Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Regrets, I've Had a Few: Approximating (Subgame-Perfect) Nash Equilibria via Deep Regret Actor-Critic

Marvin Bongiovi

marvin.bongiovi@tu-dortmund.de

TU Dortmund University

Research Article

Keywords: Multi-Agent Reinforcement Learning, Regret Minimization, Subgame-Perfect Nash Equilibrium, Actor-Critic

Posted Date: October 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10072911/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Regrets, I've Had a Few: Approximating (Subgame-Perfect) Nash Equilibria via Deep Regret Actor-Critic

Marvin Bongiovi

Business and Economics, Technical University Dortmund, Dortmund, 44221, North Rhine-Westphalia, Germany.

Corresponding author(s). E-mail(s): marvin.bongiovi@tu-dortmund.de;

Abstract

This paper introduces the Deep Regret Actor-Critic (DRAC) algorithm to approximate (Subgame-Perfect) Nash Equilibria in continuous general-sum games. Standard actor-critic methods struggle in these settings, as concurrent utility maximization induces severe non-stationarity. DRAC resolves this by shifting the optimization objective to regret minimization, allowing for a Centralized Training with Centralized Execution architecture that coordinates joint policy updates while enforcing individual rationality to prevent collusion. It employs a Monte Carlo solver that samples unilateral deviations against fixed opponent actions, creating a stationary target landscape. A dual-head Critic maps this stationary landscape by independently predicting the best-response utility and the resulting regret. By minimizing this predicted regret, the Actor drives the joint strategy profile toward an unexploitable equilibrium. Empirical evaluations in a sequential Cournot oligopoly and a proportional prize contest confirm convergence to the analytical equilibria. Compared to Multi-Agent Deep Deterministic Policy Gradient (MADDPG), our algorithm consistently achieves lower exploitability and approximates the analytical equilibrium quantities more precisely.

Keywords: Multi-Agent Reinforcement Learning, Regret Minimization, Subgame-Perfect Nash Equilibrium, Actor-Critic

JEL Classification: C45 , C63 , C72 , C73

1 Introduction

Multi-agent reinforcement learning (MARL) provides a versatile framework for modeling complex interactive tasks (Gronauer & Diepold, 2022). Yet, scaling these methods to continuous action spaces in general-sum games remains a challenge due to the inherent non-stationarity of the learning process. Standard independent reinforcement learning methods typically fail in these settings because agents optimize their individual utilities independently. Since all agents continuously update their strategies, the learning environment becomes a moving target. As emphasized in a recent mathematical overview by Fuente, i Alonso, and Casadellà (2024), this non-stationarity undermines standard convergence guarantees; consequently, rather than converging to a stable equilibrium, agents frequently become trapped in cyclic policy adaptations.

To find such stable equilibria, some approaches successfully rely on regret-based frameworks. While algorithms like Deep Counterfactual Regret Minimization (Deep CFR) have scaled to large state spaces by utilizing Monte Carlo traversals to collect regrets and train neural networks to generalize these values, extending them to continuous spaces remains difficult (Brown, Lerer, Gross, & Sandholm, 2019). Although Deep CFR efficiently samples the game tree, the generation of its neural network targets relies on explicitly evaluating every available action at a given decision node. This requirement restricts the algorithm to finite, discrete domains.

Conversely, while standard Actor-Critic algorithms operate in continuous spaces, their coupled value estimations leave them highly vulnerable to error propagation. Because action-value functions depend on the joint policy, a flawed policy update by one agent inherently alters the action-value function of the others, embedding the error into the learning landscape. As demonstrated in a recent study on auction settings by Graf, Zobernig, Schmidt, and Klöckl (2024), even though continuous control algorithms like Deep Deterministic Policy Gradient can replicate complex equilibria, their empirical success in multi-agent settings remains highly sensitive to hyperparameter choices - such as the size of the replay buffer or the balance of learning rates - and degrades significantly as the number of players increases.

Recent works explore alternative structural approaches to stabilize continuous general-sum games. For instance, Near-Potential Policy Optimization (Kalanther, Bharvirkar, Sastry, & Maheshwari, 2026) addresses non-stationarity by projecting competitive dynamics onto a shared, local potential function. Assuming an infinite horizon, NePPO maximizes this proxy to steer the system toward an α-approximate Nash Equilibrium. However, the potential-based objective optimizes for approximate Nash equilibria without ensuring sequential rationality. Furthermore, its reliance on nested optimization - utilizing inner reinforcement learning solvers and zeroth-order updates - limits its scalability.

In contrast, methods that explicitly target sequential rationality typically require highly specialized training paradigms. For example, Temporal-Induced Self-Play (TISP) achieves an ϵ-Perfect Bayesian Equilibrium in two-player stochastic games by explicitly approximating recursive, backward-looking reasoning (Chen, Zhou, Wu, & Fang, 2021). However, this is accomplished by training the policies in reverse order, starting from the final round and working backward.

