Composing Efficient, Robust Tests for Policy Selection (Supplementary Material)

Dustin Morrill[1] **Thomas J. Walsh**[1] **Daniel Hernandez**[1] **Peter R. Wurman**[1] **Peter Stone**[1,2]

[1]Sony AI, New York, NY, USA
[2]Department of Computer Science, The University of Texas at Austin, Austin, TX USA

1 GLOSSARY

Policy. A policy to solve a control problem or play a game, potentially generated by an RL algorithm.

Deployment policy. A policy used in production, *e.g.*, deployed to end users, used in a competition, or integrated into a technology demonstration.

Deployment candidate. A policy in consideration for deployment.

Test. The aggregate result of test cases applied to a policy.

Test case. An atomic unit of a test that reveals a particular skill or emulates a specific deployment scenario. RPOSST selects a small number of test cases and a distribution over them so that we can avoid executing all conceivable test cases on every deployment candidate every time we want to deploy a policy.

Test case result. The numerical result of evaluating a policy on a test case. This number should be a good estimate of the policy's expected performance in the test case scenario, but it maybe noisy if the test case is stochastic, *e.g.*, the average test case result observed from Monte Carlo rollouts.

Test score. The final score produced by a test, *i.e.*, the average test case result across test cases, perhaps weighted by the relative importance of each test case.

Tuning policy. A policy used at the start of the RPOSST procedure to gather information about test cases. Each tuning policy is evaluated on each test case to construct the test case result matrix that forms the basis of RPOSST's loss function.

2 THEORY BACKGROUND

We make use of six basic results, which are restated here for completeness.

Proposition 2.1 (Azuma-Hoeffding inequality.)**.** *For constants $\langle c^t \in \mathbb{R} \rangle_{t=1}^T$, martingale difference sequence $\langle Y^t \in \mathbb{R} \rangle_{t=1}^T$ where $|Y^t| \leq c^t$ for each t, and $\tau \geq 0$,*

$$\mathbb{P}\left[\left|\sum_{t=1}^{T} Y^t\right| \geq \tau\right] \leq 2\exp\left(\frac{-\tau^2}{2\sum_{t=1}^{T}(c^t)^2}\right).$$

For proof, see that of Theorem 3.14 by McDiarmid [1998].

Proposition 2.2 (Regret matching$^+$ regret bound)**.** *Consider an online decision process with m actions and the set of bounded, linear loss functions, $\mathcal{L} = [0, L]^m$. Regret matching$^+$ accumulates pseudoregrets $q^{1:t} = [q^{1:t-1} + \rho^t]_+$, $q^{1:0} = \mathbf{0}$, where $\rho^t = (\ell^t)^\top \sigma^t - \ell^t$ is the instantaneous regret on round t under loss function $\ell^t \in \mathcal{L}$, and $\sigma^t = q^{1:t-1}/(\mathbf{1}^\top q^{1:t-1})$ if $\mathbf{1}^\top q^{1:t-1} > 0$ or $\sigma^t = \frac{1}{m}\mathbf{1}$ otherwise, is regret matching$^+$'s action distribution on round t. After T rounds, regret matching$^+$'s cumulative regret is bounded as $\sum_{t=1}^{T} \rho^t \leq L\sqrt{Tm}$.*

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

For proof, see Tammelin et al. [2015].

Proposition 2.3 (The linearization trick)**.** *Consider an online decision process with convex decision set $\Theta \subseteq \mathbb{R}^m$ and a set of bounded, convex loss functions $\mathcal{L} \subseteq \{\ell \mid \ell : \Theta \to [0, L]\}$, where each loss function $\ell \in \mathcal{L}$ has subgradients with bounded maximum magnitude,* i.e.*, $\|\nabla\ell(\theta)\|_\infty \leq G$, for all $\theta \in \Theta$. The instantaneous regret under loss function $\ell \in \mathcal{L}$ is upper bounded by the instantaneous regret under the loss function subgradient $\nabla\ell(\theta)$ given decision $\theta \in \Theta$,* i.e.*,*

$$\ell(\theta) - \ell(\theta') \leq (\nabla\ell(\theta))^\top \theta - (\nabla\ell(\theta))^\top \theta'.$$

Proof. From the convexity of ℓ, its first-order Taylor expansion lower bounds ℓ, *i.e.*, $\ell(\theta') \geq \ell(\theta) + (\nabla\ell(\theta))^\top(\theta' - \theta)$, for all $\theta, \theta' \in \Theta$. Therefore,

$$\begin{aligned}\ell(\theta) - \ell(\theta') &\leq \ell(\theta) - \Big(\ell(\theta) + (\nabla\ell(\theta))^\top(\theta' - \theta)\Big) \\ &= (\nabla\ell(\theta))^\top\theta - (\nabla\ell(\theta))^\top\theta',\end{aligned}$$

as required. □

Proposition 2.4 (Lemma 2 of Lockhart et al. [2019a,b])**.** *Assume that on each round t of an online decision process with decision set $\Theta \subseteq \mathbb{R}^m$ and bounded loss functions from $\mathcal{L} \subseteq \{\ell \mid \ell : \Theta \to [0, L]\}$, the loss function ℓ^t maximizes the loss of $\theta^t \in \Theta$ chosen by the decision-maker,* i.e.*, $\ell^t \in \arg\max_{\ell\in\mathcal{L}} \ell(\theta^t)$. On the round t^* where the minimum loss was observed, $t^* \in \arg\min_{t\in\{1,\ldots,T\}} \ell^t(\theta^t)$, the decision θ^{t^*} has a maximum loss that is no more than $\frac{1}{T}\rho^{1:T}(\theta)$ larger than that of any alternative decision $\theta \in \Theta$,* i.e.*, $\ell^{t^*}(\theta^{t^*}) - \ell^\theta(\theta) \leq \frac{1}{T}\rho^{1:T}(\theta)$, where $\ell^\theta(\theta) \in \mathcal{L}$ is a loss function that maximizes the loss on θ.*

Proof. Since the loss function on each round is chosen to maximize loss, the average regret for not choosing $\theta \in \Theta$ is lower bounded as

$$\begin{aligned}\frac{1}{T}\rho^{1:T} &\geq \frac{1}{T}\min_{t\in\{1,\ldots,T\}} T\ell^t(\theta^t) - \frac{1}{T}\sum_{t=1}^{T}\ell^t(\theta) \\ &\geq \ell^{t^*}(\theta^{t^*}) - \ell^\theta(\theta),\end{aligned}$$

as required. □

Proposition 2.5 (Theorem 4 of Johanson et al. [2012])**.** *Assume that on each round t of an online decision process with decision set $\Theta \subseteq \mathbb{R}^m$ and bounded (possibly random) loss functions from $\mathcal{L} \subseteq \{\ell \mid \ell : \Theta \to [0, L]\}$, the loss function ℓ^t maximizes the loss of $\theta^t \in \Theta$ chosen by the decision-maker,* i.e.*, $\ell^t \in \arg\max_{\ell\in\mathcal{L}} \ell(\theta^t)$. The loss function that the decision-maker observes on each round t may be a random loss function $\hat{\ell}^t$ where $\mathbb{E}\Big[\hat{\ell}^t\Big] = \ell^t$. On round $T' \sim \mathrm{Unif}(\{1, \ldots, T\})$ after T rounds of the online decision process, the decision $\theta^{T'}$ has a maximum loss that is no more than $\frac{1}{qT}\rho^{1:T}(\theta)$ larger than that of any alternative decision $\theta \in \Theta$ with probability $1 - q$, $q \in (0, 1]$,* i.e.*, $\ell^{T'}(\theta^{T'}) - \ell^\theta(\theta) \leq \frac{1}{qT}\rho^{1:T}(\theta)$ holds with probability $1 - q$, where $\ell^\theta(\theta) \in \mathcal{L}$ is a loss function that maximizes the loss on θ and the cumulative regret $\rho^{1:T}$ is with respect to the expected loss functions, $\langle\ell^t\rangle_{t=1}^T$.*

See Johanson et al. [2012] for proof.

Proposition 2.6 (Successive Rejects error probability)**.** *Consider a best action identification task with m actions from set $\mathcal{A}$. Each time an action $a \in \mathcal{A}$ is selected, a random sample of that action's loss, $\ell(a) \in [-0.5, 0.5]$, under a fixed but random loss function ℓ, is observed. The goal is to identify an action $a^* \in \mathcal{A}^* \subset \mathcal{A}$ with the lowest expected loss, $\mathbb{E}[\ell(a^*)]$, after T samples. The probability that the action returned by the Successive Rejects algorithm is in $\mathcal{A}^*$ is at least*

$$1 - \frac{m(m-1)}{2}\exp\left(-\frac{T-m}{\overline{\log}(m)H_2}\right),$$

where $\overline{\log}(m) = \frac{1}{2} + \sum_{i=2}^{m}\frac{1}{i}$, $H_2 = \max_{i\in\{1,\ldots,|\mathcal{A}\setminus\mathcal{A}^*|\}} \frac{i}{\left(\mathbb{E}\left[\ell(a_{(i)})\right] - \mathbb{E}[\ell(a^*)]\right)^2}$, and $a_{(i)}$ is the action that achieves the i^{th} smallest loss (with ties broken arbitrarily) among the suboptimal actions.

