the actions within this polytope have the same weight, which is equal to 1 (uniformly weighted). As a result,  $\log W_1 = \log \left( \sum_{p \in \mathcal{P}_1} \int_{\mathcal{A}} 1 d\alpha \right) = \log(\lambda(\mathcal{A}))$ . Let us analyze the term  $\log W_T$  now:

$$\begin{aligned} \log W_T &= \log \left( \sum_{p \in \mathcal{P}_T} w_T(p) \right) = \log \left( \int_{\mathcal{A}} w_T(\alpha) d\alpha \right) \\ &= \log \left( \sum_{p \in \mathcal{P}_T \setminus \bar{\mathcal{P}}_T} \lambda(p) \exp \left( -\eta \sum_{t=1}^T \hat{\ell}(p, \mathbf{r}_t(p)) \right) + \int_{\bigcup \bar{\mathcal{P}}_T} \exp \left( -\eta \sum_{t=1}^T \hat{\ell}(\alpha, \mathbf{r}_t(\alpha)) \right) d\alpha \right) \end{aligned} \quad (15)$$

where the last equality is due to the fact that not further grinded polytopes have maintained the *same* estimated loss,  $\hat{\ell}$ , for *all* their containing points at each timestep  $t$  and we denote by  $\bar{\mathcal{P}}_T$  the set of point-polytopes contained in  $\mathcal{P}_T$ .

Since the horizon  $T$  is finite, set  $\bar{\mathcal{P}}_T$  is essentially a set of points, thus, it has a Lebesgue measure of 0, we have that:

$$\int_{\bigcup \bar{\mathcal{P}}_T} \exp \left( -\eta \sum_{t=1}^T \hat{\ell}(\alpha, \mathbf{r}_t(\alpha)) \right) d\alpha = 0$$

Let  $\alpha^* = \arg \min_{\alpha \in \mathcal{A}} \sum_{t=1}^T \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))$  (i.e., the best fixed action in hindsight among the *all* actions after  $T$  timesteps, irrespective of whether it belongs to  $\bigcup \bar{\mathcal{P}}_T$  or  $\bigcup \mathcal{P}_T \setminus \bar{\mathcal{P}}_T$ ) and  $\underline{p} \in \mathcal{P}_T \setminus \bar{\mathcal{P}}_T$  be the polytope with the smallest Lebesgue measure in  $\mathcal{P}_T \setminus \bar{\mathcal{P}}_T$  (i.e., excluding point-polytopes). Then, the Equation (15) becomes:

$$\log W_T \geq \log \left( \lambda(\underline{p}) \exp \left( -\eta \sum_{t=1}^T \hat{\ell}(\alpha^*, \mathbf{r}_t(\alpha^*)) \right) \right) = \log(\lambda(\underline{p})) - \eta \sum_{t=1}^T \hat{\ell}(\alpha^*, \mathbf{r}_t(\alpha^*)) \quad (16)$$

As a result:

$$Q = \log W_T - \log W_1 \geq \log \left( \frac{\lambda(\underline{p})}{\lambda(\mathcal{A})} \right) - \eta \sum_{t=1}^T \hat{\ell}(\alpha^*, \mathbf{r}_t(\alpha^*)) \geq \log \left( \frac{\lambda(\underline{p})}{\lambda(\mathcal{A})} \right) - \eta \sum_{t=1}^T \hat{\ell}(\alpha^*, \mathbf{r}_t(\alpha^*)) \quad (17)$$

We move on to the upper bound of  $Q$  now. First, we will analyze the quantity  $\log(W_{t+1}/W_t)$ .

$$\begin{aligned}
\log\left(\frac{W_{t+1}}{W_t}\right) &= \log\left(\frac{\int_{\mathcal{A}} w_t(\alpha) \exp\left(-\eta \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))\right) d\alpha}{W_t}\right) \\
&= \log\left(\int_{\mathcal{A}} q_t(\alpha) \exp\left(-\eta \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))\right) d\alpha\right) \\
&\leq \log\left(\int_{\mathcal{A}} q_t(\alpha) \left(1 - \eta \hat{\ell}(\alpha, \mathbf{r}_t(\alpha)) + \frac{\eta^2}{2} \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))^2\right) d\alpha\right) \\
&\hspace{15em} (e^{-x} \leq 1 - x + \frac{x^2}{2}, x \in [0, 1]) \\
&\leq \log\left(1 - \eta \int_{\mathcal{A}} q_t(\alpha) \hat{\ell}(\alpha, \mathbf{r}_t(\alpha)) d\alpha + \frac{\eta^2}{2} \int_{\mathcal{A}} q_t(\alpha) \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))^2 d\alpha\right) \quad (\int_{\mathcal{A}} q_t(\alpha) d\alpha = 1) \\
&\leq -\eta \int_{\mathcal{A}} q_t(\alpha) \hat{\ell}(\alpha, \mathbf{r}_t(\alpha)) d\alpha + \frac{\eta^2}{2} \int_{\mathcal{A}} q_t(\alpha) \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))^2 d\alpha \quad (\log(1-x) \leq x, x \leq 0)
\end{aligned}$$