To address these limitations, we introduce the Deep Regret Actor-Critic (DRAC) algorithm. DRAC mitigates non-convergence by shifting the optimization objective from independent utility maximization to regret minimization. Structurally, this shift allows DRAC to operate within a Centralized Training with Centralized Execution (CTCE) paradigm. While applying CTCE in non-cooperative settings typically induces collusion via joint reward maximization, jointly minimizing regret strictly enforces individual rationality, enabling coordinated policy updates without compromising the (Subgame-Perfect) Nash Equilibrium. By definition, such an equilibrium is a joint policy under which no player can improve their payoff through unilateral deviation in any given state, guaranteeing an individual regret - and thus exploitability - of zero. Since identifying the exact Best Response is computationally intractable in continuous domains, DRAC approximates it by applying Monte Carlo sampling directly to the action space. For each agent, the algorithm samples candidate unilateral deviations, while holding the actions of all other players fixed. By evaluating this sampled subset of continuous deviations, DRAC approximates the overall regret surface without requiring exhaustive action evaluation.

Evaluating these unilateral deviations against fixed opponent actions serves two critical functions. First, it reduces the search space complexity from an exponential $\mathcal{O}(|\mathcal{A}|^N)$ to a linear $\mathcal{O}(|\mathcal{A}|\cdot N)$, where N is the number of players. Second, it structurally stabilizes the target estimation process, mitigating the cascade of the approximation errors. As noted, in standard MARL, a flawed policy update propagates errors into the action-value functions of other agents. DRAC addresses this by computing regret targets strictly against the fixed actions of a sampled transition, decoupling the immediate evaluation from the opponents' concurrent policy updates. Therefore, even if a flawed update temporarily alters opponent behavior, the underlying regret evaluations remain consistent. This isolation limits approximation errors from cascading and promotes a more stationary learning landscape.

Ultimately, the Actor utilizes this approximated regret surface to learn a joint strategy profile. By systematically minimizing the predicted regret, DRAC reduces the overall exploitability of the joint policy. Because this structural isolation reduces the accumulation of approximation errors, the algorithm allows for a more reliable convergence to a Nash Equilibrium in continuous normal-form games, while also providing a stabilizing learning signal to approximate a Subgame-Perfect Nash Equilibrium in dynamic games.

2 Description of Deep Regret Actor-Critic

This section outlines the algorithm. DRAC is designed to approximate Nash Equilibria in normal-form games and Subgame-Perfect Nash Equilibria (SPNE) in dynamic games, operating under perfect information and general-sum payoff structures. Although we assume perfect information, restricting network inputs to individual observations offers a natural extension to imperfect information settings, as the replay buffer's empirical distribution implicitly resolves information sets. In such decentralized settings, independent actors can be jointly optimized by minimizing the sum of their predicted regrets, ensuring coordinated policy updates. Furthermore, while our

discussion focuses on pure strategies in continuous action spaces, the algorithm can be readily adapted for discrete domains to find mixed equilibria. In such cases, it outputs action probabilities and calculates regret based on the expected values of these mixed profiles.

To achieve this, DRAC employs an Actor-Critic architecture, specifically utilizing a dual-head Critic. We begin by detailing these two components:

2.1 The Actor

Given a state space $\mathcal{S}$, an individual action space $\mathcal{A}$, and a total of N agents, the Actor is modeled as a neural network $\pi_\theta : \mathcal{S} \to \mathcal{A}^N$, parameterized by weights θ, which outputs a joint action vector $\mathbf{a}$. Each component $\pi_{\theta,i}(s)$ represents the action of agent $i \in \{1, \ldots, N\}$ in state $s \in \mathcal{S}$:

$$\mathbf{a} = \pi_\theta(s). \tag{1}$$

2.2 The (Dual-Head) Critic

The Critic, modeled as a neural network C_ϕ parameterized by weights ϕ, evaluates the strategic quality of the joint action via two distinct output heads. Let $\mathbf{a}_{-i}$ denote the joint action of all players except agent i.

1. **V-Head:** Predicts the maximum utility agent i could achieve if they were to play optimally against the fixed actions of the other agents $\mathbf{a}_{-i}$.

$$\hat{V}_i(s, \mathbf{a}_{-i}) \approx \max_{a'_i \in \mathcal{A}} Q_i(s, a'_i, \mathbf{a}_{-i}), \tag{2}$$

   where Q_i represents the true state-action value function for agent i, and a'_i is a unilateral action deviation. Note that this estimated best response value is invariant to the specific action a_i played, as it represents this agent's maximum payoff in this state given the other agents' actions. Excluding a_i from the V-Head ensures the network independently estimates the best-response value, reserving a_i strictly for the regret calculation.
2. **R-Head:** Predicts the difference between the best-response utility and the utility of the action actually performed:

$$\hat{R}_i(s, \mathbf{a}) \approx \hat{V}_i(s, \mathbf{a}_{-i}) - Q_i(s, a_i, \mathbf{a}_{-i}). \tag{3}$$

3 Learning Process

The learning process of DRAC alternates between data collection via environmental interaction, target estimation, and network optimization. The complete training loop is summarized in Algorithm 1.

