POLICY AS DATA: REPLAY-BASED POLICY DUAL AVERAGING VIA ADVANTAGE REGRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Actor-critic methods reuse past experience to improve sample efficiency. However, historical data are typically regarded as off-policy samples for the current policy-improvement update. This work introduces Regularized Dual Averaging Actor Critic (RDA2C), which assigns a distinct role to replay. In regularized dual averaging, the subsequent policy is determined by accumulated policy-improvement feedback, so historical advantage estimates contribute directly to the actor objective rather than solely to the most recent update. RDA2C stores state-action samples with critic-estimated advantage labels, fits a dual score model Z_θ to the aggregated dataset, and derives the current policy from the accumulated score model using the entropy mirror map. In this way, replay defines an empirical dual objective from which the policy is computed. To analyze RDA2C, we establish a finite-time value-gap decomposition, separating the regularized dual-averaging term from errors due to stale-replay supervised fitting, critic bias, finite-buffer variance, and replay coverage, and stating the assumptions under which each error is bounded. RDA2C accepts advantage labels from any critic. With GAE labels, RDA2C outperforms PPO on six of eight MuJoCo tasks and eight of twelve Atari games. RDA2C also outperforms AAPDA, the closest dual-averaging baseline, on six of eight MuJoCo tasks. With twin-Q labels, RDA2C matches SAC at matched batch size and update frequency.

1 INTRODUCTION

Experience replay is an important mechanism for improving sample efficiency in deep reinforcement learning by allowing previously collected transitions to support multiple learning updates (Mnih et al., 2015; Schaul et al., 2015; Fedus et al., 2020). For value-based learning, the role of replay is relatively straightforward. Past transitions serve to calculate supplementary targets for temporal-difference updates or Bellman regression. In contrast, using historical data for policy learning introduces additional complexities. The policy improvement signal is defined with respect to the current policy, whereas replayed actions originate from previous behavior policies. Standard actor updates therefore treat replayed samples as off-policy data for the current policy-improvement step (Wang et al., 2017; Espeholt et al., 2018). Near-on-policy methods limit reuse and constrain the likelihood ratio between the behavior and current policies, while more explicitly off-policy methods use importance weighting or related corrections (Schulman et al., 2017; Espeholt et al., 2018; Wang et al., 2017). Alternatively, methods such as DDPG, TD3, and SAC learn an off-policy critic from replay and update the actor through that critic, so the actor update itself stays on-policy while the critic absorbs the off-policy data, using target networks and slowly changing value estimates to stabilize learning (Lillicrap et al., 2016; Fujimoto et al., 2018; Haarnoja et al., 2018).

We propose an alternative perspective and objective for the use of replay data. The analysis begins with regularized dual averaging (RDA), which determines the subsequent decision rule using accumulated first-order feedback instead of relying solely on a local update around the latest iterate. For RDA2C, this feedback is provided by critic-estimated advantages. From this perspective, an older advantage estimate is one of the observations that defines the cumulative dual objective from which the next policy is computed, and it counts as much as the newest one. Replay thus serves both as a source of additional optimization steps on prior experience and as a statistical representation of the actor's optimization problem.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

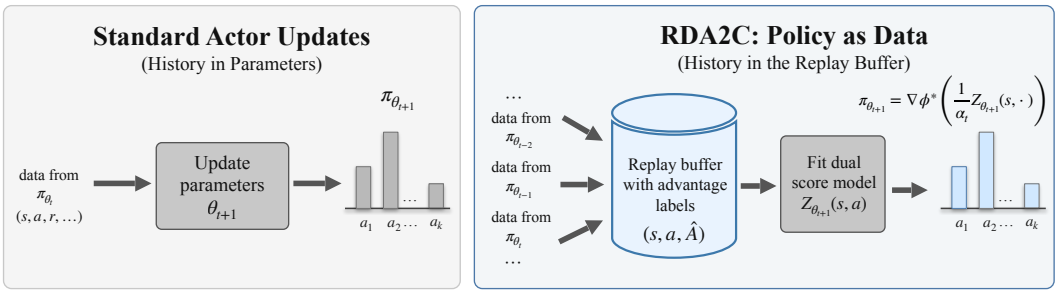


Figure 1: **Left** Standard actor updates compress policy-improvement history into the current parameters. **Right** In RDA2C, the policy is determined by the replayed data. The buffer stores accumulated advantage evidence, the dual score summarizes that evidence, and the mirror map converts it into the next policy.

The distinction is most apparent when comparing dual averaging with local mirror descent. Methods such as natural policy gradient, TRPO, PPO, and policy mirror descent can be interpreted as local policy-space updates (Kakade, 2001; Schulman et al., 2015; 2017; Lan, 2023). Their optimization history is summarized by the most recent policy or actor parameters. In contrast, regularized dual averaging and Follow the Regularized Leader optimize accumulated linearized feedback in conjunction with a regularizer (Nesterov, 2009; Xiao, 2010; McMahan, 2011). This cumulative formulation does not claim a universal optimization rate advantage over mirror descent. It highlights a concrete statistical object, a historical average of advantage feedback that can be estimated from replay.

We instantiate this idea in Regularized Dual Averaging Actor Critic. At each iteration, RDA2C stores state-action samples along with critic-estimated advantage labels in a replay buffer. A dual score model $Z_\theta(s, a)$ is trained via regression on this aggregated dataset, thereby estimating a replay-weighted average of historical policy-improvement feedback. The current policy is subsequently derived directly from the learned score using the entropy mirror map. Thus the learned dual model provides the policy's natural-coordinate representation, with no separately optimized projection actor.

This construction motivates the phrase *policy as data*. In conventional recursive actor updates, historical improvement information is implicitly compressed into the current network parameters. In RDA2C, the replay buffer explicitly stores the advantage evidence that defines a moving empirical dual objective, and Z_θ summarizes this objective. Modifying the contents of the buffer alters the actor objective, and fitting Z_θ and applying the mirror map computes the policy induced by the updated data. This perspective turns replay from a mechanism for reusing stale experience into an explicit representation of the policy-improvement problem (Figure 1), leading to the algorithmic, theoretical, and empirical contributions below.

- **A replay-based dual-averaging actor update.** We derive an entropy-regularized actor–critic method in which replayed advantage-labeled samples define an empirical cumulative dual objective, a learned score model Z_θ estimates that objective, and the policy is obtained by an exact mirror-map readout for both finite and continuous action spaces.