Summing up for the  $T$  timesteps the latter becomes:

$$\sum_{t=1}^T \log\left(\frac{W_{t+1}}{W_t}\right) \leq -\sum_{t=1}^T \eta \int_{\mathcal{A}} q_t(\alpha) \hat{\ell}(\alpha, \mathbf{r}_t(\alpha)) d\alpha + \sum_{t=1}^T \frac{\eta^2}{2} \int_{\mathcal{A}} q_t(\alpha) \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))^2 d\alpha \quad (18)$$

Combining Equations (17) and (18) we get that:

$$\sum_{t=1}^T \int_{\mathcal{A}} q_t(\alpha) \hat{\ell}(\alpha, \mathbf{r}_t(\alpha)) d\alpha - \sum_{t=1}^T \hat{\ell}(\alpha^*, \mathbf{r}_t(\alpha^*)) \leq \frac{\eta}{2} \sum_{t=1}^T \int_{\mathcal{A}} q_t(\alpha) \hat{\ell}(\alpha, \mathbf{r}_t(\alpha))^2 d\alpha + \frac{1}{\eta} \log\left(\frac{\lambda(\mathcal{A})}{\lambda(\underline{p})}\right)$$

▲

We are now ready for the proof of Theorem 3.3.

*Proof of Theorem 3.3.* By taking the expectation  $\mathbb{E}_{\alpha_t \sim \mathcal{D}_t}$  in Lemma B.7 we get that:

$$\sum_{t=1}^T \int_{\mathcal{A}} q_t(\alpha) \mathbb{E}_{\mathcal{D}_t} [\hat{\ell}(\alpha, \mathbf{r}_t(\alpha))] d\alpha - \sum_{t=1}^T \mathbb{E}_{\mathcal{D}_t} [\hat{\ell}(\alpha^*, \mathbf{r}_t(\alpha^*))] \leq \frac{\eta}{2} \sum_{t=1}^T \int_{\mathcal{A}} q_t(\alpha) \mathbb{E}_{\mathcal{D}_t} [\hat{\ell}(\alpha, \mathbf{r}_t(\alpha))^2] d\alpha + \frac{1}{\eta} \log\left(\frac{\lambda(\mathcal{A})}{\lambda(\underline{p})}\right)$$

Combining Lemmas B.2, B.3 with the latter we get:

$$\begin{aligned}
&\sum_{t=1}^T \int_{\mathcal{A}} q_t(\alpha) \ell(\alpha, \mathbf{r}_t(\alpha)) d\alpha - \sum_{t=1}^T \ell(\alpha^*, \mathbf{r}_t(\alpha^*)) \\
&\leq \sum_{t=1}^T \frac{\eta}{2} \int_{\mathcal{A}} \frac{q_t(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[\alpha]} d\alpha + \frac{1}{\eta} \log\left(\frac{\lambda(\mathcal{A})}{\lambda(\underline{p})}\right) \\
&\leq \sum_{t=1}^T \eta \int_{\mathcal{A}} \frac{\pi_t(\alpha)}{\Pr_{\mathcal{D}_t}^{in}[\alpha]} d\alpha + \frac{1}{\eta} \log\left(\frac{\lambda(\mathcal{A})}{\lambda(\underline{p})}\right) \quad (\gamma \leq \frac{1}{2}) \\
&\leq \sum_{t=1}^T \eta \left(4 \log\left(\frac{4\lambda(\mathcal{A}) |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l|}{\gamma \cdot \lambda(\underline{p}_t)}\right) + \lambda(\mathcal{P}_{t,\sigma_t}^m)\right) + \frac{1}{\eta} \log\left(\frac{\lambda(\mathcal{A})}{\lambda(\underline{p})}\right) \\
&\hspace{15em} (\text{Lemma B.5})
\end{aligned}$$

Using the fact that  $\int_{\mathcal{A}} \pi_t(\alpha) d\alpha \leq \int_{\mathcal{A}} q_t(\alpha) d\alpha + \gamma$ , the latter becomes:

$$\mathcal{R}(T) \leq \gamma T + \eta \sum_{t=1}^T \left( 4 \log \left( \frac{4\lambda(\mathcal{A}) |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l|}{\gamma \cdot \lambda(\underline{p}_t)} \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m) \right) + \frac{1}{\eta} \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right)$$

Setting  $\gamma = \eta$ :

$$\mathcal{R}(T) \leq \eta \sum_{t=1}^T \left( 1 + 4 \log \left( \frac{4\lambda(\mathcal{A}) |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l|}{\eta \cdot \lambda(\underline{p}_t)} \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m) \right) + \frac{1}{\eta} \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right)$$

which can be upper bounded as follows, since  $\eta_t \in (0, 1)$ :

$$\mathcal{R}(T) \leq \sum_{t=1}^T \eta_t \left( 3 \log \left( 4 \frac{\lambda(\mathcal{A}) |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l| t}{\lambda(\underline{p})} \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m) \right) + \frac{1}{\eta_T} \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \quad (19)$$