3.1 Data Collection and Exploration

Before optimization can occur, the agents interact with the environment to populate a shared replay buffer $\mathcal{D}$. At each time step, given the current state s, the agents compute a joint action using the current Actor policy $\pi_\theta(s)$. For continuous exploration, stochastic Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2_{\text{expl}} I)$ is added, yielding the executed joint action $\mathbf{a} = \pi_\theta(s) + \epsilon$. The variance σ^2_{expl} is annealed over time to focus sampling on the vicinity of the equilibrium.

The environment executes this perturbed action $\mathbf{a}$, yielding a reward vector $\mathbf{r}$ and the subsequent state s'. The complete transition tuple $(s, \mathbf{a}, \mathbf{r}, s')$ is then stored in the replay buffer $\mathcal{D}$.

3.2 Monte Carlo Solver Step: Target Generation

To generate training targets for the dual-head Critic, DRAC performs a (localized) Monte Carlo search over the action space for each agent i using a sampled batch from the replay buffer:

1. **Opponent Fixation:** Hold the joint action of all other agents, $\mathbf{a}_{-i}$ strictly constant.
2. **Deviation Sampling:** Generate K candidate unilateral deviations $\{\tilde{a}^{(k)}\}_{k=1}^{K}$. Typically, these are sampled from a Gaussian distribution centered on the agent's executed action a_i with variance σ^2_{MC}, i.e., $\tilde{a}^{(k)} \sim \mathcal{N}(a_i, \sigma^2_{\text{MC}} I)$, to focus the search locally. Alternatively, deviations can be drawn from a uniform distribution over the entire action space $\mathcal{A}$ for global exploration. For low-dimensional action spaces, a deterministic grid search can be employed. Regardless of the generation method, all samples are strictly bounded to the limits of $\mathcal{A}$.
3. **Utility Evaluation:** Compute the utility $u_i^{(k)}$ for each sampled deviation. In one-shot games or the terminal round of a dynamic game, this utility equals the immediate environmental reward. In sequential settings, the utility is defined as the sum of the immediate reward and the discounted value of the subsequent state, estimated via the Target Networks ($\pi_{\theta'}$ and $C_{\phi'}$).
4. **Target Construction & Rectification:** First, the approximate best-response utility y_i^V is identified as the maximum over the sampled deviations:

$$y_i^V = \max_{k \in \{1,\dots,K\}} u_i^{(k)}. \tag{4}$$

Subsequently, the target regret y_i^R is computed. Because the MC estimator approximates a continuous space with finite samples, the sampled maximum may underestimate the value of the best response, falling below the utility of the executed action. To prevent learning negative regrets, we apply a rectification step using the target action-value $\tilde{Q}_i$:

$$y_i^R = \max\left(0,\, y_i^V - \tilde{Q}_i\right), \tag{5}$$

where $\tilde{Q}_i = r_i + \gamma \hat{V}_{\phi',i}(s', \pi_{\theta'}(s')_{-i})$ represents the state-action value evaluated via the target networks.

3.3 Network Optimization

Following the target generation, the networks are optimized using a sampled batch of size B from the replay buffer.

Critic Update:

To update the Critic parameters ϕ, standard losses such as the Mean Squared Error are applicable. However, because Monte Carlo estimations can exhibit high variance, we recommend employing the Huber Loss $\mathcal{H}_\delta$ in practice. By transitioning from a quadratic penalty for small errors to a linear penalty for large errors, it stabilizes gradient updates against outliers:

$$\mathcal{H}_\delta(e) = \begin{cases} \frac{1}{2}e^2 & \text{for } |e| \leq \delta, \\ \delta \cdot (|e| - \frac{1}{2}\delta) & \text{otherwise.} \end{cases} \tag{6}$$

The empirical Critic loss aggregates the estimation errors of the online network C_ϕ across both the V-Head and the R-Head for all N agents, averaged over the mini-batch of size B:

$$\mathcal{L}(\phi) = \frac{1}{B} \sum_{b=1}^{B} \sum_{i=1}^{N} \left(\mathcal{H}_\delta(y_{i,b}^V - \hat{V}_{\phi,i,b}) + \mathcal{H}_\delta(y_{i,b}^R - \hat{R}_{\phi,i,b}) \right), \tag{7}$$

where $y_{i,b}^V, y_{i,b}^R$ are the generated targets and $\hat{V}_{\phi,i,b}, \hat{R}_{\phi,i,b}$ denote the online network predictions for the b-th sample in the batch.