- **A finite-time analysis of replayed dual feedback.** We give a value-gap decomposition that separates the standard RDA optimization term from the errors introduced by a practical replay implementation. The analyzed update closely tracks the implemented one, with the bounds applying to the replayed labels, finite FIFO window, and fitted score model. The decomposition covers local stale-replay drift, supervised fitting error, critic-label bias, finite-buffer variance, and replay coverage.

- **Evaluation across critic constructions and action spaces.** We instantiate RDA2C using both GAE and twin-Q advantage labels. Across eight MuJoCo-v5 tasks and twelve Atari games, the results show that the same replay-based dual-score actor can be used with on-policy-style value critics, off-policy twin-Q critics, continuous Gaussian policies, and discrete softmax policies.

2 PRELIMINARIES

We consider a discounted Markov decision process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \rho_0, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(\cdot \mid s, a)$ is the transition kernel, $r(s, a)$ is the reward function, ρ_0 is the initial state distribution, and $\gamma \in [0, 1)$ is the discount factor. A stationary policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ maps each state s to an action distribution. We write $\tau = (s_0, a_0, s_1, a_1, \ldots) \sim \pi$ for a trajectory generated by $s_0 \sim \rho_0$, $a_t \sim \pi(\cdot \mid s_t)$, and $s_{t+1} \sim P(\cdot \mid s_t, a_t)$. The discounted return objective is $J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$. For a policy π, the value function, action-value function, and advantage function are $V^\pi(s) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,|\, s_0 = s\right]$, $Q^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}\left[V^\pi(s')\right]$, and $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$. Equivalently, $V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}[Q^\pi(s, a)]$, and hence $\mathbb{E}_{a\sim\pi(\cdot|s)}[A^\pi(s, a)] = 0$. A policy π induces the normalized discounted state visitation distribution $d^\pi(s) = (1 - \gamma)\sum_{t=0}^{\infty} \gamma^t \mathbb{P}_\pi(s_t = s)$, where the probability is over trajectories generated by π. We write $\bar{d}_t = \frac{1}{t}\sum_{k=1}^{t} d^{\pi_k}$ for the average historical mixture of visitation distributions, and use d without a policy superscript to denote an arbitrary state density over $\mathcal{S}$. We also write μ^π for the induced discounted *state-action* occupancy, with $\mu^\pi(s, a) = d^\pi(s)\pi(a \mid s)$ and $d^\pi(s) = \sum_a \mu^\pi(s, a)$, replacing the sum by an integral for continuous actions.

3 BACKGROUND

Mirror descent is a first-order method for optimization over non-Euclidean geometries (Nemirovsky & Yudin, 1983; Beck & Teboulle, 2003). It measures movement with a Bregman divergence induced by a strictly convex mirror potential in place of the Euclidean norm. Let $\varphi : \Delta(\mathcal{A}) \to \mathbb{R}$ be a differentiable, strictly convex statewise mirror potential on the action simplex. In this paper, we use the statewise negative entropy $\varphi(p) = \sum_{a\in\mathcal{A}} p(a) \log p(a)$ as the mirror potential. The potential on the full policy is its state-weighted average, $\varphi_d(\pi) = \mathbb{E}_{s\sim d}[\varphi(\pi(\cdot \mid s))]$ for a state weighting d, and every statewise statement below extends to the policy space through this average. For a fixed state, the Bregman divergence induced by this potential is the KL divergence $D_\varphi(p, q) = D_{\mathrm{KL}}(p\|q)$. The entropy mirror potential also identifies the action probabilities $\pi(\cdot \mid s)$ as primal variables and policy logits as dual coordinates. We represent the latter by $\log \pi(\cdot \mid s)$, omitting action-independent offsets in $\nabla\varphi$ throughout because they do not affect the map back to the simplex. Conversely, a dual score vector $z \in \mathbb{R}^{|\mathcal{A}|}$ maps back to the simplex by the inverse entropy mirror map, equivalently $\nabla\varphi^*(z)$ with $\varphi^*(z) = \log \sum_{a\in\mathcal{A}} \exp(z(a))$, yielding $\pi_z(a \mid s) = \nabla\varphi^*(z)(a) = \frac{\exp(z(a))}{\sum_{b\in\mathcal{A}} \exp(z(b))}$. We call this map from dual scores to a policy the readout. It is the gradient of the conjugate potential, so it belongs to the mirror geometry, and any method that maintains dual scores obtains its policy through it.

3.1 DIRECT FUNCTIONAL REPRESENTATION

In the direct functional representation, the policy itself is the state indexed collection of action distributions $\{\pi(\cdot \mid s) : s \in \mathcal{S}\}$. At iteration t, actor–critic methods obtain a local functional gradient direction from the critic and use it to update the policy. In policy space, this can be viewed as freezing the current visitation density d^{π_t} and forming a linearized improvement objective around π_t. With respect to the d^{π_t}-weighted inner product $\langle f, \pi\rangle_d = \mathbb{E}_{s\sim d}[\sum_a f(s, a)\pi(a \mid s)]$ between score functions and policies, the current-policy advantage A^{π_t} represents the functional gradient of J at π_t, satisfying $\langle \nabla_\pi J(\pi_t), \pi\rangle_{d^{\pi_t}} = \langle A^{\pi_t}, \pi\rangle_{d^{\pi_t}}$, up to terms independent of π. We therefore use A^{π_t} as the functional gradient representative and write the local linearized improvement term as $\langle A^{\pi_t}, \pi\rangle_{d^{\pi_t}}$. Because d^{π_t} changes across rounds, accumulated feedback needs one common coordinate system.

3.2 REGULARIZED DUAL AVERAGING IN POLICY SPACE

Regularized dual averaging (RDA) uses the same policy-space mirror geometry as Policy Mirror Descent (PMD) (Lan, 2023), but applies it to a cumulative linearized objective rather than to a one-step local surrogate (Nesterov, 2009; Xiao, 2010). In its general form, RDA selects $x_{t+1} = \arg\min_x\{\frac{1}{t}\sum_{k=1}^{t}\langle g_k, x\rangle + \Psi(x) + \frac{\beta_t}{t}\varphi(x)\}$ for loss gradients g_k, and the three parts play different roles. The averaged linear terms carry the feedback, and every round contributes one term at equal

weight. The composite term Ψ is persistent, maintaining a constant weight relative to the feedback average and thereby shaping the solution to which the iterates converge. The stabilizer φ damps early iterates, and its relative weight β_t/t must vanish for the feedback to take over, which holds whenever β_t grows sublinearly. Our instantiation takes both Ψ and φ to be entropy, with persistent weight λ and, in our experiments, a constant $\beta_t = \beta$. The equal-weight sum over past feedback is also the term our replay window later truncates with a finite window that keeps only recent rounds, trading the variance reduction of a long memory against the staleness of old advantage estimates (Section 5).