Then, tuning  $\eta$  in Equation (19) to be

$$\eta = \sqrt{\frac{\log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right)}{T \cdot \left( \log \left( 4 \frac{\lambda(\mathcal{A}) |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l| T}{\lambda(\underline{p})} \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m) \right)}}$$

we get that the Stackelberg regret is upper bounded by:

$$\mathcal{R}(T) \leq \mathcal{O} \left( \sqrt{\max_{t \in [T]} \left\{ 4 \log \left( 4 \frac{\lambda(\mathcal{A}) |\mathcal{P}_{t,\sigma_t}^u \cup \mathcal{P}_{t,\sigma_t}^l| T}{\lambda(\underline{p})} \right) + \lambda(\mathcal{P}_{t,\sigma_t}^m) \right\}} \cdot \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \cdot T \right)$$

▲

## C Supplementary Material for Section 4

**Lemma C.1.** Fix a  $\mathbf{r} = \mathbf{x} = (u)^d$ , where by  $(u)^d$  we denote the  $d$ -dimensional vector with  $u \in [1/4, 3/4]$  in every dimension. There exists a utility model for the agents, and a pair of adversarial environments  $U$  and  $L$  such that  $\mathbf{r}_t(\alpha) = \mathbf{x}_t = \mathbf{x}, \forall \alpha \in \mathcal{A}, \forall t \in [T]$ , and the sequence of  $y_1, \dots, y_T$  is i.i.d. conditional on the choice of the adversary, such that:

$$\max_{\nu \in \{U, L\}} \min_{\alpha^* \in \mathcal{A}} \mathbb{E}_{\nu} \left[ \sum_{t \in [T]} \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) - \sum_{t \in [T]} \ell(\alpha^*, \mathbf{r}_t(\alpha^*)) \right] \geq \frac{1}{9\sqrt{2}} \sqrt{T}$$

*Proof.* We are going to show this for the case where the agents  $\forall t \in [T]$  are *truthful*, i.e., they decide to report  $\mathbf{r}_t(\alpha) = \mathbf{x}_t, \forall \alpha \in \mathcal{A}, \forall t \in [T]$ . Of course, the learner does not know (and cannot infer) that, so fix a  $\delta > 0$  for the  $\delta$ -boundedness of the agents' utility function. We will prove the lemma only for deterministic strategies for the learner. As is customary, the claim for general strategies can be concluded by averaging over the learner's internal randomness and Fubini's theorem.

Fix an  $\varepsilon > 0$ , and a scalar  $u \in [1/4, 3/4]$ , and define the adversarial environments as follows:  $U$  is such that  $y_t = +1$  with probability  $1/2 + \varepsilon$  and  $y_t = -1$  with probability  $1/2 - \varepsilon$ , and  $L$  is such that  $y_t = -1$  with probability  $1/2 + \varepsilon$  and  $y_t = +1$  with probability  $1/2 - \varepsilon$ . This means that under  $U$ , the majority of times the label is  $+1$ , and under  $L$ , the majority of times the label is  $-1$ . As a result, under  $U$ , *any* action  $\alpha$  such that  $\langle \alpha, \mathbf{x} \rangle \geq 2\delta$  is *optimal* and under  $L$ , *any* action  $\alpha$  such that  $\langle \alpha, \mathbf{x} \rangle \leq -2\delta$  is *optimal*.

Take a sequence of actions  $\alpha_1, \dots, \alpha_T$  and let  $T_{\geq\delta}$  denote the *number* of timesteps for which  $\langle \alpha_t, \mathbf{r} \rangle \geq \sqrt{d}\delta$ , and  $T_{\leq-\delta}$  the number of timesteps for which  $\langle \alpha_t, \mathbf{r} \rangle \leq -\sqrt{d}\delta$ . Clearly,  $T_{\leq-\delta} + T_{\geq\delta} \leq T$ . Observe first that

$$\begin{aligned} \mathbb{E}_U[\mathcal{R}(T)] &\geq \mathbb{E}_U[\mathcal{R}(T_{\leq-\delta})] \\ &\geq \sum_{t \in [T_{\leq-\delta}]} \left[ 1 \cdot \left(\frac{1}{2} + \varepsilon\right) - 1 \cdot \left(\frac{1}{2} - \varepsilon\right) \right] \\ &\geq 2\varepsilon \mathbb{E}_U[T_{\leq-\delta}] \end{aligned} \tag{20}$$

where the first inequality is due to the fact that  $\ell(\alpha_t, \mathbf{x}) = 0 = \ell(\alpha_U^*, \mathbf{x}), \forall t \in [T_{\geq\delta}]$  and any optimal action  $\alpha_U^*$  under  $U$  as we reasoned before. The second inequality uses the following two facts: first, that  $\ell(\alpha_U^*, \mathbf{x}) = 1, \forall t \in [T_{\leq-\delta}]$ , i.e., the best fixed action in hindsight when one encounters adversarial environment  $U$  is an action that estimates the label of  $\mathbf{x}$  to be 1. Second, that when playing against environment  $U$ , a learner incurs loss of 1 every time that she predicted the label of  $\mathbf{x}$  to be  $-1$  (which happens in at least all  $T_{\leq-\delta}$  timesteps), and the actual label was 1 (which happens with probability  $1/2 + \varepsilon$ ). Similarly, we also see that

$$\mathbb{E}_L[\mathcal{R}(T)] \geq 2\varepsilon \mathbb{E}_L[T_{\geq\delta}] \tag{21}$$