See Audibert et al. [2010] for proof.

3 SEQUENTIAL-MOVE MODEL THEORY

Lemma 3.1. *Consider a k-of-N game with m actions and the set of bounded, convex loss functions $\mathcal{L} = \{\ell \mid \ell : \triangle^m \to [0, L]\}$, where each loss function $\ell \in \mathcal{L}$ has subgradients with bounded maximum magnitude,* i.e.*, $\|\nabla\ell(\sigma)\|_\infty \leq G$, for all $\sigma \in \triangle^m$. Let the k-worst loss functions from N of those sampled from the given uncertainty distribution Ψ on round t be $\langle \ell^t_{(i)} \in \mathcal{L}\rangle_{i=1}^k$. The randomly sampled k-of-N loss function on round t is then the average $\bar{\ell}^t = \frac{1}{k}\sum_{i=1}^k \ell_{(i)}$. After T rounds, regret matching$^+$ on the random loss gradients $\nabla\bar{\ell}^t(\sigma^t)$ has no more than $2G\sqrt{Tm} + 2L\sqrt{2T\log 1/p}$ cumulative regret on the expected k-of-N losses, $\left\langle \mathbb{E}[\bar{\ell}]^t \right\rangle_{t=1}^T$, with probability $1-p$, $p > 0$.*

Proof. Since regret matching$^+$ observes and learns directly from $\nabla\bar{\ell}^t$, its regret for not always choosing $\sigma \in \triangle^m$, under the sampled loss functions, is deterministically upper bounded as

$$R^{1:T} = \sum_{t=1}^T \underbrace{\bar{\ell}^t(\sigma^t) - \bar{\ell}^t(\sigma)}_{\doteq R^t} \leq 2G\sqrt{Tm},$$

where $\langle \sigma^t \in \triangle^m\rangle_{t=1}^T$ are the decisions made by regret matching$^+$. This bound comes from regret matching$^+$'s regret bound on linear losses (Proposition 2.2) and the linearization trick (Proposition 2.3), which states that the regret on loss gradients upper bounds that of the loss itself, *i.e.*, $R^{1:T} \leq \sum_{t=1}^T \big(\nabla\bar{\ell}^t(\sigma^t)\big)^\top \sigma^t - \big(\nabla\bar{\ell}^t(\sigma)\big)^\top \sigma$.

The rest of the proof largely follows the proof of Farina et al. [2020]'s Proposition 1. The sequence of differences, $\langle \mathbb{E}[R^t] - R^t \leq 2L\rangle_{t=1}^T$, is a bounded martingale difference sequence.

The probability that the expected cumulative regret, $\mathbb{E}[R^{1:T}]$, is bounded by the cumulative sampled regret plus slack $\lambda \geq 0$ is bounded according to the Azuma-Hoeffding inequality (Proposition 2.1) as

$$\begin{aligned}
&\mathbb{P}\big[\mathbb{E}[R^{1:T}] \leq R^{1:T} + \lambda\big] && (1)\\
&\leq \mathbb{P}\left[\sum_{t=1}^T \mathbb{E}[R^t] - R^t \leq \lambda\right] && (2)\\
&= 1 - \mathbb{P}\left[\sum_{t=1}^T \mathbb{E}[R^t] - R^t \geq \lambda\right] && (3)\\
&\leq 1 - \exp\left(\frac{2\lambda^2}{4T(2L)^2}\right). && (4)
\end{aligned}$$

Setting $\lambda = 2L\sqrt{2T\log(1/p)}$ ensures that

$$\mathbb{E}[R^{1:T}] \leq R^{1:T} + 2L\sqrt{2T\log 1/p}$$

with probability $1-p$. Since $R^{1:T} \leq 2L\sqrt{Tm}$,

$$\mathbb{E}[R^{1:T}] \leq 2G\sqrt{Tm} + 2L\sqrt{2T\log 1/p}$$

with probability $1-p$, as required. □

Theorem 3.2. *After $T' \sim \mathrm{Unif}(\{1, \ldots, T_1\})$, $T_1 > 0$, rounds of its optimization game, Algorithm 1 selects an m-tuple of test cases, τ^* and weights $\hat{\sigma}^{T'}_{\tau^*} \in \triangle^m$ that, with probability $(1-p)(1-q)(1-\alpha)$, $p, q, \alpha > 0$, are $\frac{\varepsilon}{q}$-optimal for Equation* (2)*, where $\varepsilon = \mathcal{O}\Big(\sqrt{\frac{1}{T_1}m} + \sqrt{\frac{1}{T_1}\log(1/p)}\Big)$ and $\alpha = \mathcal{O}\big(\mathrm{e}^{-T_2}\big)$.*

Proof. Recall that the k-of-N loss $\bar{\ell}^t$ that RPOSST$_{\text{SEQ}}$ updates from on each round $t = 1, \ldots, T_1$ is a Monte Carlo estimate of the k-of-N percentile loss,

$$L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}^t_\tau) = \inf_{y\in\mathcal{Y}} \int\limits_{\substack{\eta\in[0,1]\\ \mathbb{P}_{\pi_j,\sigma}\left[\ell(\hat{\sigma}^t_\tau;\pi_j,\sigma)\leq y(\eta)\right]\geq\eta}} y(\eta)\mu_{k\text{-of-}N}(d\eta) = \mathbb{E}_{\pi_j,\sigma}[\bar{\ell}^t], \tag{5}$$

where $(\pi_j, \sigma) \sim \Psi$. The sequence of test case weights, $\langle \sigma_\tau^t \rangle_{t=1}^{T_1}$, for each m-tuple of test cases $\tau \subset \mathcal{T}$ is therefore random. All of the following probabilities and expectations are with respect to these random variables.

Lemma 3.1 guarantees that RPOSST$_{\text{SEQ}}$, in generating the test case weight sequence $\langle \sigma_\tau^t \rangle_{t=1}^{T_1}$ has no more than $C = 2G\sqrt{T_1 m} + 2L\sqrt{2T_1 \log 1/p}$ cumulative regret on the k-of-N percentile losses,

$$\rho_{\sigma_\tau}^{1:T_1} = \sum_{t=1}^{T} L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}_\tau^t) - L_{\mu_{k\text{-of-}N},\Psi}(\sigma_\tau),$$

for not always selecting test case weights σ_τ, with probability $1 - p$. That is, $1 - p = \mathbb{P}\big[\rho_{\sigma_\tau}^{1:T_1} \leq C\big]$.

Proposition 2.5 guarantees that, on round $T' \sim \text{Unif}(1, \dots, T_1)$, the weights for each m-tuple are $\frac{1}{qT_1}\rho_{\sigma_\tau}^{1:T_1}$ close to optimal for Equation (5), with probability $1 - q$. That is, $1 - q = \mathbb{P}\left[L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}_\tau^{T'}) - L_{\mu_{k\text{-of-}N},\Psi}(\sigma_\tau) \leq \frac{\rho_{\sigma_\tau}^{1:T_1}}{qT_1}\right]$, and this holds regardless of the value of $\rho_{\sigma_\tau}^{1:T_1}$, *i.e.*, $\mathbb{P}\left[L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}_\tau^{T'}) - L_{\mu_{k\text{-of-}N},\Psi}(\sigma_\tau) \leq \frac{\rho_{\sigma_\tau}^{1:T_1}}{qT_1}\right] = \mathbb{P}\left[L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}_\tau^{T'}) - L_{\mu_{k\text{-of-}N},\Psi}(\sigma_\tau) \leq \frac{\rho_{\sigma_\tau}^{1:T_1}}{qT_1} \mid \rho_{\sigma_\tau}^{1:T_1} \leq C'\right]$ for all $C' \in \mathbb{R}$.