In actor–critic methods, the feedback from round t is generated under the visitation distribution d^{π_t}, so each linear term retains the state weighting of the policy that produced it. To regularize different policies on the same historical state support, we weight by the historical mixture $\bar{d}_t$ from Section 2. The resulting state-weighted policy-space RDA update is

$$\pi_{t+1} = \arg\max_{\pi\in\Pi} \left\{ \frac{1}{t}\sum_{k=1}^{t} \langle A^{\pi_k}, \pi\rangle_{d^{\pi_k}} - \Psi_{\bar{d}_t}(\pi) - \frac{\beta_t}{t}\varphi_{\bar{d}_t}(\pi) \right\}, \tag{1}$$

where $\Psi_{\bar{d}_t}$ is the persistent composite regularizer, $\varphi_{\bar{d}_t}$ is a strongly convex stabilizer, and $\beta_t \geq 0$ is the stabilizer weight. With $g_k = -A^{\pi_k}$, minimizing the averaged loss is maximizing the averaged advantage, which gives the sign convention of (1).

4 Regularized Dual Averaging Actor Critic

We now present RDA2C, a replay-based actor–critic algorithm that instantiates the policy space RDA update from the previous section. The central object in RDA2C is a dual score function $Z_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which approximates the state-density-weighted average advantage Z_t defined in Section 4.1. Throughout, a score is a function on state-action pairs read in dual (logit) coordinates, not the likelihood score $\nabla_\theta \log \pi_\theta$ used in policy gradient methods. RDA2C estimates the cumulative policy improvement signal required by RDA and then converts this score directly into a policy using entropy geometry.

Concretely, at each policy round the critic attaches an advantage label to each sampled state–action pair. These labeled examples are retained for several recent rounds, and Z_θ is trained by supervised regression to predict their replay-weighted average. The policy is then obtained directly from the learned score through the entropy mirror map. RDA2C thus separates the actor update into advantage-label generation, dual-score regression, and mirror-map readout.

4.1 Deriving the Dual Function

RDA is written in terms of an average dual object, so we first rewrite the accumulated, state weighted exact advantage feedback as a single inner product under the historical state mixture. For states with $\bar{d}_t(s) > 0$, define the density weighted average advantage score by

$$\frac{1}{t}\sum_{k=1}^{t} \langle A^{\pi_k}, \pi\rangle_{d^{\pi_k}} = \langle Z_t, \pi\rangle_{\bar{d}_t}, \quad \text{where} \quad Z_t(s,a) \coloneqq \frac{\sum_{k=1}^{t} d^{\pi_k}(s) A^{\pi_k}(s,a)}{\sum_{k=1}^{t} d^{\pi_k}(s)}. \tag{2}$$

The cumulative advantage feedback is then represented by the single average dual score Z_t.

For RDA2C, we set the composite regularizer in (1) as $\Psi_{\bar{d}_t} = \lambda\varphi_{\bar{d}_t}$. Both terms use the negative-entropy geometry, with the roles described in Section 3.2, where $\lambda\varphi_{\bar{d}_t}$ is the persistent entropy floor, and $(\beta_t/t)\varphi_{\bar{d}_t}$ is the transient stabilizer. Together they give the effective entropy temperature $\alpha_t = \lambda + \beta_t/t$. Thus, after rewriting the normalized cumulative advantage feedback as $\langle Z_t, \pi\rangle_{\bar{d}_t}$, the RDA update becomes $\pi_{t+1} = \arg\max_{\pi\in\Pi} \{\langle Z_t, \pi\rangle_{\bar{d}_t} + \alpha_t H_{\bar{d}_t}(\pi)\}$, where $H_{\bar{d}_t} = -\varphi_{\bar{d}_t}$.

Although the density-weighted objective in (1) is written over the product policy space, it decomposes statewise. For any state with $\bar{d}_t(s) > 0$, the same positive scalar $\bar{d}_t(s)$ multiplies both the linear score term and the entropy term, so it does not affect the optimizer at that state, provided the score vector $Z_t(s,\cdot)$ can be set independently at each state, as in the tabular case. The density weights therefore determine how the average score Z_t is formed and which states are covered, and once $Z_t(s,\cdot)$ is fixed, the policy at state s is obtained by the entropy mirror-map readout. With function approximation the weights matter again, because the approximator trades larger error at low-weight states for smaller

Algorithm 1 RDA2C: Regularized Dual Averaging Actor Critic

Input: replay capacity or window schedule, regularization λ, stabilizer schedule $\{\beta_t\}_{t\geq 1}$
Initialize dual score model Z_θ, critic learner $\mathcal{C}$, replay buffer $\mathcal{B}_0 = \emptyset$, and initial policy π_1
for iteration $t = 1, \ldots, T$ **do**
  Collect trajectories using current policy π_t
  Compute advantage labels $\hat{A}_t$ with critic learner $\mathcal{C}$ *e.g., GAE or twin-Q labels*
  Form labeled data $\mathcal{D}_t = \{(s_i, a_i, \hat{A}_i)\}$ and push $\mathcal{D}_t$ into $\mathcal{B}_t$ *evicting old data as needed*
  Update critic learner $\mathcal{C}$ using the backbone's return or TD targets
  Set temperature $\alpha_t = \lambda + \beta_t/t$
  Fit Z_θ on minibatches from $\mathcal{B}_t$ using (4)
  Read out the next policy $\pi_{t+1} = \nabla\varphi^*(Z_{\theta_t}/\alpha_t)$
end for

error at high-weight ones. For finite action spaces, the entropy conjugate readout $\nabla\varphi^*$ is the softmax map. For continuous action spaces, it is the normalized Boltzmann readout. We use the quadratic score class $Z_\theta(s,a) = -\frac{1}{2}\sum_i (a_i - \mu_{\theta,i}(s))^2/\sigma^2_{\theta,i}(s) + b_\theta(s)$, for which this readout is the diagonal Gaussian $\mathcal{N}\big(\mu_\theta(s), \alpha_t \operatorname{diag}(\sigma^2_\theta(s))\big)$, and the bias $b_\theta(s)$ cancels in the normalization (Appendix A.2). Solving the statewise objective in closed-form gives the readout:

$$\pi^{\mathrm{RDA}}_{t+1}(\cdot \mid s) = \operatorname{softmax}\left(\frac{Z_t(s,\cdot)}{\alpha_t}\right) \tag{3}$$

4.2 Policy as data: fitting the aggregated advantage

The update in (3) is defined through the average score Z_t from (2), and no algorithm can maintain that object exactly, because it averages full advantage functions over rounds. RDA2C realizes the average statistically instead, through dataset aggregation (Ross et al., 2011). Each round contributes advantage-labeled samples, a sliding window retains the recent rounds, and one squared-loss regression fits a single score model to everything retained. The regression does the averaging. The mirror map does the readout. The full loop is Algorithm 1.