Let  $\Pr_U, \Pr_L$  the distributions of  $T_{\leq-\delta}, T_{\geq\delta}$  for adversarial environments  $U, L$  respectively, and let  $\Pr_m$  be the distribution of timesteps when  $y_t = +1$  with probability  $1/2$ . From Pinsker's inequality, and denoting by  $\text{KL}(p, q)$  the KL-divergence between distributions  $p, q$ , we have the following:

$$\mathbb{E}_U[T_{\leq-\delta}] \geq \mathbb{E}_m[T_{\leq-\delta}] - T \sqrt{\frac{\text{KL}(\Pr_U, \Pr_m)}{2}} \tag{22}$$

and

$$\mathbb{E}_L[T_{\geq\delta}] \geq \mathbb{E}_m[T_{\geq\delta}] - T \sqrt{\frac{\text{KL}(\Pr_U, \Pr_m)}{2}} \tag{23}$$

Then, from the data processing inequality for the KL-divergence we get:

$$\text{KL}\left(\Pr_U, \Pr_m\right) \leq T \text{KL}\left(\text{Bern}\left(\frac{1}{2} + \varepsilon\right), \text{Bern}\left(\frac{1}{2}\right)\right) \leq 4T\varepsilon^2 \tag{24}$$

and

$$\text{KL}\left(\Pr_L, \Pr_m\right) \leq T \text{KL}\left(\text{Bern}\left(\frac{1}{2} + \varepsilon\right), \text{Bern}\left(\frac{1}{2}\right)\right) \leq 4T\varepsilon^2 \tag{25}$$

Plugging in Equations (24) and (25) in Equations (22) and (23) we get:

$$\mathbb{E}_U[T_{\leq-\delta}] \geq \mathbb{E}_m[T_{\leq-\delta}] - T\varepsilon\sqrt{2T}$$

and

$$\mathbb{E}_L [T_{\geq \delta}] \geq \mathbb{E}_m [T_{\geq \delta}] - T\varepsilon\sqrt{2T}$$

Finally, averaging Equations (20) and (21) and using the latter two Equations we get:

$$\max_{\nu \in \{U, L\}} \mathbb{E} [\mathcal{R}(T)] \geq \frac{\mathbb{E}_U [\mathcal{R}(T)] + \mathbb{E}_L [\mathcal{R}(T)]}{2} \geq \varepsilon \left( T - 2\varepsilon T\sqrt{2T} \right) \quad (26)$$

Tuning  $\varepsilon = \frac{1}{3\sqrt{2T}}$  gives the result. ▲

*Proof of Theorem 4.1.* We focus again on truthful agents, i.e., agents where  $\mathbf{r}_t(\alpha_t) = \mathbf{x}_t, \forall t \in [T]$ . Since this behavior is a subset of the available behaviors for the agents, the result will carry over for any other  $\delta$ -bounded behavioral model. Let us assume without loss of generality that  $2^c = \lambda(\mathcal{A})/\lambda(\underline{p})$  for some  $c$ . Let  $\Phi$ , such that  $\Phi = \log(\lambda(\mathcal{A})/\lambda(\underline{p}))$ , be a set of *phases* created from  $T/\Phi$  consecutive timesteps in  $T$ . We create the following problem instance.

First focus on phase  $\phi = 1$ , and let the feature vector be  $\mathbf{x}_t = \mathbf{u} = (u)^d, u = 1/2$ , and we specify the adversarial environments  $U$  and  $L$  exactly as in Lemma C.1. Then, after  $T/\Phi$  timesteps one of the two adversarial environments must have caused regret of at least  $\sqrt{T/162\Phi}$ . The feature vector for the next phase  $\phi + 1$  will be *orthogonal* to  $1/2^d$  and such that  $|\mathbf{u}_i| = 1/2, \forall i \in [d]$ . The feature vector for phase  $\phi + 2$  will be parallel to feature vector of  $\phi$ , and will have  $|\mathbf{u}_i| = 1/4, \forall i \in [d]$ . We continue with this pattern for the feature vector until phase  $\Phi$ .

By constructing the adversarial environment like that, we can guarantee that at every phase  $\phi$  there exists an action that would have been the best-fixed for *all* previous phases despite which sequence of adversarial environments  $U, L$  occurred. As a result, the regret for all  $\Phi$  phases is equal to the sum of regrets of each phase. Additionally, the Lebesgue measure of the smallest polytope is  $\underline{p}_\phi$  and is a non-increasing function of the phases, even if it is announced to the learner that  $\delta = 0$ . As a result,

$$\mathbb{E} \left[ \sum_{t \in [T]} \ell(\alpha_t, \mathbf{r}_t(\alpha_t)) \right] - \min_{\alpha^* \in \mathcal{A}} \mathbb{E} \left[ \sum_{t \in [T]} \ell(\alpha^*, \mathbf{r}_t(\alpha^*)) \right] \geq \log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right) \frac{1}{9\sqrt{2}} \sqrt{\frac{T}{\log \left( \frac{\lambda(\mathcal{A})}{\lambda(\underline{p})} \right)}} \quad (27)$$