Actor Update:

Once the Critic is updated, the Actor parameters θ are optimized by minimizing the sum of predicted individual regrets via the online Critic C_ϕ. While maximizing a shared objective typically induces collusion in multi-agent settings, minimizing total regret avoids this issue. Since the individual regrets are lower-bounded by zero, minimizing their sum forces each agent's individual exploitability toward zero. This joint update directly steers the policy π_θ toward an equilibrium state:

$$\mathcal{L}(\theta) = \frac{1}{B} \sum_{b=1}^{B} \sum_{i=1}^{N} \hat{R}_{\phi,i}(s_b, \pi_\theta(s_b)). \tag{8}$$

Target Network Update:

Following the parameter optimization, the weights of the target networks are updated via an exponential moving average using a soft-update rate $\tau \in (0, 1]$:

$$\theta' \leftarrow \tau\theta + (1 - \tau)\theta', \tag{9}$$

$$\phi' \leftarrow \tau\phi + (1 - \tau)\phi'. \tag{10}$$

Algorithm 1 Deep Regret Actor-Critic (DRAC)

Require: Replay Buffer $\mathcal{D}$, Batch size B, MC Sample size K
Require: Variances σ^2_{expl} (exploration) and σ^2_{MC} (MC search)
Require: Learning rates η_C, η_A, Soft-update rate τ, Discount factor γ

Initialize Actor π_θ and dual-head Critic C_ϕ (with heads $\hat{V}, \hat{R}$)
Initialize Target Networks $\pi_{\theta'} \leftarrow \pi_\theta$, $C_{\phi'} \leftarrow C_\phi$
while training **do**
    // 1. Data Collection
    Observe state s
    Execute joint action $\mathbf{a} = \pi_\theta(s) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2_{\text{expl}} I)$
    Observe reward r and next state s', store transition $(s, \mathbf{a}, r, s')$ in $\mathcal{D}$
    Anneal σ^2_{expl} over time to focus sampling
    // 2. Target Generation (Mini-Batch)
    Sample a mini-batch $\{(s_b, \mathbf{a}_b, r_b, s'_b)\}_{b=1}^{B}$ from $\mathcal{D}$
    for each agent $i \in \{1, \ldots, N\}$ **do**
        Fix opponents' actions $\mathbf{a}_{-i,b}$
        Sample K deviations $\{\tilde{a}^{(k)}_{i,b}\}_{k=1}^{K}$ (e.g., via $\mathcal{N}(a_{i,b}, \sigma^2_{\text{MC}} I)$), bounded to $\mathcal{A}$
        Evaluate utilities $u^{(k)}_{i,b}$ via immediate reward or Target Networks $\pi_{\theta'}, C_{\phi'}$
        $y^V_{i,b} \leftarrow \max_{k \in \{1,\ldots,K\}} u^{(k)}_{i,b}$
        $\tilde{Q}_{i,b} \leftarrow r_{i,b} + \gamma \hat{V}_{\phi',i}(s'_b, \pi_{\theta'}(s'_b)_{-i})$
        $y^R_{i,b} \leftarrow \max\left(0,\ y^V_{i,b} - \tilde{Q}_{i,b}\right)$
    end for
    // 3. Network Optimization
    Compute Critic loss $\mathcal{L}(\phi) = \frac{1}{B} \sum_{b=1}^{B} \sum_{i=1}^{N} \left(\mathcal{H}_\delta(y^V_{i,b} - \hat{V}_{\phi,i,b}) + \mathcal{H}_\delta(y^R_{i,b} - \hat{R}_{\phi,i,b}) \right)$
    Update Critic parameters: $\phi \leftarrow \phi - \eta_C \nabla_\phi \mathcal{L}(\phi)$
    Compute Actor loss $\mathcal{L}(\theta) = \frac{1}{B} \sum_{b=1}^{B} \sum_{i=1}^{N} \hat{R}_{\phi,i}(s_b, \pi_\theta(s_b))$
    Update Actor parameters: $\theta \leftarrow \theta - \eta_A \nabla_\theta \mathcal{L}(\theta)$
    // 4. Target Network Update
    $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$
    $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$
end while

4 Experiment I: Sequential Cournot Oligopoly

To establish a baseline in a dynamic setting, we employ a repeated N-player Cournot Oligopoly over T rounds. In this classic economic game, independent firms compete by simultaneously choosing the production quantity of a homogeneous good in each time step.