Combining these two results, we see that the probability that $\hat{\sigma}_\tau^{T'}$ has at most $\frac{C}{qT_1}$ excess k-of-N percentile loss is

$$\begin{aligned}
\mathbb{P}\left[L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}_\tau^{T'}) - L_{\mu_{k\text{-of-}N},\Psi}(\sigma_\tau) \leq \frac{C}{qT_1}\right] &= \mathbb{P}\left[L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}_\tau^{T'}) - L_{\mu_{k\text{-of-}N},\Psi}(\sigma_\tau) \leq \frac{\rho_{\sigma_\tau}^{1:T_1}}{qT_1}, \rho_{\sigma_\tau}^{1:T_1} \leq C\right] && (6)\\
&= \mathbb{P}\left[L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}_\tau^{T'}) - L_{\mu_{k\text{-of-}N},\Psi}(\sigma_\tau) \leq \frac{\rho_{\sigma_\tau}^{1:T_1}}{qT_1} \mid \rho_{\sigma_\tau}^{1:T_1} \leq C\right]\mathbb{P}\big[\rho_{\sigma_\tau}^{1:T_1} \leq C\big] && (7)\\
&= \mathbb{P}\left[L_{\mu_{k\text{-of-}N},\Psi}(\hat{\sigma}_\tau^{T'}) - L_{\mu_{k\text{-of-}N},\Psi}(\sigma_\tau) \leq \frac{\rho_{\sigma_\tau}^{1:T_1}}{qT_1}\right]\mathbb{P}\big[\rho_{\sigma_\tau}^{1:T_1} \leq C\big] && (8)\\
&= (1-p)(1-q). && (9)
\end{aligned}$$

The last remaining step is to complete the outer minimization in Equation (2) to select a single m-tuple of test cases. Since the k-of-N loss observed on each round is random, we cannot compute a simple argmin using the test case weights on round T', and are instead faced with a best arm identification problem. For this, we run the Successive Rejects algorithm, which we know from Proposition 2.6 identifies a minimum loss m-tuple of test cases with probability at least

$$\alpha = 1 - \frac{m(m-1)}{2}\exp\left(-\frac{T_2 - m}{\overline{\log}(m)H_2}\right).$$

The probability of selecting the best m-tuple using the test case weights on round T' is independent of whether or not the regret bound C was actually achieved or if the test case weights on T' are actually nearly optimal for any given m-tuple, the probability of which we previously characterized as $(1-p)(1-q)$. Therefore, the probability of achieving $\frac{C}{qT_1}$-optimality given each m-tuple and selecting the best m-tuple is the product $(1-p)(1-q)(1-\alpha)$, as required. □

The $\sqrt{|\mathcal{T}|}$ dependence in Theorem 3.2 could be improved to $\sqrt{\log(|\mathcal{T}|)}$ if regret matching$^+$ (within or without CFR, respectively) was replaced with an algorithm like Hedge [Freund and Schapire, 1997], but this tends to lead to worse performance in practice (see, *e.g.*, Tammelin et al. [2015], Burch [2017]).

In the deterministic CVaR RPOSST case, we get the following corollary.

Corollary 3.3. *Assume that $\Psi \in \triangle^d$ for some finite $d \geq 1$. After T rounds of the CVaR(η) RPOSST$_{\text{SEQ}}$ optimization game, where the protagonist chooses m-size tests according to regret matching$^+$ against a best response antagonist, τ^* and $\sigma_{\tau^*}^{t^*}$ are ε-optimal for Equation* (2) *under the η-fractile CVaR robustness measure, where $\varepsilon = \mathcal{O}\left(\sqrt{\frac{1}{T}m}\right)$.*

Proof. Proposition 2.2 and Proposition 2.4 ensures that there is a round $t_\tau^* \leq T$ where $\sigma_\tau^{t_\tau^*}$ is $2G\sqrt{m\frac{1}{T}}$-optimal on the deterministic k-of-N losses. Since the k-of-N loss function observed on each round is deterministic, we can perform a

simple minimization across $\{1, \ldots, T\}$ and the m-tuple of test cases to find the minimizers t^* and τ^*, leading to the stated optimality guarantee. □

4 DETERMINISTIC CVAR(η) RPOSST$_{\text{SEQ}}$ PSEUDOCODE

Pseudocode for CVaR(η) RPOSST$_{\text{SEQ}}$ is presented in Algorithm 1.

5 SIMULTANEOUS-MOVE MODEL

We present a more in-depth description of the simultaneous move antagonist model which describes the RPOSST$_{SIM}$ as introduced in Section 4. This description is complemented by pseudocode describing its workings in Algorithm 2.

In this model, the antagonist does not observe which m-tuple of test cases, τ, is sampled from the protagonist's $\hat{\sigma}^t_{\mathcal{T}} \in \triangle^{|\mathcal{T}|^m}$ distribution, making the antagonist role more difficult. The simultaneous move model corresponds to the policy testing use case where a new m-tuple of test cases is sampled independently for each test that is performed. Effectively, the protagonist and antagonist choose τ and $\left\langle\left\langle \pi_{j_{(i)}}, \sigma_{(i)} \right\rangle\right\rangle_{i=1}^{k}$ respectively in a simultaneous fashion. In this model, the antagonist must choose a single list of tuples $\left\langle\left\langle \pi_{j_{(i)}}, \sigma_{(i)} \right\rangle\right\rangle_{i=1}^{k}$ that will lead to a large loss across all of the m-tuples of test cases that the protagonist might choose, thereby preventing the antagonist from exploiting the lacking aspects of each individual m-tuple.

The protagonist in the simultaneous move model must carefully choose $\hat{\sigma}^t_{\mathcal{T}}$ and each m-tuple distribution, $[\hat{\sigma}^t_\tau]_{\tau \in \mathcal{T}^m}$, to thwart the antagonist. We organize the protagonist's actions into two sequential decisions: first choosing the m-tuple τ and then choosing $\hat{\sigma}^t_\tau$ given τ. We then use CFR$^+$ to refine both $\hat{\sigma}^t_{\mathcal{T}}$ and each $\hat{\sigma}^t_\tau$ after each round.

Instantiating the percentile performance loss of Equation (1) for the simultaneous move model, the RPOSST objective is,

$$\min_{\substack{\hat{\sigma}_{\mathcal{T}} \in \triangle^{|\mathcal{T}|^m} \\ [\hat{\sigma}_\tau \in \triangle^m]_{\tau \in \mathcal{T}^m}}} \inf_{y \in \mathcal{Y}} \int_{\substack{\eta \in [0,1] \\ \mathbb{P}\left[\mathbb{E}_{\tau \sim \hat{\sigma}_{\mathcal{T}}}[\ell(\hat{\sigma}_\tau, \sigma, \pi_j)] \leq y(\eta)\right] \geq \eta}} y(\eta) \mu_{k\text{-of-}N}(d\eta), \tag{10}$$

where $\sigma, \pi_j \sim \Psi$.

After (linearly) averaging the protagonist's choices of $\hat{\sigma}^t_{\mathcal{T}}$ and $[\hat{\sigma}^t_\tau]_{\tau \in \mathcal{T}^m}$ across each round, Algorithm 2 returns the average distributions $\bar{\hat{\sigma}}^T_{\mathcal{T}}$ and $\left[\bar{\hat{\sigma}}^T_\tau\right]_{\tau \in \mathcal{T}^m}$.

The simultaneous-move model can be made deterministic using a CVaR measure in the same way as the sequential-move model. If we fix the ratio k/N and allow $N \to \infty$, the k-of-N robustness measure converges toward the CVaR measure at the k/N fractile. Furthermore, if our the distribution characterizing our uncertainty, Ψ, is over a discrete set of manageable size, then we can run RPOSST on CVaR robustness measures. In RPOSST$_{\text{SIM}}$, the lowest loss test case distributions across all rounds can also be tracked instead of averaging all of the distributions.

6 EXPERIMENTAL DETAILS

In this section we provide further details on some of the experimental setups used in Section 5.

All CVaR(1%) RPOSST$_{\text{SEQ}}$ procedures were run on a 16 core AMD® Ryzen 7 5800h CPU with 30.7 GiB of memory. See Table 1 for the time required to run CVaR(1%) RPOSST$_{\text{SEQ}}$ on each domain.

6.1 RACING ARROWS

Racing Arrows is a two-player, zero-sum, one-shot, continuous action game that replicates simple aspects of a passing scenario in a race featuring a "leader" player and faster "follower" player. The goal of the follower is to pass the leader while the goal of the leader is to block the follower.

Both players privately choose an angle in the half-circle between 0 and pi for their arrow. The speed of each player is represented as the length of their arrow. The leader and follower are assigned a speed according to their roles, where the

leader's speed of 0.8 is slightly slower than the follower's speed of 1 to give the follower a chance to pass. The distance a player travels is the height of their arrow, *i.e.*, speed $\cdot \sin(\text{angle})$.

The follower is blocked and the leader wins if the difference between the two arrows is below $\pi/10$, that is, the leader is close enough to block the follower. If the follower is not blocked, then the player who traveled the farthest wins. Players receive $+1$ for a win, 0 for a loss, or 0.5 if they travel exactly the same distance (these payoffs sum to the constant $+1$, which is isomorphic to true zero-sum payoffs).

6.2 ANNUAL COMPUTER POKER COMPETITION

The Annual Computer Poker Competition (ACPC) was run to test autonomous poker playing agents from 2006 to 2017. The logs of play are freely available online.[1] Typically, these competitions are Texas hold'em variants: two-player limit, two-player no-limit, and 3-player limit, where "limit" and "no-limit" indicates whether players are only allowed to bet in fixed increments or if they can bet any number of chips from their current stack, respectively. Chip stacks reset to their initial sizes after every hand (Doyle's game) so that players can be evaluated on their average one-hand performance across deck shufflings and seat positions.

To reduce variance, hands are played in duplicate, which means that the same deck order is played out multiple times so that each player has a turn playing with the same hands. For example, if Alice in seat 1 is dealt the ace and king of spades and Bob in seat 2 is dealt the 2 and 7 of hearts in one hand, then Alice and Bob will also play the same hand in opposite positions, where Bob is dealt the ace and king of spades in seat 1, and Alice is dealt the 2 and 7 of hearts. Alice's duplicate score is then the number of chips she wins over what Bob won in the same position, averaged across both positions.