Three variants of the algorithm follow from what kind of advantage feedback is available. The first stores the full vector $A(s,\cdot)$ at each visited state. It shows most directly how regression performs dual averaging: when a state is visited in several rounds, the regression drives $Z_\theta(s,\cdot)$ toward the average of the stored vectors. It requires enumerating the action set and a critic that can be queried at every action, including actions the recent policies never took. The second stores only the action taken at each visited state, which is all a value critic can label. Regression then targets the behavior-weighted average at each pair instead of the unweighted one, an extra error that the analysis in Section 5 accounts for, in exchange for feasibility with large action spaces and on-policy labels. The third sits between the two. With a Q critic, we score two actions sampled from the current policy at each replayed state, one as the label and one as the baseline, which gives a one-sample advantage label without enumeration. The GAE settings on MuJoCo and Atari use the second variant, and the twin-Q MuJoCo setting uses the third. Appendix A gives the label construction.

Formally, for the second variant, at iteration t the current policy π_t generates labeled samples $\mathcal{D}_t = \{(s_{t,i}, a_{t,i}, \widehat{A}_{t,i})\}_{i=1}^{n_t}$, and a FIFO or sliding window retains the recent rounds, $\mathcal{B}_t = \mathrm{Window}(\mathcal{D}_1, \ldots, \mathcal{D}_t)$. At actor update time, minibatches are drawn from the realized buffer, so the sampled tuples follow the buffer's empirical distribution. Let P^{rep}_t denote the population replay law over labeled triples $(s, a, \widehat{A})$ induced by the window at iteration t. The population dual regression problem is to minimize

$$\mathcal{L}_{\mathrm{dual}}(\theta, t) = \mathbb{E}_{(s,a,\widehat{A})\sim P^{\mathrm{rep}}_t}\left[(Z_\theta(s,a) - \widehat{A})^2\right]. \tag{4}$$

In implementation, this loss is approximated by minibatches from $\mathcal{B}_t$. The reported GAE runs on MuJoCo and Atari replace the squared loss by the clipped variant of Appendix A.4, which the ablation in Appendix D.4 shows matters in practice. The twin-Q runs use the squared loss on standardized labels.

For the squared-loss objective, the population regression target is the replay conditional mean label $\widehat{Z}_t(s,a) = \mathbb{E}_{P^{\mathrm{rep}}_t}[\widehat{A} \mid S = s, A = a]$. The regression therefore fits the replay Bayes target $\widehat{Z}_t$, and

fitting on a finite buffer introduces additional sampling error. This is the sense in which the policy is represented by data, with the buffer defining an empirical dual objective, Z_θ serving as its fitted representation, and the entropy readout of Z_θ giving the current policy.

The action weighting is the one gap the analysis closes by assumption: the bounds charge it to a coverage coefficient, which is finite when the replayed policies put positive probability on every action at every state the comparator visits (Appendix F.4).

4.3 Connections to Key Baselines

We highlight the algorithmic differences of the baselines most relevant to our method; a broader discussion of related work appears in Appendix G.

Connection to SAC. SAC and the twin-Q variant of RDA2C use the same critic-learning procedure but differ in their actor targets and update mechanisms. SAC's actor tracks the softmax of the latest Q estimate, whereas RDA2C derives its policy from a fitted replay-weighted average of recent critic advantage labels. SAC optimizes a KL projection objective using stochastic gradients on sampled actions. RDA2C instead fits a quadratic dual score to stored labels by supervised regression and obtains its policy through the entropy mirror map.

Despite these differences, we hypothesize comparable performance under matched critic learning. Tomar et al. (2022) show that SAC's actor step is an entropy-regularized follow-the-regularized-leader update, the same form as the regularized dual averaging step behind RDA2C, so with the critic shared the two methods differ only in how the cumulative score is formed. They also report that off-policy MDPO-KL performs on par with SAC across all six MuJoCo tasks in their evaluation, providing an empirical precedent for comparable performance despite differences in the actor update.

This parity is useful because RDA2C is not tied to a particular critic construction. The same actor update can use either GAE labels from a value critic or advantage labels from twin-Q critics, and applies to both discrete and continuous action spaces. SAC, by contrast, relies on a soft Q-critic construction for its actor update and is therefore not directly applicable in settings where only trajectory-based GAE labels are available. Our results show that RDA2C improves over PPO in the GAE setting while matching SAC when twin-Q labels are available, supporting a single actor formulation across both regimes.

Connection to AAPDA. AAPDA (Gao et al., 2026) and RDA2C both build on policy dual averaging, but differ in how they obtain the policy from the accumulated feedback. AAPDA learns a separate actor to approximate the policy dual-averaging target. RDA2C instead fits the dual score from replayed advantage labels and obtains the policy directly through the entropy mirror map, without a separate actor-fitting stage.

5 Theoretical Analysis

The analysis is for finite action spaces and separates the entropy-RDA optimization term from the replay score errors introduced by the fitted actor. The fitted actor is the entropy readout of the learned replay score. As in (3), π_{θ_t} is the policy that Algorithm 1 reads out at the end of round $t-1$ from $Z_{\theta_{t-1}}$ with temperature α_{t-1},

$$\pi_{\theta_t}(\cdot \mid s) = \mathrm{softmax}\big(Z_{\theta_{t-1}}(s,\cdot)/\alpha_{t-1}\big), \qquad \alpha_{t-1} = \lambda + \beta/(t-1), \quad \lambda > 0,\ \beta \geq 0. \tag{5}$$

Here λ is the persistent entropy floor and $\beta/(t-1)$ is the decaying RDA stabilizer. The analysis specializes the stabilizer schedule $\{\beta_t\}$ of Algorithm 1 to a constant, $\beta_t \equiv \beta$, giving $\alpha_t = \lambda + \beta/t$. The proof converts policy-space regret into value loss through the performance difference lemma in Appendix F.1.