This concludes our proof. ▲

## D Supplementary Material for Section 5

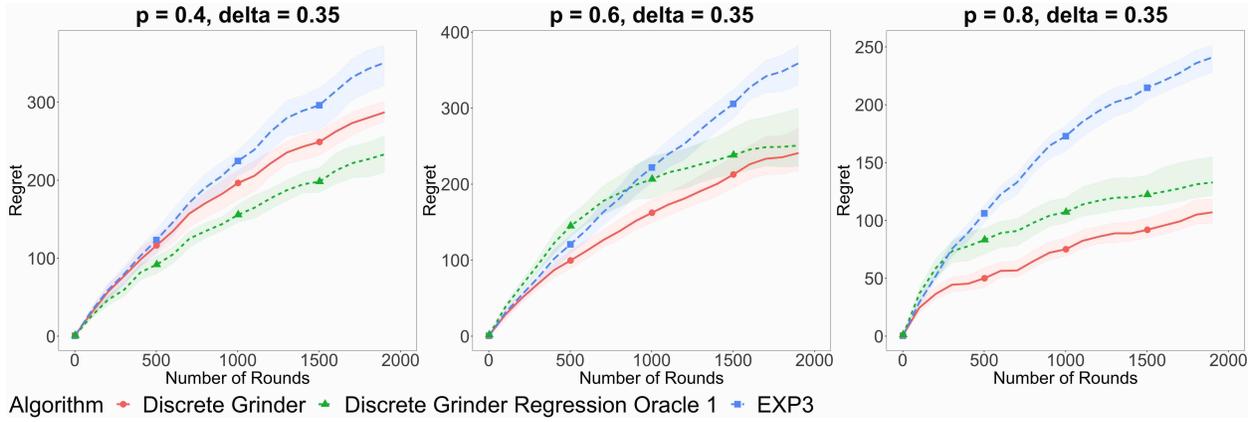
### D.1 Different Values for $\delta$ , Regression Oracle 1

In this subsection, we include the results of our simulations for  $\delta = 0.35$  in Figure 5 and  $\delta = 0.7$  in Figure 6 both for the predefined action set and the continuous one.

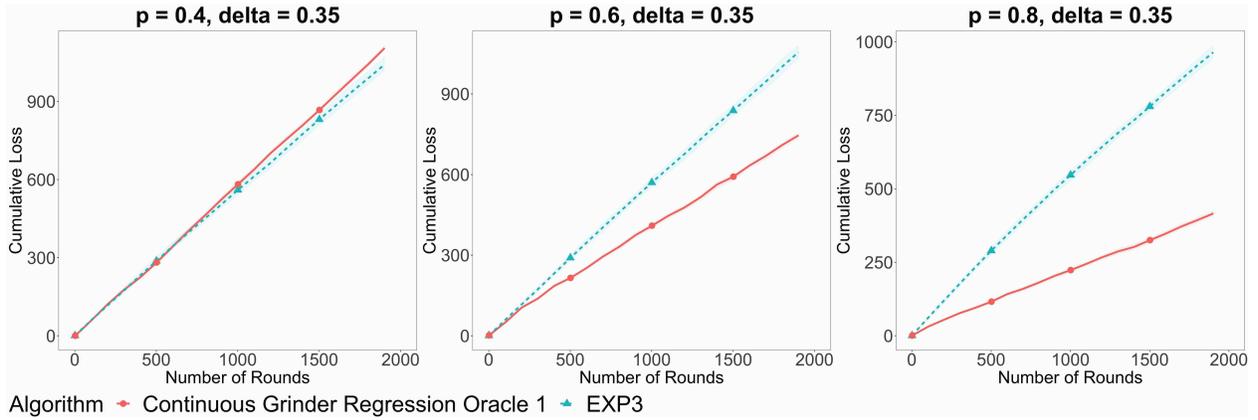
### D.2 Implementing GRINDER for Continuous Action Space

In order to implement GRINDER, we used the `polytope` library<sup>14</sup>, which is part of the TuLiP python package. Other than some rounding-error fixes, we did not intervene with the core methods of the package.