Our experiments use duplicate score data, *i.e.*, a test case result here is a duplicate score between two agents, from the 2012 two-player limit and the 2017 two-player no-limit events.

6.3 GRAN TURISMO™ 7

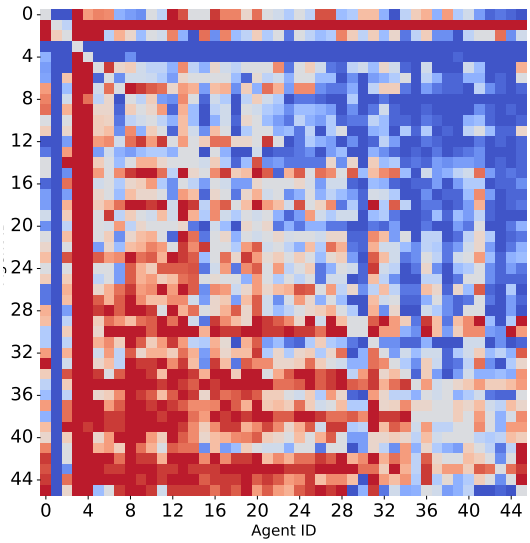
Our Gran Turismo™ 7 experiments were conducted using the Gran Turismo™ 7 racing simulator. Previous versions of the Gran Turismo™ 7 franchise have been used to exhibit reinforcement learning results [Fuchs et al., 2021, Song et al., 2021] including outracing top human drivers [Wurman et al., 2022]. Note our focus was not on agent training but rather the problem of selecting the best policy for a deployment, so for training we used the same training parameters reported by by Wurman et al. except for changes to training scenarios to match the track and car combination chosen for this experiment, training only for one-on-one competition, and utilizing a version of self-play to simplify the training process.

The experiment was conducted at the Trial Mountain racetrack (see Figure 1a) with the RL policy (and any RL-trained opponent policies) driving a Chevrolet Corvette C7 Stingray '14 using Sport Hard tires. The track and car were chosen because the long straightaways and sharp turns at Trial Mountain led to competitive racing among various RL policies as there are many different areas of the track where passes can occur and the long straightaways allow the agent to use the slipstream of the other car to stay in touch with the car in front.

From a single one-on-one training run we evaluated checkpoints from epochs 5, 200, and then every 75 epochs between epoch 1000 and 4000 for a total of 43 checkpoints. We also evaluated 3 built-in AI agent using cars and tires that made them competitive with the RL agents. Overall we evaluated 46 policies, each of which was considered as a candidate deployment policy or an opponent in a test case.

To create the result matrix shown in Figure 1b, each race was run 20 times with a side-by-side standstill start with the candidate and opponent policies swapping sides half the time to enforce symmetry. An agent would obtain 1 or 0 for winning or losing the race respectively. The diagonal denoting a race between an agent against itself was filled in with 0.5 entries. As a second experiment on Gran Turismo™ 7 for a non-zero sum game, using the sportsmanship rule mentioned in Section 5 we recomputed the result matrix from Figure 1b so as to penalize trajectories where any car collisions had happened, giving both agents a payoff of -1. We remove the entries in the result matrix related to built-in AIs as they are highly collision averse and therefore the sportsmanship constraints would not change their test results, reducing the test case pool size to 43.

(a)



(b)

Figure 1: Figure 1a shows a screenshot of two RL agents racing at the Trial Mountain racetrack. The layout of the track can be seen in the top right. Figure 1b shows the result matrix for the zero-sum experiment. Blue / red colors indicate positive / negative winrates from the point of view of the column player. Agents 0-2 correspond to the different built-in AIs, with the remaining agents being the trained RL agents sorted according to skill. Diagonal values denote an agent playing against itself, which we artificially set to 50%.

6.4 SUPPLEMENTAL EXPERIMENTAL RESULTS

Figure 2 in Section 5 analyses the quantitative performance of RPOSST and its algorithmic ablations with respect to measuring test scores on a holdout set of unseen candidate deployment policies. We complement that analysis with a qualitative study of behaviors exhibited by the algorithms using the large GT experiment with holdout of size 20 as a representative example. We are interested in examining (1) how deterministic each algorithm's output is with respect to the selection of test case pairs and (2) whether different algorithms choose the same test-cases.

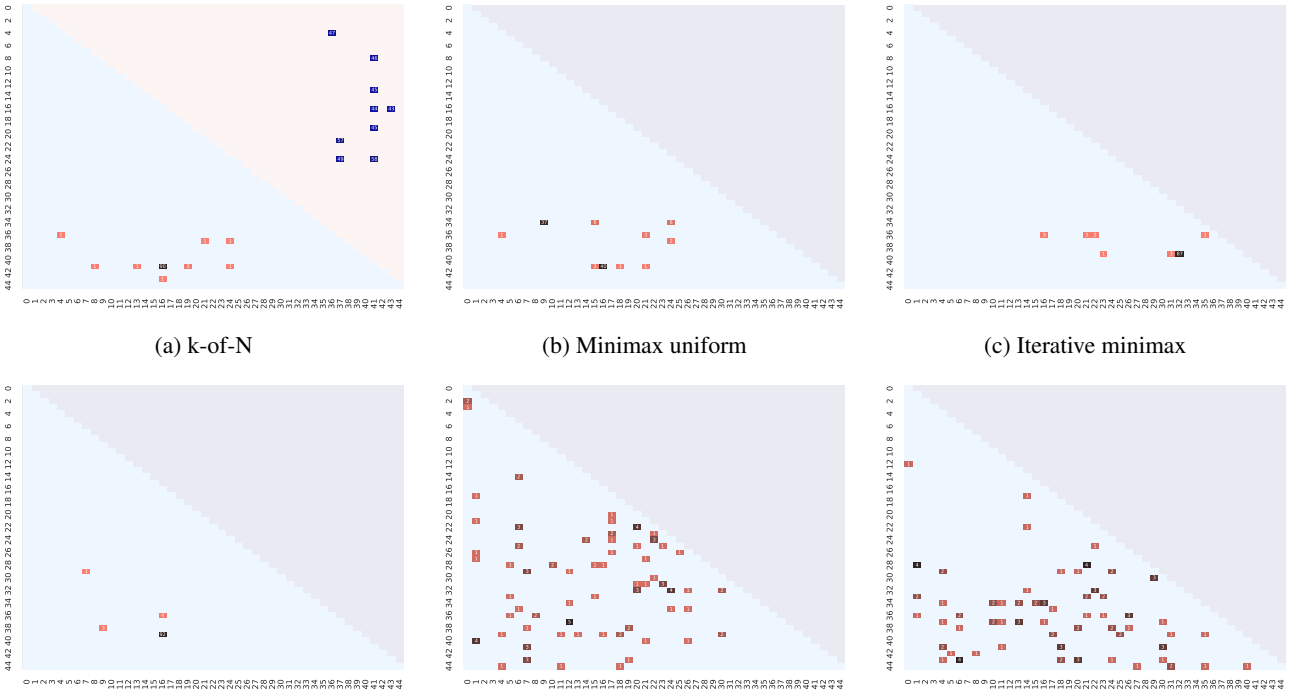
The lower triangular matrices from Figure 2 show the frequency at which test case pairs were chosen over the 100 seeds. The top 2 most selected test case pairs for each algorithm are presented in Table 2. We observe that RPOSST, alongside Iterative minimax and Minimax(TNP) uniform are very deterministic algorithms, favouring the selection of the same test case pair over 90%, 87% and 92% of the seeds respectively. We deem this a desirable property, as variance in evaluation scenarios is undesirable because it can hamper interpretability and reproducibility. In contrast, Minimax uniform exhibits a bimodal choice. The remaining algorithms feature a very high variance in their choice of test case pairs, with their most chosen test case pair being selected 5% of the time, spreading selection widely.

From Table 2, test case 16 is heavily favoured by half of the algorithms (RPOSST$_{\text{SEQ}}$, Minimax uniform and Minimax(TNP) uniform), followed to a lesser extent by test case 41. This indicates that all these algorithms find useful structure in such pairs of agents.

In Figures 3 to 8, we show the performance of RPOSST$_{\text{SEQ}}$ and baselines in each domain across test sizes ($m \in \{1, 2, 3\}$) and holdout proportions (20%, 40%, and 60%). Figure 9 shows the results for the 500 policy Racing Arrows experiment where the leader policies are treated as test cases.