Replay window and regression bridge. The theory analyzes a replay window $\mathcal{W}_t = \{t - W_t, \ldots, t-1\}$ of length $W_t = \max\{1, \lfloor t^\zeta \rfloor\}$. Using the state-action occupancy notation from Section 2, set $\mu_k = \mu^{\pi_{\theta_k}}$, i.e., $\mu_k(s,a) = d^{\pi_{\theta_k}}(s)\pi_{\theta_k}(a \mid s)$, and let $\mu_t(s,a) = W_t^{-1} \sum_{k \in \mathcal{W}_t} \mu_k(s,a)$ be the population replay marginal. The squared-loss replay regression fits $Z_{\theta_{t-1}}$ to the replay

Bayes target of (4) and, with true advantage labels $A_k = A^{\pi_{\theta_k}}$, this target is $Z_t^A(s,a) = \big(\sum_{k\in\mathcal{W}_t}\mu_k(s,a)A_k(s,a)\big)/\big(\sum_{k\in\mathcal{W}_t}\mu_k(s,a)\big)$, on the support of μ_t. Thus Z_t^A is the replay-weighted average advantage that bridges the current advantage A_t and the learned score $Z_{\theta_{t-1}}$, where $A_k = A^{\pi_{\theta_k}}$ are the advantages of the learned policies. The comparator coverage by replay is measured by $C_{\mathrm{cov}}(\pi^\star) = \max_{1\le t\le T}\operatorname{ess\,sup}_{s,a} d^{\pi^\star}(s)/\mu_t(s,a)$, which is finite if $\mu_t(s,a) > 0$ for every action a at every state with $d^{\pi^\star}(s) > 0$, a universal action coverage condition at comparator states.

We first state the bound for the regularized value gap, the average regret against the comparator $\pi^\star$ with the persistent entropy correction kept on the left-hand side:

$$\mathcal{G}_T^{\mathrm{reg}}(\pi^\star) := \frac{1}{T-1}\sum_{t=2}^{T}\left[\big(J(\pi^\star) - J(\pi_{\theta_t})\big) + \frac{\lambda}{1-\gamma}\mathbb{E}_{s\sim d^{\pi^\star}}\big(\varphi(\pi_{\theta_t}(\cdot\mid s)) - \varphi(\pi^\star(\cdot\mid s))\big)\right].$$

Theorem 5.1 (Fitted replay value bound)**.** *Assume $\mathcal{A}$ finite, $|r(s,a)| \le R_{\max}$, the policy path stability of Assumption F.2 (constant C_π) and the replay error bounds of Assumption F.8 (Appendix F.5), and $C_{\mathrm{cov}}(\pi^\star) < \infty$. Let $\zeta \in (0,1)$ and $C_{\mathrm{loc}} = \frac{(1+\gamma)R_{\max}C_\pi}{(1-\gamma)^2(1-2^{\zeta-1})}$. Then for every stationary comparator $\pi^\star$ satisfying the above assumptions,*

$$\mathcal{G}_T^{\mathrm{reg}}(\pi^\star) \le \frac{1}{1-\gamma}\Bigg[\frac{\beta\ln|\mathcal{A}|(1+\ln T)}{T-1} + \frac{2C_{\mathrm{loc}}}{\zeta}\frac{T^\zeta}{T-1} + 2\sqrt{C_{\mathrm{cov}}(\pi^\star)}\Big(\bar{\epsilon}_{\mathrm{fit}} + \epsilon_0 + \tfrac{C_{\mathrm{var}}}{1-\zeta/2}\tfrac{T^{1-\zeta/2}}{T-1}\Big)\Bigg] =: \mathcal{B}_T(\pi^\star).$$

Converting the regularized gap to ordinary value costs an entropy bias term. The conversion turns the guarantee into an ordinary average regret bound against $\pi^\star$, comparable with unregularized methods, with an explicit bias term that scales with λ and represents the price of the entropy floor retained on the left-hand side of the regularized bound.

Corollary 5.2 (Unregularized value bound)**.** *Under the assumptions of Theorem 5.1, moving the entropy correction to the right gives*

$$\frac{1}{T-1}\sum_{t=2}^{T}\big(J(\pi^\star) - J(\pi_{\theta_t})\big) \le \frac{\lambda\ln|\mathcal{A}|}{1-\gamma} + \mathcal{B}_T(\pi^\star),$$

with $\mathcal{B}_T(\pi^\star)$ the right hand side of Theorem 5.1. With $\zeta = 2/3$ and fixed $\lambda > 0$ the ordinary value gap converges at rate $T^{-1/3}$ to the floor set by the entropy bias $\frac{\lambda\ln|\mathcal{A}|}{1-\gamma}$ and the persistent fitting and critic error $\frac{2\sqrt{C_{\mathrm{cov}}(\pi^\star)}}{1-\gamma}(\bar{\epsilon}_{\mathrm{fit}} + \epsilon_0)$.

6 Experiments

We evaluate whether the replay estimator proposed in Section 4 gives a useful actor update in both continuous and discrete control. The experiments ask five questions: whether RDA2C improves over PPO when both methods use GAE labels, whether the same actor holds parity with SAC when labels come from twin-Q critics, whether it improves on AAPDA, the prior dual-averaging actor, at that method's released defaults, whether the same update carries over to image-based discrete control on Atari, and whether the replay window behaves like the finite replay and drift terms in Section 5. Additional ablations and control experiments in Appendix D examine the effects of individual algorithmic components and implementation choices.