State and Action Space:

To test the networks' capacity to generalize across different environments, we define the state space at time step t as $s_t = (\alpha, \beta, t)$. At the beginning of each episode, the market base price α and the price sensitivity β are sampled from uniform distributions

defined by their respective lower and upper bounds, $\alpha \sim \mathcal{U}(\alpha_{\min}, \alpha_{\max})$ and $\beta \sim \mathcal{U}(\beta_{\min}, \beta_{\max})$, and remain static throughout the episode. The continuous action space for each agent $i \in \{1, \ldots, N\}$ corresponds to its chosen production quantity $a_{i,t} = q_{i,t} \in [0, q_{\max}]$, bounded by the maximum capacity $q_{\max}$.

Reward Structure:

The market price is determined by the state-dependent linear inverse demand function $P(Q_t \mid s_t) = \alpha - \beta Q_t$, where $Q_t = \sum_{j=1}^{N} q_{j,t}$ is the aggregate market quantity. Without loss of generality, we assume zero marginal production costs, as any constant cost can be absorbed into the base price α. Thus, the immediate reward for agent i in state s_t, given the joint action profile $\mathbf{a}_t = (q_{1,t}, \ldots, q_{N,t})$, is its profit:

$$r_i(\mathbf{a}_t \mid s_t) = q_{i,t} \left(\alpha - \beta \sum_{j=1}^{N} q_{j,t} \right). \tag{11}$$

Evaluation Metrics:

Because the market parameters are static, solving the first-order conditions yields a unique, state-conditional and symmetric SPNE where agents play the Nash quantity in every round:

$$q^*(s_t) = \frac{\alpha}{\beta(N+1)}. \tag{12}$$

Because high empirical rewards may indicate collusion rather than the SPNE, cumulative returns are an insufficient metric. Instead, we use the analytical optimum to evaluate the learned joint policy across two dimensions:

1. **Mean Relative Absolute Deviation (MRAD):** To measure convergence to the equilibrium quantities, we calculate the scale-invariant distance to the theoretical optimal quantity:

$$\mathrm{MRAD}(\mathbf{a}_t, s_t) = \frac{1}{N} \sum_{i=1}^{N} \frac{|q_{i,t} - q^*(s_t)|}{q^*(s_t)}. \tag{13}$$

2. **Exploitability:** Because the objective of the individual agents is regret minimization, we evaluate the global convergence of the learned joint policy via its analytical exploitability. For any joint action profile $\mathbf{a}_t$, let $\mathbf{a}_{-i,t}$ denote the action vector of all agents except i, with their aggregate quantity defined as $Q_{-i,t} = \sum_{j \neq i} q_{j,t}$. The Best Response for agent i against the fixed actions of its opponents in state s_t is derived from the first-order condition as:

$$\mathrm{BR}_i(\mathbf{a}_{-i,t} \mid s_t) = \max \left(0, \min \left(q_{\max}, \frac{\alpha - \beta Q_{-i,t}}{2\beta} \right) \right). \tag{14}$$

   Let $Q_i^*(\mathbf{a}_t \mid s_t)$ denote the optimal state-action value under SPNE continuation. The exploitability of the joint policy is then defined as the sum of regrets across all

agents in state s_t:

$$\text{Exploitability}(\mathbf{a}_t, s_t) = \sum_{i=1}^{N} \Big(Q_i^*(\text{BR}_i(\mathbf{a}_{-i,t} \mid s_t), \mathbf{a}_{-i,t} \mid s_t) - Q_i^*(\mathbf{a}_t \mid s_t) \Big). \quad (15)$$

Experimental Setup and Results:

For the continuous Cournot oligopoly experiments, we simulated a repeated market with $N = 3$ agents over $T = 5$ rounds, where individual production capacities were bounded by $q_{i,t} \in [0, 2.0]$. At the beginning of each episode, market parameters were uniformly sampled: base price $\alpha \sim \mathcal{U}(1.0, 2.0)$ and price sensitivity $\beta \sim \mathcal{U}(0.5, 2.0)$. To establish a baseline, we trained the DRAC algorithm alongside MADDPG (Lowe et al., 2017) over 200,000 epochs, periodically averaging the evaluation metrics across a fixed grid of 500 equidistant state configurations. A complete list of hyperparameters is provided in Table A1. As illustrated in Figure 1, DRAC's exploitability converged asymptotically to 1.37×10^{-4}, while the MRAD reached 0.011, demonstrating a mere 1.1% mean deviation from the analytical SPNE quantities. MADDPG achieved an MRAD of 0.019 (1.9%) and an exploitability of 4.27×10^{-4}. Interestingly, although both algorithms closely approximate the equilibrium quantities, DRAC achieves consistently lower exploitability alongside a lower MRAD. This indicates that explicitly learning regret provides a more stable learning signal for approximating equilibria than independent utility maximization. The raw experimental results are provided in Bongiovi (2026).