We note that as m increases, the error on the holdout set typically decreases, particularly for RPOSST, since larger tests

[1] http://www.computerpokercompetition.org/downloads/competitions



(a) k-of-N (b) Minimax uniform (c) Iterative minimax

(d) Minimax(TNP) uniform (e) Miniaverage uniform (f) Minimax(TTD) uniform

Figure 2: Triangular matrices denoting frequencies of test case pairs choosen by all 6 algorithmic ablations over 100 runs of the winrate GT experiment on a holdout of size 20%. This visualization is possible because only 2 test cases where chosen as output. The upper triangular matrix from Figure 2a denotes average probability mass given to test case i. All other algorithms are limited to uniformly mixing over test cases so the upper triangular matrix is omitted for clarity.

have the capacity to be strictly more accurate. A qualitative analysis of these results suggests that there are few substantial differences between RPOSST tests of different sizes or with different, reasonably sized, holdout sets. Furthermore, the performance ordering of the tested algorithms remains the same as the results presented in the main paper.

References

Jean-Yves Audibert, Sébastien Bubeck, and Rémi Munos. Best arm identification in multi-armed bandits. In *COLT*, pages 41–53. Citeseer, 2010.

Neil Burch. *Time and space: Why imperfect information games are hard*. PhD thesis, University of Alberta, 2017.

Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Stochastic regret minimization in extensive-form games. In *International Conference on Machine Learning*, pages 3018–3028, 2020.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Florian Fuchs, Yunlong Song, Elia Kaufmann, Davide Scaramuzza, and Peter Dürr. Super-human performance in Gran Turismo Sport using deep reinforcement learning. *IEEE Robotics and Automation Letters*, 6(3):4257–4264, 2021. doi: 10.1109/LRA.2021.3064284.

Michael Johanson, Nolan Bard, Neil Burch, and Michael Bowling. Finding optimal abstract strategies in extensive form games. In *26th AAAI Conference on Artificial Intelligence (AAAI-12)*, 2012.

Edward Lockhart, Marc Lanctot, Julien Pérolat, Jean-Baptiste Lespiau, Dustin Morrill, Finbarr Timbers, and Karl Tuyls. Computing approximate equilibria in sequential adversarial games by exploitability descent. In *IJCAI 2019*, 2019a.

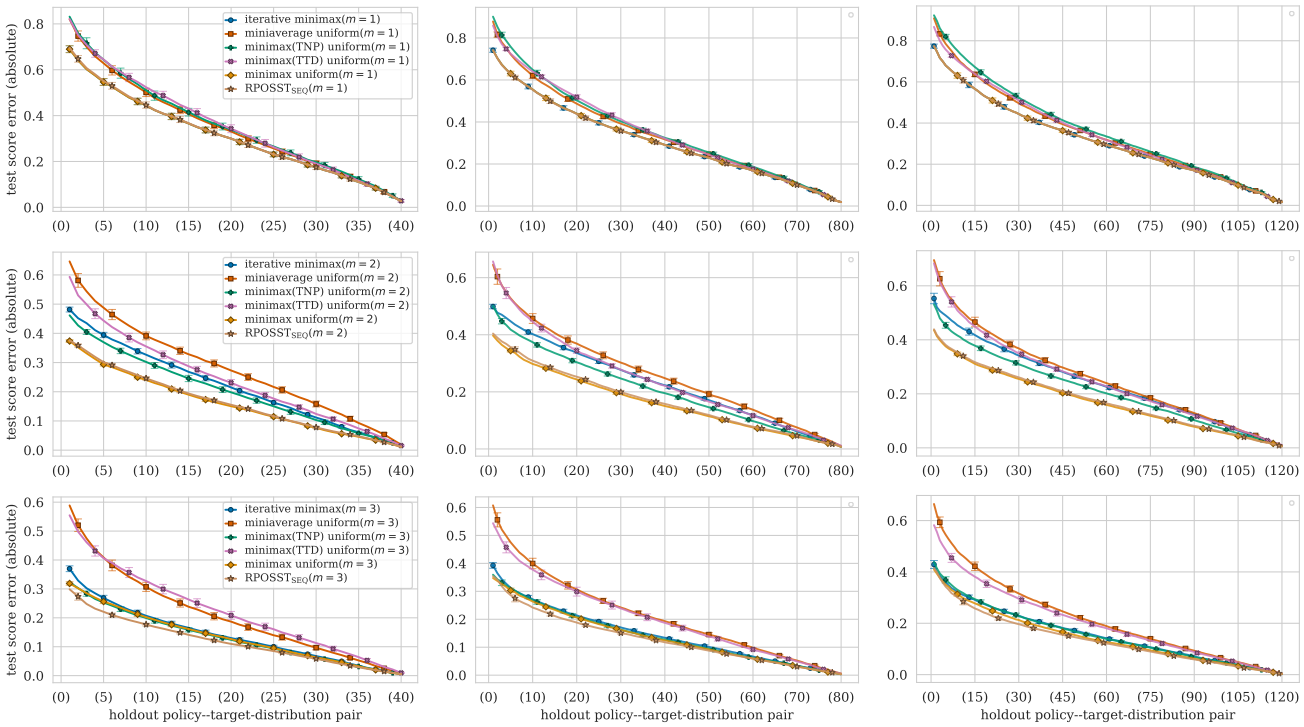


Figure 3: Expected test score error (absolute difference) across holdout-policy–target-distribution pairs on Racing Arrows where test cases are 50 follower policies. Each row uses a different setting for the test size ($m = 1$ top, $m = 2$ middle, and $m = 3$ bottom) and each column uses a different holdout proportion (20% held out in the left column, 40% middle, and 60% right). 100 sets of holdout policies were sampled. Holdout-policy–target-distribution pairs are sorted according to test score error. Each RPOSST$_{\text{SEQ}}$ instance was run for 500 rounds ($T = 500$). Errorbars represent 95% t-distribution confidence intervals.

Edward Lockhart, Marc Lanctot, Julien Pérolat, Jean-Baptiste Lespiau, Dustin Morrill, Finbarr Timbers, and Karl Tuyls. Computing approximate equilibria in sequential adversarial games by exploitability descent. *arXiv preprint arXiv:1903.05614*, 2019b.

Colin McDiarmid. Concentration. In *Probabilistic methods for algorithmic discrete mathematics*, pages 195–248. 1998.

Yunlong Song, HaoChih Lin, Elia Kaufmann, Peter Dürr, and Davide Scaramuzza. Autonomous Overtaking in Gran Turismo Sport Using Curriculum Reinforcement Learning. In *ICRA*, 2021.

Oskari Tammelin, Neil Burch, Michael Johanson, and Michael Bowling. Solving heads-up limit texas hold'em. In *24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, 2015.

Peter R Wurman, Samuel Barrett, Kenta Kawamoto, James MacGlashan, Kaushik Subramanian, Thomas J Walsh, Roberto Capobianco, Alisa Devlic, Franziska Eckert, Florian Fuchs, et al. Outracing champion gran turismo drivers with deep reinforcement learning. *Nature*, 602(7896):223–228, 2022.

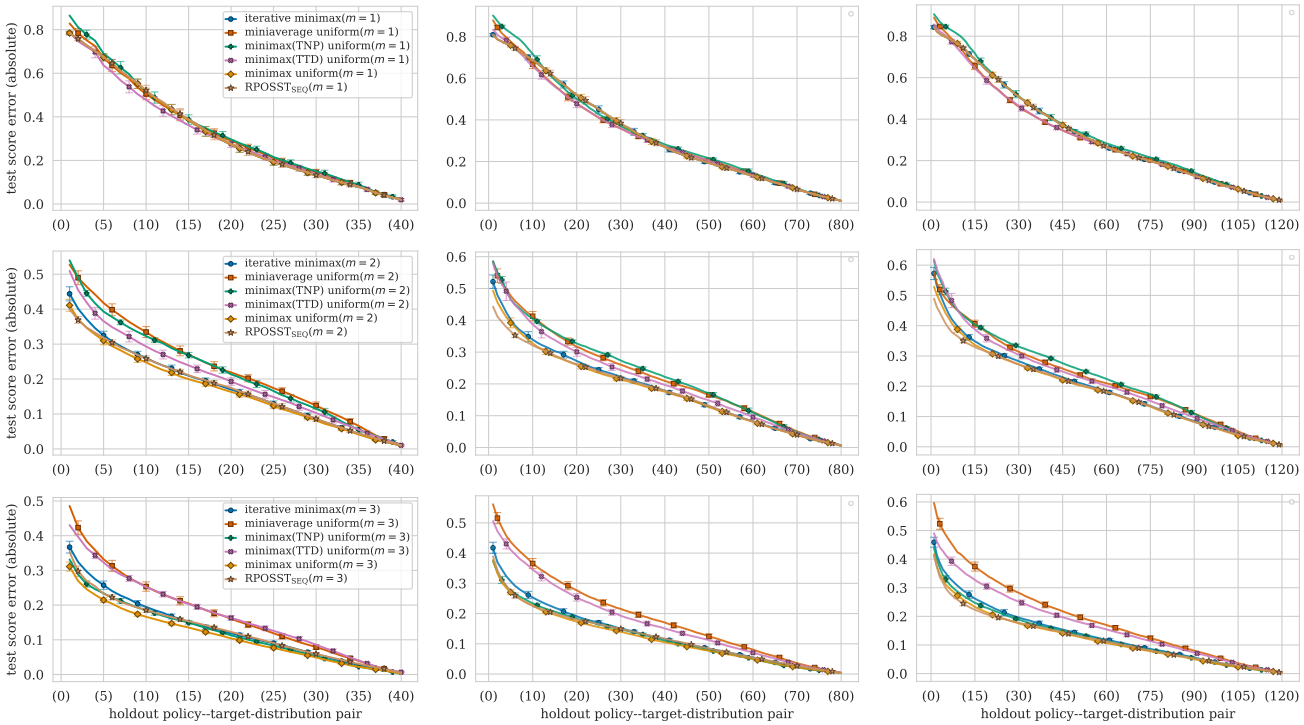


Figure 4: Expected test score error (absolute difference) across holdout-policy–target-distribution pairs on Racing Arrows where test cases are 50 leader policies. Each row uses a different setting for the test size ($m = 1$ top, $m = 2$ middle, and $m = 3$ bottom) and each column uses a different holdout proportion (20% held out in the left column, 40% middle, and 60% right). 100 sets of holdout policies were sampled. Holdout-policy–target-distribution pairs are sorted according to test score error. Each RPOSST$_{\text{SEQ}}$ instance was run for 500 rounds ($T = 500$). Errorbars represent 95% t-distribution confidence intervals.