6.1 Experimental setup

Environments and evaluation. For the continuous control suite, we use eight Gymnasium MuJoCo v5 tasks (Todorov et al., 2012). Both the GAE and twin-Q critic MuJoCo comparisons run for 1M environment steps and use 10 random seeds. The SAC comparison reports deterministic mean action evaluation over 20 episodes every 50K steps. For the Atari suite, we evaluate on twelve Atari 2600

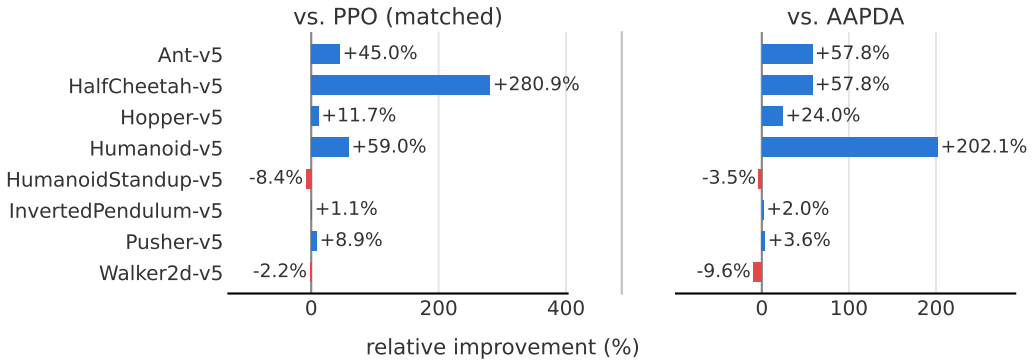


Figure 2: **RDA2C improves over matched PPO on six of eight MuJoCo tasks and over AAPDA on six of eight.** Bars show per-environment relative improvement, $100 \cdot (\bar{R}_{\text{ours}} - \bar{R}_{\text{base}})/|\bar{R}_{\text{base}}|$. Left: PPO matched to RDA2C's architecture and hyperparameters (GAE labels, online return). Right: AAPDA at its released defaults (deterministic evaluation). Exact values: Appendix B.

games (Bellemare et al., 2013) for 10M frames with 10 random seeds, using the EnvPool vectorized environment wrapper (Weng et al., 2022). Across both suites, we follow the evaluation and reporting conventions used in the corresponding baseline literature, as detailed in Appendix B.

Baselines and model classes. The GAE-based MuJoCo comparison uses PPO (Schulman et al., 2017) as the baseline. We match the 2×128 trunks, learning rate, gradient clipping, GAE λ, batch sizes, update epochs, and horizon, so the actor update is the intended difference. Both methods use a value network separate from the actor. RDA2C derives its policy from a dual score that is quadratic in the actions (Appendix A.2), whereas PPO directly parameterizes a Gaussian policy with a global log-standard-deviation.

The twin-Q MuJoCo comparison uses SAC (Haarnoja et al., 2018) with matched transition-replay settings, twin-Q critic architectures, target updates, optimizer, batch size (512), and policy update frequency (1). RDA2C retains the critic-learning procedure but replaces SAC's actor update by fitting a quadratic dual score to replayed advantage labels and obtaining the policy through the entropy mirror map.

We also compare against actor-accelerated policy dual averaging (AAPDA) (Gao et al., 2026) on MuJoCo. AAPDA learns a separate actor to approximate the policy dual-averaging target, whereas RDA2C reads its policy directly from the fitted dual score. We use AAPDA's released defaults without per-task tuning, with both methods trained for 1M environment steps over 10 paired training seeds. The evaluation protocol is detailed in Appendix B.

For Atari, RDA2C uses a discrete softmax readout and separate convolutional feature extractors for the score model and value function, while PPO keeps the shared trunk of its standard implementation (Huang et al., 2022).

6.2 Experimental results

GAE labels: RDA2C improves over PPO on six of eight MuJoCo tasks Figure 2 summarizes the relative improvement over matched PPO, and Appendix Table 4 reports the raw mean and standard deviation over 10 seeds. RDA2C improves over matched PPO on six of eight MuJoCo tasks and gives a fixed-suite average per-environment relative improvement of $+49.5\%$ (paired seed bootstrap 95% CI $[+42.1\%, +56.2\%]$).[1] The largest gains are on `HalfCheetah-v5`, where RDA2C reaches 7099 compared with PPO's 1864, and `Humanoid-v5`, where RDA2C reaches 3768 compared with PPO's 2370. These gains are obtained against a strengthened PPO baseline. The matched configuration used here has higher mean returns than PPO's CleanRL defaults on five of eight tasks (Appendix B).

[1] Percentile bootstrap intervals can undercover at seed counts like ours: Jordan et al. (2020) measure failure rates above twice the nominal level, and in a simulation matched to our data the suite-level (mujoco and atari suites) procedure used here covered a nominally 95% target 90.5% of the time. We therefore read these intervals as descriptive uncertainty summaries. This caveat applies to every bootstrap interval in the paper.

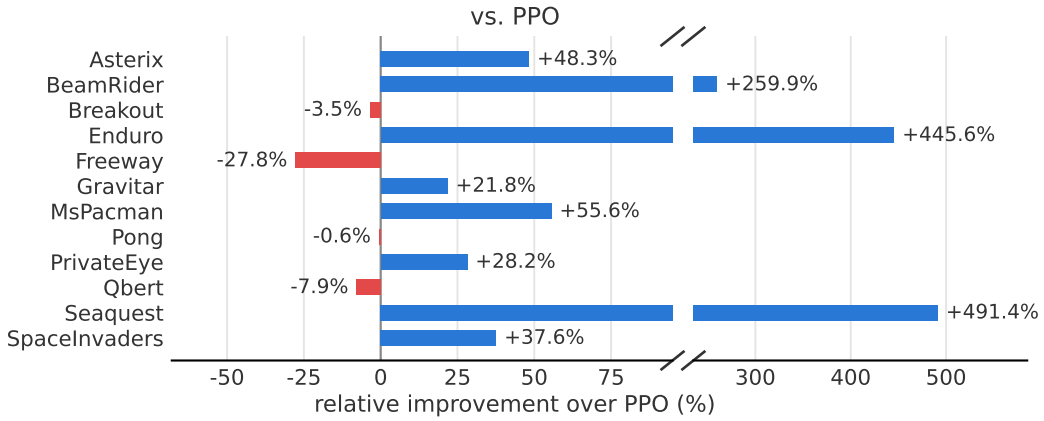


Figure 3: **RDA2C improves over PPO on eight of twelve Atari games.** Bars show relative improvement over PPO, $100 \cdot (\bar{R}_{\mathrm{RDA2C}} - \bar{R}_{\mathrm{PPO}})/|\bar{R}_{\mathrm{PPO}}|$, after 10M frames. The value axis is broken between $+95\%$ and $+235\%$. The bars for `BeamRider-v5`, `Enduro-v5`, and `Seaquest-v5` continue in the right segment. Exact mean $\pm$ std values are reported in Appendix B, Table 9.