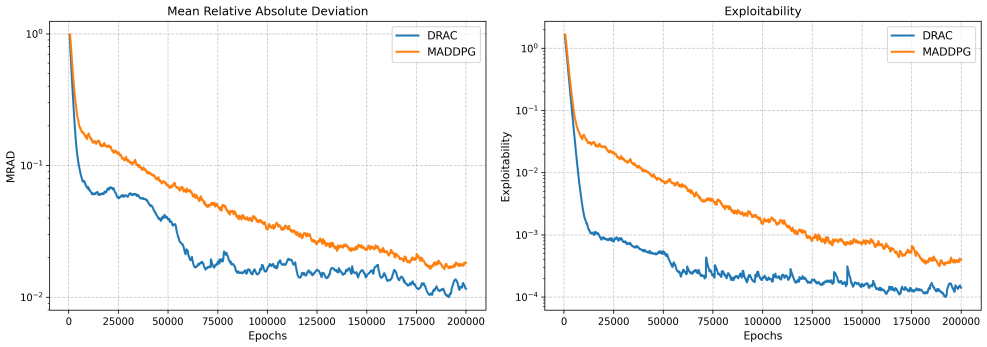

Fig. 1 Convergence of DRAC and MADDPG algorithms in the Sequential Cournot Oligopoly environment over 200,000 training epochs. The graphs display the exponential moving average of the MRAD (left) and the exploitability (right).

5 Experiment II: Sequential Proportional Prize Contest

To evaluate sequential rationality, we formulate a dynamic resource allocation problem as a sequential proportional prize contest, adapting the theoretical framework of Sela and Erez (2013), which guarantees a unique, pure-strategy SPNE. This model incorporates a resource depletion factor $\alpha \in [0, 1]$, where $0 \leq \alpha \leq 1$ represents the rate at which the resource budget decreases proportionally to the resources allocated in each stage.

In this environment, two agents compete over a fixed number of rounds T, each starting with an initial budget $B_{i,1} = 1$. In every round t, players allocate a portion of their remaining budget. We denote this absolute resource allocation as $b_{i,t}$, which is strictly constrained by their available budget $B_{i,t}$. The immediate reward for player i, given the joint allocation profile $\mathbf{b}_t = (b_{1,t}, b_{2,t})$, is its proportional share of a normalized prize $V = 1$:

$$r_{i,t}(\mathbf{b}_t) = \frac{b_{i,t}}{b_{1,t} + b_{2,t}}. \tag{16}$$

If the joint allocation profile is the zero vector ($\mathbf{b}_t = \mathbf{0}$), the prize is divided equally, yielding $r_{i,t} = 1/2$. Budgets update according to $B_{i,t+1} = B_{i,t} - \alpha b_{i,t}$.

State and Action Space:

For a two-player game over T rounds, the state at time step t is $s_t = (B_{1,t}, B_{2,t}, t)$. To maintain fixed action bounds across different budgets, the continuous action for agent i is defined as the budget fraction $a_{i,t} \in [0, 1]$. The absolute allocation is then computed as $b_{i,t} = a_{i,t} B_{i,t}$.

Evaluation Metrics:

Following Sela and Erez (2013), solving the two-player multi-stage game via backward induction yields a unique SPNE. For the condition $\frac{1}{T-t+1} \leq \alpha < 1$, the optimal strategy dictates that an agent allocates a fraction of its current budget adjusted by the depletion factor:

$$b^*_{i,t}(s_t) = \frac{B_{i,t}}{(T - t + 1)\alpha}. \tag{17}$$

Since unspent budget holds no utility after round T, rational players exhaust their entire remaining budget in the final stage. To analyze equilibrium convergence, we evaluate the learned allocations $\mathbf{b}_t = (b_{1,t}, b_{2,t})$ against the analytical SPNE for a given reachable state s_t along two dimensions:

1. **Mean Relative Absolute Deviation (MRAD):** We quantify the scale-invariant distance to the analytical equilibrium:

$$\mathrm{MRAD}(s_t, \mathbf{b}_t) = \frac{1}{2} \sum_{i=1}^{2} \frac{|b_{i,t} - b^*_{i,t}(s_t)|}{b^*_{i,t}(s_t)}. \tag{18}$$

2. **Exploitability:** Given the opponent's allocation $b_{-i,t}$, the analytical Best Response BR_i in state $s_t = (B_{i,t}, B_{-i,t}, t)$ follows directly from the contest's first-order conditions:

$$\mathrm{BR}_i(b_{-i,t} \mid s_t) = \max\left(0, \min\left(B_{i,t}, \frac{(B_{i,t} + B_{-i,t})\sqrt{b_{-i,t}}}{\alpha\sqrt{b_{-i,t}} + \sqrt{(T-t)\alpha(B_{-i,t} - \alpha b_{-i,t})}} - b_{-i,t}\right)\right). \tag{19}$$