<sup>14</sup><https://github.com/tulip-control/polytope/tree/master/requirements>

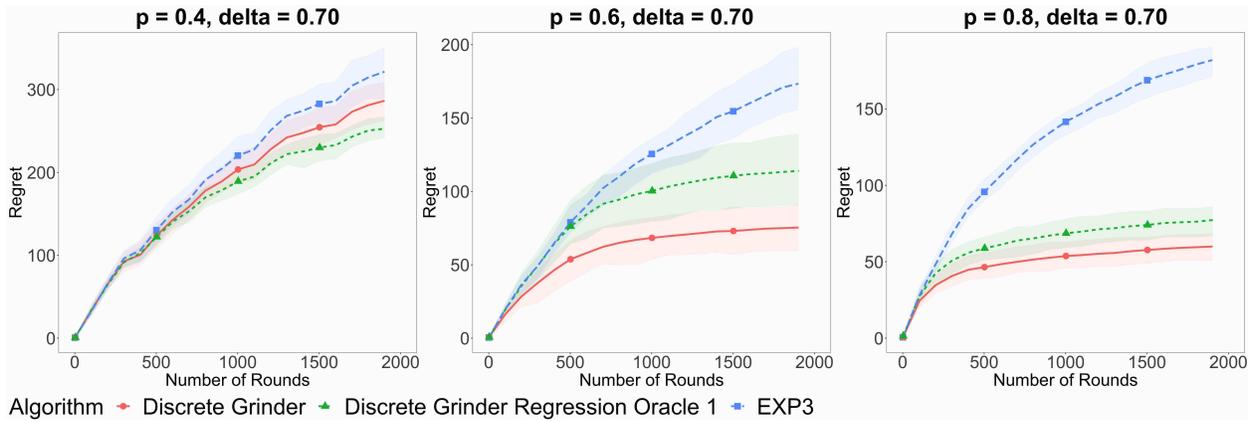


(a) GRINDER vs. EXP3 for a predefined action set.

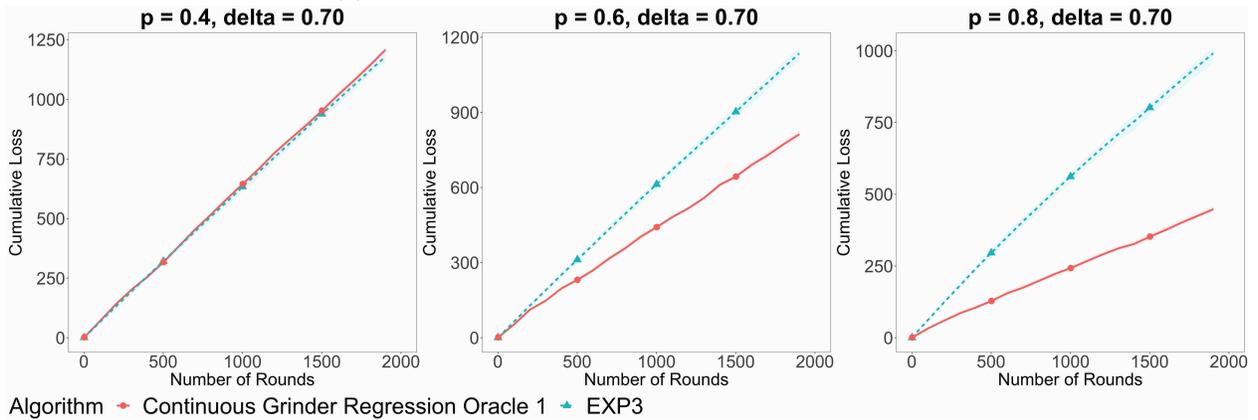


(b) GRINDER vs. EXP3 for a continuous action set.

Figure 5: Regret Performance of GRINDER vs EXP3 for  $\delta = 0.35$ ; in all cases, GRINDER converges faster.



(a) GRINDER vs. EXP3 for a predefined action set.



(b) GRINDER vs. EXP3 for a continuous action set.

Figure 6: Regret Performance of GRINDER vs EXP3 for  $\delta = 0.7$ ; in all cases, GRINDER converges faster.

In order to implement the 2-stage action draw method, we first chose a polytope (according to the probability function prescribed by the GRINDER algorithm) and then, by using *rejection sampling* from the bounding box around the polytope, we chose the action associated with it. Note that this is equivalent to the theoretical 2-stage draw.

As was also the case with the EXP3 algorithm, we lower bound the draw probabilities of each polytope by  $10^{-7}$ . In order to speed up our algorithm’s performance, we also used the heuristic of bounding the allowable volume of any polytope to be greater than or equal to 0.01, but in all the simulations that we tried, we saw comparable regret results even without the heuristic.

### D.3 Logistic Regression on Spammers

In this subsection, we will outline our implementation of the logistic regression algorithm on the spammers’ past data, which serves as an estimate of the in-probability for each action. For the ease of exposition, we will provide the description of the oracle for the case of a predefined action set, and subsequently, we will outline the way it generalizes to the continuous implementation.

Before we embark on this, allow us first to observe that we already have a very crude (but potentially useful) lower bound for *every* action  $j \in \mathcal{A}$ . Indeed, each action *always* updates itself, and actions that belong in the upper and lower polytope sets are always updated by all actions within these sets. The latter is due to the fact that for any hyperplane chosen within these sets, there is no possible manipulation from the perspective of the agent. Let us denote this crude lower bound for each action  $j \in \mathcal{A}$  by  $c^j$ .