Inputs: $\langle \eta, T, m, \Psi, \tau^0, \ell \rangle$

$q_\tau^{1:0} \leftarrow \mathbf{0} \in \mathbb{R}^{m+|\tau^0|}$ **for** $\tau \in \mathcal{T}^m$

$t^* \leftarrow 1$

$\bar{v}^{t^*} \leftarrow -\infty$

for $t \leftarrow 1, \ldots, T$ **do**

  for $\tau \in \mathcal{T}^m$ **do**

    $z^t \leftarrow \mathbf{1}^\top q_\tau^{1:t-1}$

    $\hat{\sigma}_\tau^t \leftarrow q_\tau^{1:t-1} / z^t$ **if** $z^t > 0$ **else** $\mathbf{1}/m$

    // Fill in zeros so that $\hat{\sigma}_\tau^t \in \triangle^{|\mathcal{T}|}$.

    $\hat{\sigma}_\tau^t(x) \leftarrow 0$ **for** $x \in \mathcal{T} \setminus (\tau \cup \tau^0)$

    $\left[\ell_{\tau,(i)}\right]_{i=1}^d \leftarrow \mathcal{L}_\eta(\hat{\sigma}_\tau^t, \eta, \Psi, \ell)$

    $v_\tau^t \leftarrow -\sum_{i=1}^d \dfrac{\partial \ell_{\tau,(i)}, \pi_{j_{(i)}}}{\partial \hat{\sigma}_\tau^t}$

    // Update regret matching$^+$.

    $\bar{v}_\tau^t \leftarrow (\hat{\sigma}_\tau^t)^\top v_\tau^t$

    $\rho_\tau^t \leftarrow v_\tau^t - \bar{v}_\tau^t$

    $q_\tau^{1:t} \leftarrow [q_\tau^{1:t-1} + \rho_\tau^t]_+$

    // Update the best round.

    if $\bar{v}_\tau^t > \bar{v}^{t^*}$ **then**

      $t^* \leftarrow t$

      $\bar{v}^{t^*} \leftarrow \bar{v}_\tau^t$

return $\tau^{t^*}, \hat{\sigma}_{\tau^*}^{t^*}$

Procedure $\mathcal{L}_\eta$ ***Inputs:*** $\langle \hat{\sigma}, \eta, \Psi, \ell \rangle$

  // The support of Ψ, $\text{supp}(\Psi)$, is assumed to be a finite number $d = |\text{supp}(\Psi)|$.

  for $\pi_{j_i}, \sigma_i \in \text{supp}(\Psi)$ **do**

    // Evaluate $\hat{\sigma}$.

    $\ell_i \leftarrow \ell(\hat{\sigma}; \pi_{j_i}, \sigma_i)$

  $\text{Sort}\Big(\{i \mid \ell_i\}_{i=1}^d\Big)$

  // Assign weights to each loss function.

  // Iterate over Ψ's support sorted accoding to descending loss value from the previous step.

  $\beta \leftarrow 0$

  for $\pi_{j_{(i)}}, \sigma_{(i)} \in \text{supp}(\Psi)$ **do**

    $\alpha_{(i)} = \min\big\{\Psi\big(\langle \pi_{j_{(i)}}, \sigma_{(i)} \rangle\big), \eta - \beta\big\}$

    $\beta \leftarrow \beta + \alpha_{(i)}$

  return $\left[\frac{\alpha_{(i)}}{\eta} \ell_{(i)}\right]_{i=1}^d$

Algorithm 1: Deterministic CVaR(η) RPOSST$_{\text{SEQ}}$ with regret matching$^+$

Inputs: $\langle k, N, T, m, \Psi, \tau^0, \ell \rangle$

// Initialize pseudoregrets.

$q_{\mathcal{T}}^{1:0} \leftarrow \mathbf{0} \in \mathbb{R}^{|\mathcal{T}^m|}$

$q_{\tau}^{1:0} \leftarrow \mathbf{0} \in \mathbb{R}^{m+|\tau^0|}$ **for** $\tau \in \mathcal{T}^m$

// Initialize average distributions.

$\hat{\sigma}_{\mathcal{T}}^{1:0} \leftarrow \mathbf{0} \in \mathbb{R}^{|\mathcal{T}^m|}$

$\hat{\sigma}_{\tau}^{1:0} \leftarrow \mathbf{0} \in \mathbb{R}^{m+|\tau^0|}$ **for** $\tau \in \mathcal{T}^m$

for $t \leftarrow 1, \dots, T$ **do**

  // Sample antagonist actions.

  $\pi_{j_i}, \sigma_i \sim \Psi$ **for** $i = 1 \dots N$

  // Generate test case distributions.

  $z_{\mathcal{T}}^t \leftarrow \mathbf{1}^\top q_{\mathcal{T}}^{1:t-1}$

  $\hat{\sigma}_{\mathcal{T}}^t \leftarrow q_{\mathcal{T}}^{1:t-1} / z_{\mathcal{T}}^t$ **if** $z_{\mathcal{T}}^t > 0$ **else** $\mathbf{1}/|\mathcal{T}^m|$

  for $\tau \in \mathcal{T}^m$ **do**

    $z_{\tau}^t \leftarrow \mathbf{1}^\top q_{\tau}^{1:t-1}$

    $\hat{\sigma}_{\tau}^t \leftarrow q_{\tau}^{1:t-1} / z_{\tau}^t$ **if** $z_{\tau}^t > 0$ **else** $\mathbf{1}/m$

    // Fill in zeros so that $\hat{\sigma}_{\tau}^t \in \triangle^{|\mathcal{T}|}$.

    $\hat{\sigma}_{\tau}^t(x) \leftarrow 0$ **for** $x \in \mathcal{T} \setminus (\tau \cup \tau^0)$

  // Evaluate the CFR$^+$ distributions.

  $\ell_i \leftarrow (\hat{\sigma}_{\mathcal{T}}^t)^\top [\ell(\hat{\sigma}_{\tau}^t; \pi_{j_i}, \sigma_i)]_{\tau \in \mathcal{T}^m}$ **for** $i = 1, \dots, N$

  // Sort to identify the worst k.

  $\mathrm{SortBy}\Big([\langle \sigma_i, \pi_{j_i} \rangle]_{i=1}^N, [\ell_i]_{i=1}^N\Big)$

  // Update CFR$^+$.

  for $\tau \in \mathcal{T}^m$ **do**

    $\ell_{\tau,(i)} \leftarrow \ell(\hat{\sigma}_{\tau}^t; \pi_{j_{(i)}}, \sigma_{(i)})$ **for** $i = 1, \dots, k$

    $v_{\tau}^t \leftarrow \frac{-1}{k} \sum_{i=1}^k \frac{\partial \ell_{\tau,(i)}}{\partial \hat{\sigma}_{\tau}^t}$

    $\rho_{\tau}^t \leftarrow v_{\tau}^t - (\hat{\sigma}_{\tau}^t)^\top v_{\tau}^t$

    $q_{\tau}^{1:t} \leftarrow [q_{\tau}^{1:t-1} + \rho_{\tau}^t]_+$

  $v_{\mathcal{T}}^t \leftarrow \frac{-1}{k} \sum_{i=1}^k \big[\ell_{\tau,(i)}\big]_{\tau \in \mathcal{T}^m}$

  $\rho_{\mathcal{T}}^t \leftarrow v_{\mathcal{T}}^t - (\hat{\sigma}_{\mathcal{T}}^t)^\top v_{\mathcal{T}}^t$

  $q_{\mathcal{T}}^{1:t} \leftarrow [q_{\mathcal{T}}^{1:t-1} + \rho_{\mathcal{T}}^t]_+$

  // Update average distributions.