Let $Q_i^*(b_{i,t}, b_{-i,t} \mid s_t)$ denote the optimal state-action value under SPNE continuation. We quantify the exploitability of the learned policy by computing the regret for both agents in state s_t:

$$\text{Exploitability}(s_t, \mathbf{b}_t) = \sum_{i=1}^{2} \Big(Q_i^*(\mathrm{BR}_i(b_{-i,t} \mid s_t), b_{-i,t} \mid s_t) - Q_i^*(\mathbf{b}_t \mid s_t) \Big). \tag{20}$$

Experimental Setup and Results:

We simulated a proportional prize contest with $N = 2$ agents over $T = 5$ stages, using a resource depletion factor $\alpha = 0.8$. The DRAC algorithm was trained alongside MADDPG (Lowe et al., 2017) over 200,000 epochs, with evaluation metrics periodically averaged across a fixed grid of 1601 reachable state configurations. A complete list of hyperparameters is provided in Table A2. As illustrated in Figure 2, DRAC's exploitability converged asymptotically to 7.45×10^{-4}, while the MRAD reached 0.030, demonstrating a 3.0% mean deviation from the analytical SPNE quantities. MADDPG achieved an MRAD of 0.063 (6.3%) and an exploitability of 2.48×10^{-3}. Consistent with the Cournot experiments, explicitly targeting regret yields a more precise approximation of the equilibrium and consistently lower exploitability than independent utility maximization. The raw experimental results are provided in Bongiovi (2026).

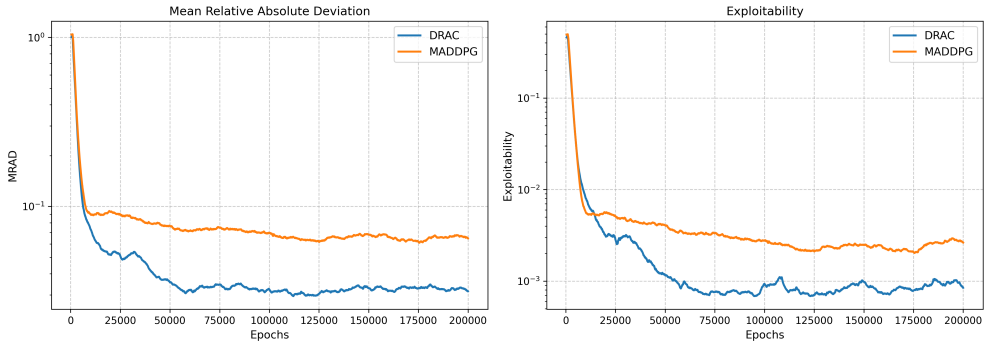

Fig. 2 Convergence of DRAC and MADDPG algorithms in the Sequential Proportional Prize Contest environment over 200,000 training epochs. The graphs display the exponential moving average of the MRAD (left) and the exploitability (right).

6 Discussion and Conclusion

In this paper, we introduced DRAC to address the challenges of non-stationarity and error propagation that typically hinder continuous multi-agent reinforcement learning from converging in general-sum games. Structurally, the algorithm operates within a Centralized Training with Centralized Execution (CTCE) paradigm. While applying CTCE in non-cooperative settings typically induces collusion via joint reward maximization, DRAC's regret objective strictly enforces individual rationality. Thus, the algorithm allows joint policies to be optimized cooperatively without compromising the non-cooperative (Subgame-Perfect) Nash Equilibrium.

In the continuous Cournot Oligopoly, DRAC prevented the cyclic adaptations typical of independent utility maximization, achieving stable, non-oscillatory convergence. In the Sequential Proportional Prize Contest, it approximated the SPNE without specialized training procedures. Across both environments, the algorithm achieved exploitability on the order of 10^{-4} and mean relative absolute deviations of $\leq 3\%$, demonstrating that DRAC can effectively learn sequential rationality in continuous domains.

While effective, DRAC introduces specific practical constraints. Approximating the Best Response via sampling is efficient and aligns well with the relatively smooth payoff landscapes of our Cournot and contest environments. In highly non-convex games, this search is more prone to overlook beneficial global deviations. Furthermore, scaling to higher-dimensional action spaces requires more Monte Carlo samples - yet the computational cost remains linear, rather than exponential, with respect to the number of players. Finally, computing regret requires simulating alternative actions to evaluate their hypothetical rewards. While trivial in a simulator, applying DRAC in black-box environments would require an environment model to predict these outcomes.