Let us first define the notion of an *admissible timestep*; admissible timesteps are the ones during which the learner encounters a spammer. Everything that we mention in this subsection uses only admissible timesteps in order to build the training set. Labels are defined as  $l_i^j = 0$  if action  $j$  was *not* updated at timestep  $i$ <sup>15</sup>, and 1 otherwise. As a first step, this oracle computes for each action  $j \in \mathcal{A}$  the probability that each action from  $\mathcal{A}$  updates  $j$ , by using a logistic regression<sup>16</sup> with feature vectors the set  $H_{1:t}$ , and  $L_{1:t}^j$  as the labels. Let  $p_i^j, i \in \mathcal{A}$  correspond to the output probabilities, i.e.,  $p_i^j$  encodes the probability that action  $j$  will be updated by action  $i$ . The in-probability of action  $j$  is ultimately defined as:

$$\Pr^{\text{in}}[j] = \max \left\{ \sum_{i \in \mathcal{A}} p_i^j \pi_t[i], c^j \right\}$$

At a high-level, it is not hard to see how this can generalize to the continuous grinding case; instead of actions, one now uses whole *polytopes*. The implementation, however, becomes significantly messier.

### D.4 Different Approximation Oracle Results

In this subsection, we will present the results for GRINDER with a *different* approximation oracle against EXP3. The only difference between this oracle and the oracle defined in Subsection D.3 is that now the admissible timesteps are *all* timesteps  $t \in [T]$ .

Using the new oracle, we see in Figure 7 that the regret performance of GRINDER becomes much better under the (more realistic) cases of  $p = 0.6$  and  $p = 0.8$ , but we perform worse than the oracle which uses only the spammers’ data in the case that the majority of the agents that we face are

<sup>15</sup>In other words, action was at a distance less than  $2\delta$  from the best-response of the timestep.

<sup>16</sup>Technically, we run a different logistic regression for every action in  $\mathcal{A}$ .

strategic ( $p = 0.4$ ). The extent to which this oracle performs worse than the oracle of Section 5 is decreased in the continuous space simulations as shown in Figure 8, potentially due to the fact that the algorithm has the opportunity to grind the space differently, and recover from some errors that the logistic regression creates.

An explanation for the fact that this oracle performs so poorly for the case that  $p = 0.4$ , but also, performs better than the omnipotent oracle for some of the cases with  $p = 0.6, 0.8$  is that in order to achieve the *best* possible *upper* bound, the step size  $\eta$  should incorporate information about the variance of the logistic regression with respect to the *true* in-probability function, which is currently missing.