  $\hat{\sigma}_{\mathcal{T}}^{1:t} \leftarrow \hat{\sigma}_{\mathcal{T}}^{1:t-1} + t\hat{\sigma}_{\mathcal{T}}^t$

  $\hat{\sigma}_{\tau}^{1:t} \leftarrow \hat{\sigma}_{\tau}^{1:t-1} + t\hat{\sigma}_{\mathcal{T}}^t(\tau)\hat{\sigma}_{\tau}^t$ **for** $\tau \in \mathcal{T}^m$

return $\left\langle \frac{\hat{\sigma}_{\mathcal{T}}^{1:T}}{\mathbf{1}^\top \hat{\sigma}^{1:T}}, \left[\frac{\hat{\sigma}_{\tau}^{1:T}}{\mathbf{1}^\top \hat{\sigma}_{\tau}^{1:T}}\right]_{\tau \in \mathcal{T}^m} \right\rangle$

Algorithm 2: RPOSST$_{\text{SIM}}$ (simultaneous model; CFR$^+$)

Table 1: Approximate amount of time required to run $T = 500$ rounds of CVaR(1%) RPOSST$_{\text{SEQ}}$ in each domain. Runtimes are similar across both variants in each domain and across holdout policy set sizes.

domain	runtime / seed
Racing Arrows	$\sim$ 2 minutes
ACPC	$\sim$ 10 seconds
GT	$\sim$ 90 seconds

Table 2: Top two test case pairs and corresponding selection frequencies chosen by each algorithm over the 100 seeds in the large GT experiment.

Algorithm	Pairs	Frequency
$\text{RPOSST}_{\text{SEQ}}$	(41, 16)	90
	(41, 19)	3
Minimax uniform	(41, 16)	40
	(34, 9)	37
Iterative minimax	(39, 32)	87
	(39, 31)	3
Minimax(TNP) uniform	(40, 16)	92
	(36, 16)	4
Miniaverage uniform	(37, 12)	5
	(40, 1)	4
Minimax(TTD) uniform	(43, 6)	4
	(28, 1)	4

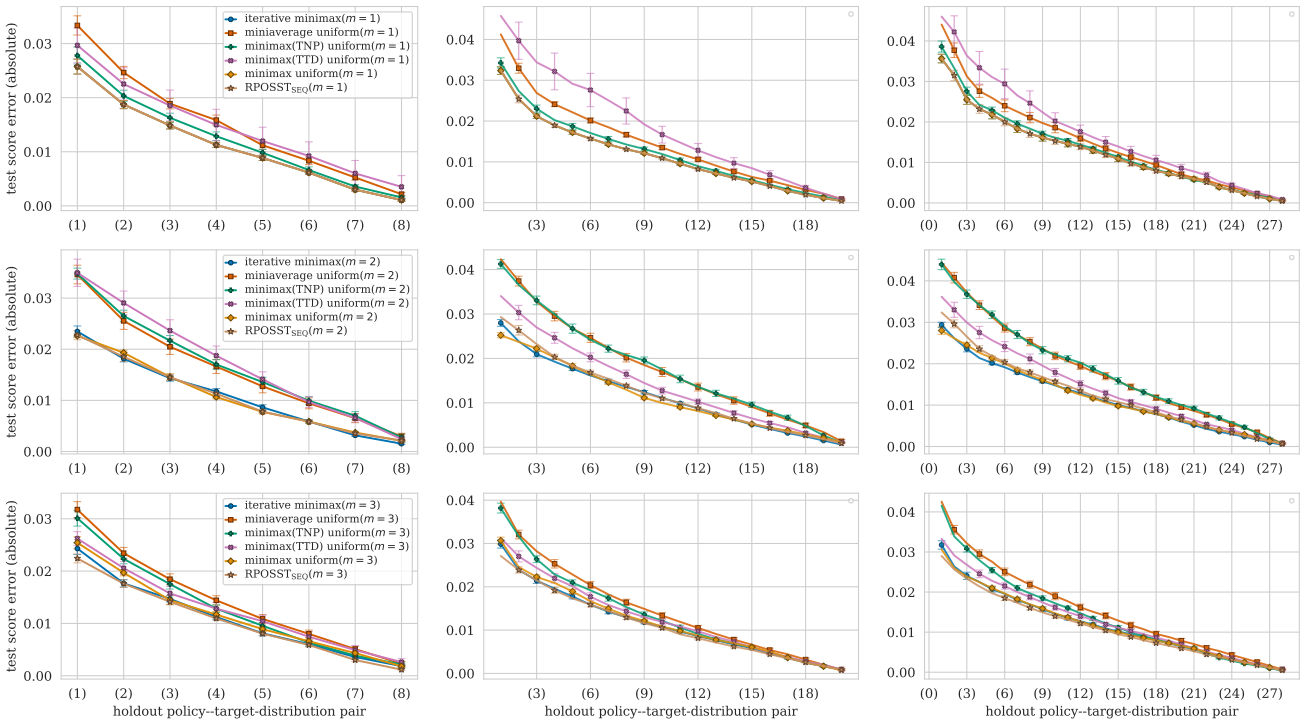


Figure 5: Expected test score error (absolute difference) across holdout-policy–target-distribution pairs on the ACPC 2012 data. Each row uses a different setting for the test size ($m = 1$ top, $m = 2$ middle, and $m = 3$ bottom) and each column uses a different holdout proportion (20% held out in the left column, 40% middle, and 60% right). 100 sets of holdout policies were sampled. Holdout-policy–target-distribution pairs are sorted according to test score error. Each $\text{RPOSST}_{\text{SEQ}}$ instance was run for 500 rounds ($T = 500$). Errorbars represent 95% t-distribution confidence intervals.

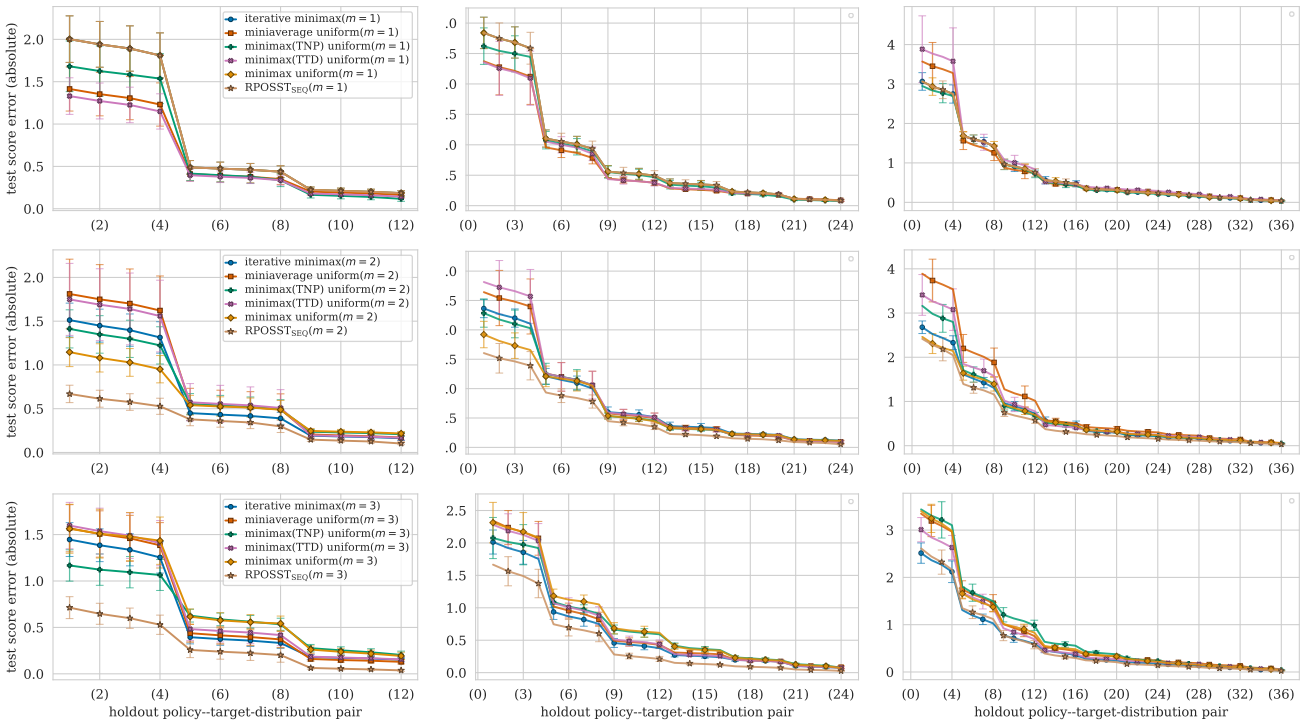


Figure 6: Expected test score error (absolute difference) across holdout-policy–target-distribution pairs on the ACPC 2017 data. Each row uses a different setting for the test size ($m = 1$ top, $m = 2$ middle, and $m = 3$ bottom) and each column uses a different holdout proportion (20% held out in the left column, 40% middle, and 60% right). 100 sets of holdout policies were sampled. Holdout-policy–target-distribution pairs are sorted according to test score error. Each RPOSST$_{\text{SEQ}}$ instance was run for 500 rounds ($T = 500$). Errorbars represent 95% t-distribution confidence intervals.