Future work will extend this framework beyond continuous, perfect-information domains. Restricting network inputs to individual observations provides a natural transition to Centralized Training with Decentralized Execution (CTDE) for imperfect-information games. In such architectures, jointly minimizing the sum of individual regrets yields a unified learning signal that coordinates simultaneous policy updates, bypassing the oscillatory traps of independent learning while preventing collusion. Finally, transitioning the Actor to output categorical probability distributions will enable the computation of mixed-strategy (Subgame-Perfect) Nash equilibria in discrete games.

Declarations

Declaration of Funding

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Declaration of competing interests

The author has no relevant financial or non-financial interests to disclose.

Data availability

The datasets generated and analysed during the current study are available in the author's GitHub repository (Bongiovi, 2026).

References

Bongiovi, M. (2026). *Regrets, i've had a few: Approximating (subgame-perfect) nash equilibria via deep regret actor-critic: Experimental results.* Retrieved from https://github.com/MarvinBongiovi/Regrets-I-ve-Had-a-Few-Approximating-Subgame-Perfect-Nash-Equilibria-via-Deep-Regret-Actor-Critic

Brown, N., Lerer, A., Gross, S., Sandholm, T. (2019, 09–15 Jun). Deep counterfactual regret minimization. K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (Vol. 97, pp. 793–802). PMLR. (doi: https://doi.org/10.48550/arXiv.1811.00164)

Chen, W., Zhou, Z., Wu, Y., Fang, F. (2021, August). Temporal induced self-play for stochastic bayesian games. *Proceedings of the thirtieth international joint conference on artificial intelligence* (p. 96–103). International Joint Conferences on Artificial Intelligence Organization. (doi: https://doi.org/10.24963/ijcai.2021/14)

Fuente, N.D.L., i Alonso, M.N., Casadellà, G. (2024). *Game theory and multi-agent reinforcement learning : From nash equilibria to evolutionary dynamics.* (doi: https://doi.org/10.48550/arXiv.2412.20523)

Graf, C., Zobernig, V., Schmidt, J., Klöckl, C. (2024). Computational performance of deep reinforcement learning to find nash equilibria. *Computational Economics*, *63*(2), 529–576, (doi: https://doi.org/10.1007/s10614-023-10360-z)

Gronauer, S., & Diepold, K. (2022). Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, *55*(2), 895–943, (doi: https://doi.org/10.1007/s10462-021-09996-w)

Kalanther, A., Bharvirkar, S., Sastry, S., Maheshwari, C. (2026). *Neppo: Near-potential policy optimization for general-sum multi-agent reinforcement learning.* (doi: https://doi.org/10.48550/arXiv.2603.06977)

Lowe, R., WU, Y., Tamar, A., Harb, J., Pieter Abbeel, O., Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. I. Guyon et al. (Eds.), *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc. (doi: https://dl.acm.org/doi/proceedings/10.5555/3295222)

Sela, A., & Erez, E. (2013). Dynamic contests with resource constraints. *Social Choice and Welfare*, *41*(4), 863–882, (doi: https://doi.org/10.1007/s00355-012-0711-1)

Appendix A Experimental Hyperparameters

Here we list the complete hyperparameter configurations for both experiments.

Table A1 Hyperparameters for the Sequential Cournot Oligopoly experiment. Where applicable, MADDPG utilizes the same hyperparameters.

Hyperparameter	**Symbol**	**Value**
Environment & Training		
Number of agents	N	3
Number of rounds	T	5
Training epochs	-	$200,000$
Network Optimization		
Actor learning rate	η_A	10^{-3}
Critic learning rate	η_C	10^{-3}
Batch size	B	128
Replay buffer capacity	$\lvert\mathcal{D}\rvert$	$25,000$
Soft-update rate	τ	0.005
Exploration		
Initial noise scale	σ_{expl}	0.25
Noise decay rate	-	10^{-5}
DRAC-Specific Parameters		
MC deviation samples	K	256
MC sampling std. dev.	σ_{MC}	0.25
MC sampling distribution	-	Truncated Normal

Table A2 Hyperparameters for the Sequential Proportional Prize Contest experiment. Where applicable, MADDPG utilizes the same hyperparameters.

Hyperparameter	**Symbol**	**Value**
Environment & Training		
Number of agents	N	2
Number of rounds	T	5
Resource depletion factor	α	0.8
Training epochs	-	$200,000$
Network Optimization		
Actor learning rate	η_A	10^{-3}
Critic learning rate	η_C	10^{-3}
Batch size	B	128
Replay buffer capacity	$\lvert\mathcal{D}\rvert$	$25,000$
Soft-update rate	τ	0.005
Exploration		
Initial noise scale	σ_{expl}	0.5
Noise decay rate	-	5×10^{-6}
DRAC-Specific Parameters		
MC deviation samples	K	256
MC sampling std. dev.	σ_{MC}	0.33
MC sampling distribution	-	Truncated Normal