Twin-Q critic labels: RDA2C matches SAC under matched settings RDA2C is higher on four tasks, ties on one, and is lower on three. Appendix Table 5 reports the underlying best deterministic evaluations within the 1M-step budget over 10 seeds. The suite-level summary is a fixed-suite average per-environment relative change of $+0.64\%$ (paired seed bootstrap 95% CI $[-2.80\%, +4.06\%]$)[1]. The largest single-task difference favors RDA2C on `Walker2d-v5` (4505 vs 4080), and the largest in SAC's favor is on `HalfCheetah-v5` (9525 vs 10119). No taskwise difference is decisive at 10 seeds. The outcome matches the expectation in Section 4.3, and it answers the question the experiment asks: the same dual-score regression works with twin-Q labels in place of GAE, at SAC's level of return. The comparison against SAC at its CleanRL-default batch size and update frequency appears in Appendix B.

AAPDA: RDA2C improves on the closest dual-averaging baseline AAPDA is the closest runnable dual-averaging baseline (Gao et al., 2026), described in Appendix G.2. We ran the authors' released defaults for 1M steps with no per-task tuning. RDA2C is higher on six of eight tasks. The right panel of Figure 2 summarizes the comparison, and Appendix Table 6 gives the per-task returns. The fixed-suite mean relative change is $+41.8\%$ (paired seed bootstrap 95% CI $[+26.3\%, +71.2\%]$)[1] and the median is $+13.8\%$ (CI $[+4.9\%, +27.0\%]$)[1]. AAPDA is ahead on `HumanoidStandup-v5` and `Walker2d-v5`.

Atari: RDA2C improves over PPO on eight of twelve games Figure 3 reports relative improvement over PPO after 10M frames, and Appendix Table 9 gives the raw returns. RDA2C improves over PPO on 8 of the 12 Atari games, with a fixed-suite average per-environment relative improvement of $+112.4\%$ (paired seed bootstrap 95% CI $[+88.5\%, +141.2\%]$)[1] and a median relative improvement of $+32.9\%$. The largest gains are on `Seaquest-v5`, `Enduro-v5`, and `BeamRider-v5`. PPO is higher on `Breakout-v5`, `Freeway-v5`, `Pong-v5`, and `Qbert-v5`. The Atari results show that the same average advantage parameterization can scale to image-based discrete control. The table reports mean running return and seed standard deviation over 10 random seeds.

7 Conclusion and Future Work

We introduced RDA2C, a replay-based actor–critic implementation of regularized dual averaging via advantage regression. In RDA2C, replayed advantage-labeled samples define an empirical cumulative dual objective, which is summarized by a learned score model. The policy is then obtained directly through the entropy mirror map. Replay then serves both as a source of additional optimization steps and as the representation of policy-improvement feedback from which the current actor is derived.

AI USE STATEMENT

In this work, we used generative AI tools to implement analysis and sweep code. Additionally, we used generative AI tools to edit parts of the paper for readability and structure and to create and modify figures. We have reviewed all AI-assisted work. All experiments were run by the authors, all AI-generated code was checked and tested by the authors, and all AI-drafted text was revised by the authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix A gives network architectures, hyperparameters for every setting, and the label constructions. Appendix B reports the numerical results behind every figure, with seed counts and evaluation protocols. The assumptions and complete proofs of the theoretical results are in Appendix F. The code for all three entrypoints follows the CleanRL single-file convention and will be released.

REFERENCES

Yasin Abbasi-Yadkori, Peter Bartlett, Kush Bhatia, Nevena Lazic, Csaba Szepesvári, and Gellért Weisz. POLITEX: Regret bounds for policy iteration using expert prediction. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3692–3702, 2019.

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Rémi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1806.06920.

Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *arXiv preprint arXiv:1908.00261*, 2019. doi: 10.48550/arXiv.1908.00261. URL https://arxiv.org/abs/1908.00261.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47: 253–279, June 2013.

Kianté Brantley, Mingyu Chen, Zhaolin Gao, Jason D. Lee, Wen Sun, Wenhao Zhan, and Xuezhou Zhang. Accelerating RL for LLM reasoning with optimal advantage regression. *arXiv preprint arXiv:2505.20686*, 2025. doi: 10.48550/arXiv.2505.20686. URL https://arxiv.org/abs/2505.20686.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1407–1416. PMLR, 2018. URL https://proceedings.mlr.press/v80/espeholt18a.html.

William Fedus, Prajit Ramachandran, Rishabh Agarwal, Yoshua Bengio, Hugo Larochelle, Mark Rowland, and Will Dabney. Revisiting fundamentals of experience replay. In *International conference on machine learning*, pp. 3061–3071. PMLR, 2020.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1587–1596. PMLR, 2018. URL https://proceedings.mlr.press/v80/fujimoto18a.html.

Ji Gao, Caleb Ju, Guanghui Lan, and Zhaohui Tong. Actor-accelerated policy dual averaging for reinforcement learning in continuous action spaces. *arXiv preprint arXiv:2603.10199*, 2026. URL https://arxiv.org/abs/2603.10199.

Zhaolin Gao, Jonathan D. Chang, Wenhao Zhan, Owen Oertell, Gokul Swamy, Kianté Brantley, Thorsten Joachims, J. Andrew Bagnell, Jason D. Lee, and Wen Sun. REBEL: Reinforcement learning via regressing relative rewards. *arXiv preprint arXiv:2404.16767*, 2024a. doi: 10.48550/arXiv.2404.16767. URL https://arxiv.org/abs/2404.16767.

Zhaolin Gao, Wenhao Zhan, Jonathan D. Chang, Gokul Swamy, Kianté Brantley, Jason D. Lee, and Wen Sun. Regressing the relative future: Efficient policy optimization for multi-turn RLHF. *arXiv preprint arXiv:2410.04612*, 2024b. doi: 10.48550/arXiv.2410.04612. URL https://arxiv.org/abs/2410.04612.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2160–2169. PMLR, 2019. URL https://proceedings.mlr.press/v97/geist19a.html.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, 2018. doi: 10.48550/arXiv.1801.01290. URL https://arxiv.org/abs/1801.01290.

Shengyi Huang, Rousslan Fernand Julien Dossa, Antonin Raffin, Anssi Kanervisto, and Weixun Wang. The 37 implementation details of proximal policy optimization. ICLR Blog Track, 2022. URL https://iclr-blog-track.github.io/2022/03/25/ppo-implementation-details/.

Ryo Iwaki. Mirror descent actor critic via bounded advantage learning. *arXiv preprint arXiv:2502.03854*, 2025. doi: 10.48550/arXiv.2502.03854. URL https://arxiv.org/abs/2502.03854.