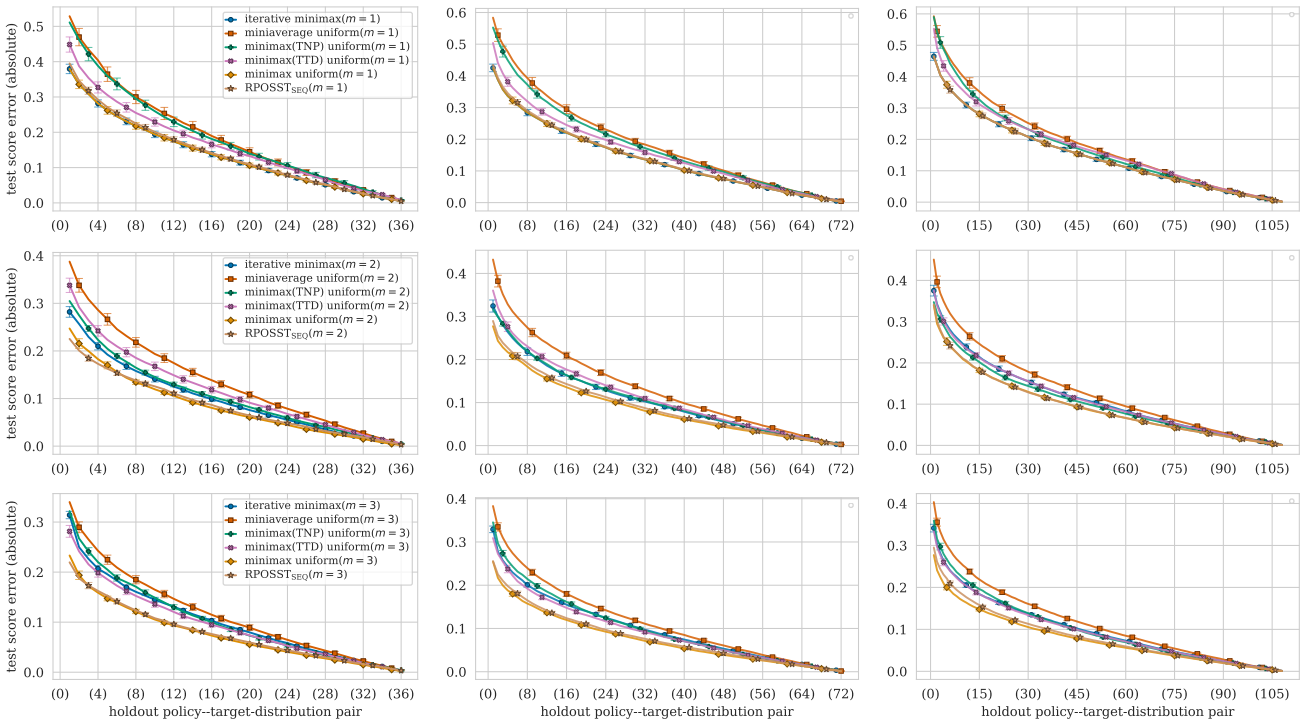


Figure 7: Expected test score error (absolute difference) across holdout-policy–target-distribution pairs in the winrate GT domain. Each row uses a different setting for the test size ($m = 1$ top, $m = 2$ middle, and $m = 3$ bottom) and each column uses a different holdout proportion (20% held out in the left column, 40% middle, and 60% right). 100 sets of holdout policies were sampled. Holdout-policy–target-distribution pairs are sorted according to test score error. Each RPOSST$_{\text{SEQ}}$ instance was run for 500 rounds ($T = 500$). Errorbars represent 95% t-distribution confidence intervals.

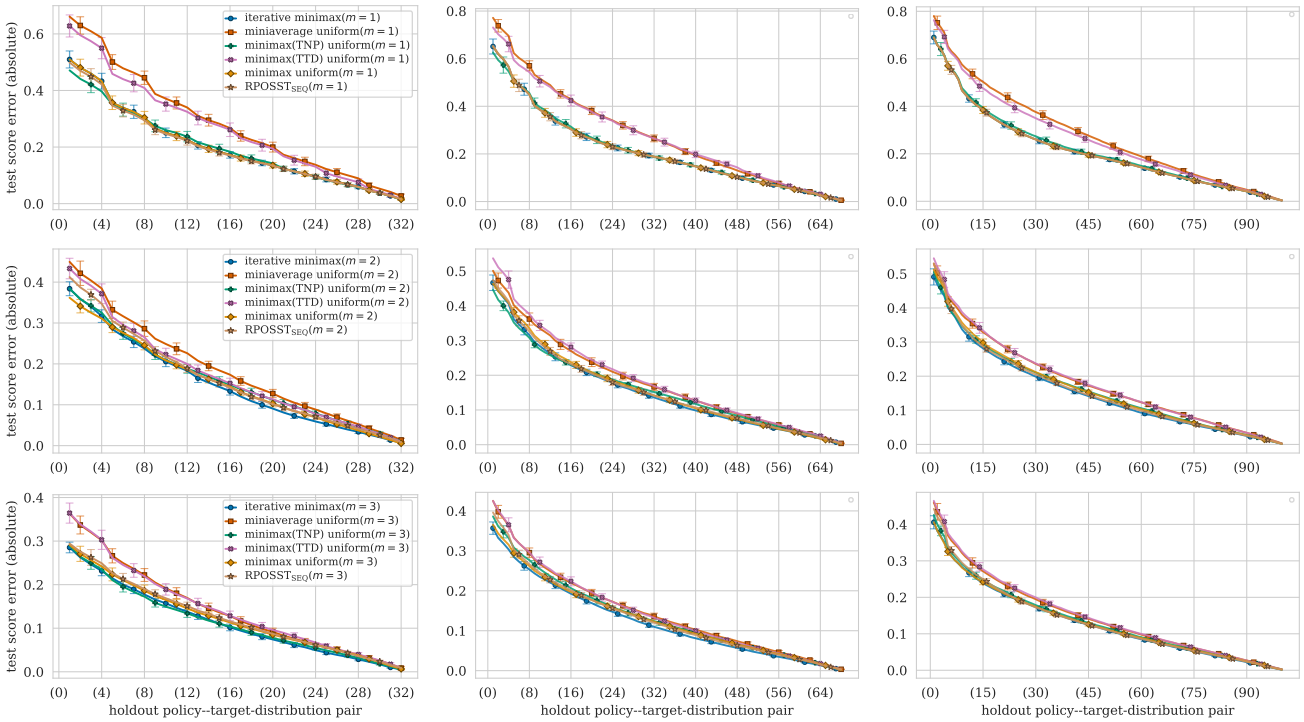
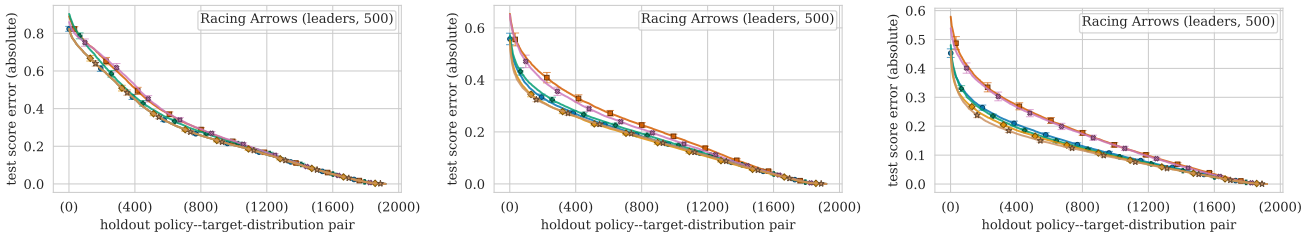


Figure 8: Expected test score error (absolute difference) across holdout-policy–target-distribution pairs in the GT domain where -1 is given for a collision. Each row uses a different setting for the test size ($m = 1$ top, $m = 2$ middle, and $m = 3$ bottom) and each column uses a different holdout proportion (20% held out in the left column, 40% middle, and 60% right). 100 sets of holdout policies were sampled. Holdout-policy–target-distribution pairs are sorted according to test score error. Each RPOSST$_{\text{SEQ}}$ instance was run for 500 rounds ($T = 500$). Errorbars represent 95% t-distribution confidence intervals.



(a) $m = 1$ (b) $m = 2$ (c) $m = 3$

Figure 9: Expected test score error (absolute difference) across holdout-policy–target-distribution pairs on Racing Arrows where test cases are leader policies. Here, 500 Racing Arrows policies were sampled for both the follower and leader role and then 96% of policies of both roles were held out before running RPOSST and each baseline. Each column uses a different setting for the test size ($m = 1$ top, $m = 2$ middle, and $m = 3$ bottom). 100 sets of holdout policies were sampled. Holdout-policy–target-distribution pairs are sorted according to test score error. Each RPOSST$_{\text{SEQ}}$ instance was run for 500 rounds ($T = 500$). Errorbars represent 95% t-distribution confidence intervals.