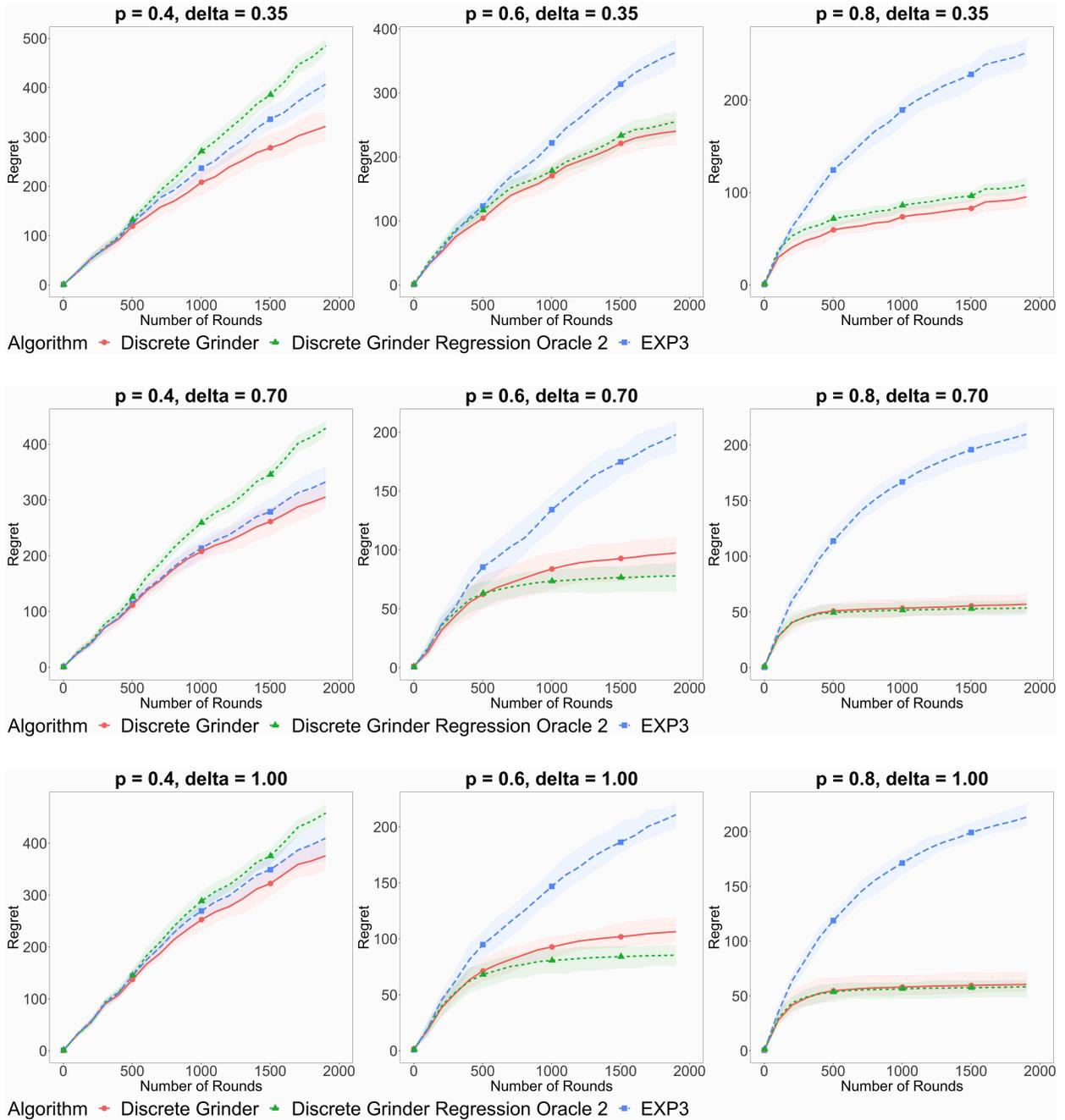


Figure 7: Regret performance of GRINDER vs. EXP3 for a predefined action set

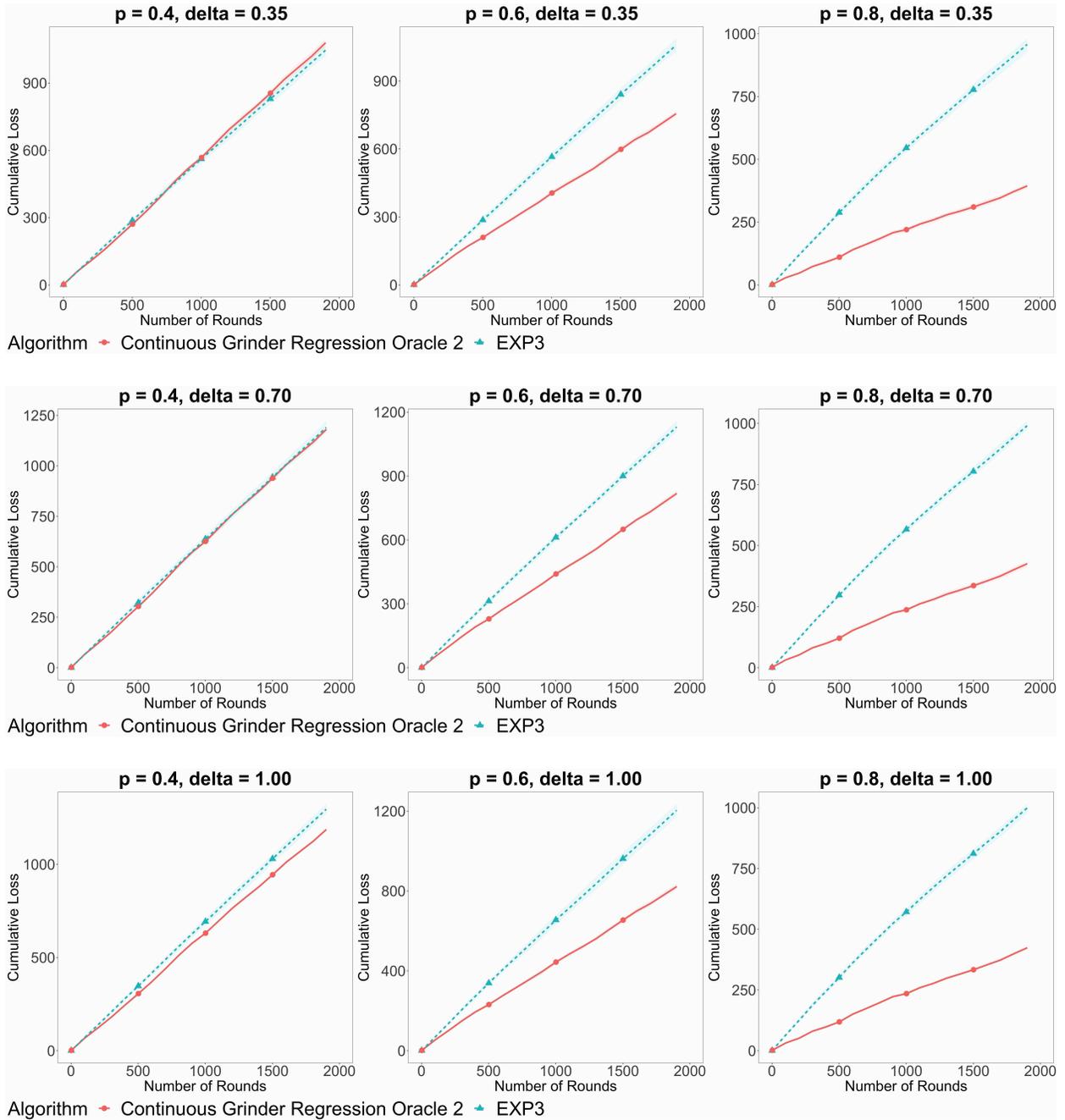


Figure 8: Regret performance of GRINDER vs. EXP3 for a continuous action set.