Scott M. Jordan, Yash Chandak, Daniel Cohen, Mengxue Zhang, and Philip S. Thomas. Evaluating the performance of reinforcement learning algorithms. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Caleb Ju and Guanghui Lan. Policy optimization over general state and action spaces. *arXiv preprint arXiv:2211.16715*, 2022. doi: 10.48550/arXiv.2211.16715. URL https://arxiv.org/abs/2211.16715.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 267–274, 2002.

Sham M. Kakade. A natural policy gradient. In *Advances in neural information processing systems*, pp. 1531–1538, 2001.

Guanghui Lan. Policy mirror descent for reinforcement learning: Linear convergence, new sampling complexity, and generalized problem classes. *Mathematical Programming*, 2023. URL https://arxiv.org/abs/2102.00135.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1509.02971.

Brendan McMahan. Follow-the-regularized-leader and mirror descent: Equivalence theorems and L1 regularization. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 525–533. PMLR, 2011. URL https://proceedings.mlr.press/v15/mcmahan11b.html.

Jincheng Mei, Chenjun Xiao, Ruitong Huang, Dale Schuurmans, and Martin Müller. On principled entropy exploration in policy optimization. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)*, 2019. URL https://www.ijcai.org/proceedings/2019/0434.pdf.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Arkadi S. Nemirovsky and David B. Yudin. *Problem complexity and method efficiency in optimization*. Wiley-Interscience, 1983.

Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical programming*, 120(1):221–259, 2009.

Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017. doi: 10.48550/arXiv.1705.07798.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Stephane Ross and J. Andrew Bagnell. Reinforcement and imitation learning via interactive no-regret learning. *arXiv preprint arXiv:1406.5979*, 2014.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635, 2011.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897, 2015.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Lior Shani, Yonathan Efroni, and Shie Mannor. Adaptive trust region policy optimization: Global convergence and faster rates for regularized mdps. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5668–5675, 2020.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization. In *International Conference on Learning Representations*, 2022. doi: 10.48550/arXiv.2005.09814. URL https://arxiv.org/abs/2005.09814.

Mariya Toneva, Alessandro Sordoni, Rémi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

Sharan Vaswani, Olivier Bachem, Simone Totaro, Robert Müller, Shivam Garg, Matthieu Geist, Marlos C. Machado, Pablo Samuel Castro, and Nicolas Le Roux. A general class of surrogate functions for stable and efficient reinforcement learning. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 8619–8649. PMLR, 2022. URL https://proceedings.mlr.press/v151/vaswani22a.html.

Nino Vieillard, Tadashi Kozuno, Bruno Scherrer, Olivier Pietquin, Rémi Munos, and Matthieu Geist. Leverage the average: an analysis of KL regularization in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020a.

Nino Vieillard, Olivier Pietquin, and Matthieu Geist. Deep conservative policy iteration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6070–6077, 2020b.

Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1611.01224.

Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh Merel, Jost Tobias Springenberg, Scott Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, and Nando de Freitas. Critic regularized regression. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Jiayi Weng, Min Lin, Shengyi Huang, Bo Liu, Denys Makoviichuk, Viktor Makoviychuk, Zichen Liu, Yufan Song, Ting Luo, Yukun Jiang, Zhongwen Xu, and Shuicheng Yan. Envpool: A highly parallel reinforcement learning environment execution engine. *arXiv preprint arXiv:2206.10558*, 2022. URL https://arxiv.org/abs/2206.10558.

Lin Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11(Oct):2543–2596, 2010.

Zhihan Xiong, Maryam Fazel, and Lin Xiao. Dual approximation policy optimization. *arXiv preprint arXiv:2410.01249*, 2024. doi: 10.48550/arXiv.2410.01249. URL https://arxiv.org/abs/2410.01249.

APPENDIX CONTENTS

A Implementation Details **15**

A.1 Network Architectures . . . 15

A.2 Action classes and label settings . . . 16

A.3 Hyperparameters . . . 16

A.4 Optional clipped dual regression . . . 18

B Main-Text Numerical Result Tables **18**

C Additional Experimental Results **22**

C.1 MuJoCo GAE label learning curves . . . 22

C.2 MuJoCo twin-Q critic label learning curves . . . 22

C.3 Atari Learning Curves . . . 24

D Ablation Studies **25**

D.1 Replay Window Size . . . 25

D.2 RDA Stabilizer . . . 28

D.3 Entropy Floor Sensitivity . . . 29

D.4 Advantage-Label Loss . . . 30

D.5 SAC Replay Buffer Size . . . 31

D.6 PPO Network Capacity Control (MuJoCo, GAE) . . . 32

E RDA2C Gradient Boosting with XGBoost **32**

E.1 Overview . . . 32

E.2 Hyperparameters . . . 33

E.3 Results . . . 34

F Proofs and implications for policy performance **35**

F.1 Performance difference lemma . . . 35

F.2 Window notation and replay Bayes target . . . 35

F.3 Layer 1: replay window Bayes target and local drift . . . 36

F.4 Layer 2: direct regularized learned readout value transfer . . . 38

F.5 Layer 3: replay regression error decomposition . . . 40

F.6 Final windowed value bound . . . 41

F.7 Density weighted surrogate and the action sampling mismatch . . . 43

F.8 Scope of the guarantee . . . 44

G Related Work **46**

G.1 Policy mirror descent, natural gradients, and trust region methods . . . 46

G.2 FTRL, regularized dual averaging, and policy dual averaging . . . 46

G.3 Replay, off policy actor–critic learning, and stale data . . . 47

G.4 Reduction and regression views of policy improvement . . . 47

A Implementation Details

A.1 Network Architectures

We utilize standard architectures for continuous and discrete control domains and report the relevant capacity differences explicitly. For GAE MuJoCo, RDA2C uses the quadratic dual score actor described below with a separate critic, while PPO uses the standard CleanRL Gaussian policy/value architecture.

Continuous control (MuJoCo, GAE labels). For the GAE MuJoCo experiments, the actor is the quadratic dual score network described in Section 4. It has a two-layer Tanh MLP trunk of width 128 and three heads: Gaussian mean, state dependent log standard deviation, and scalar bias $b_\theta(s)$. The value function is a separate two-layer Tanh MLP of width 128. During rollout collection we also store the old dual score prediction $Z_{\text{old}}(s, a)$ for each sampled transition, which is used only by the optional clipped dual regression loss of Appendix A.4.

- **Dual Score / Policy Network:** Input state $\to$ Linear(128) $\to$ Tanh $\to$ Linear(128) $\to$ Tanh $\to$ Mean / LogStd / Bias heads.
- **Value Network:** Input state $\to$ Linear(128) $\to$ Tanh $\to$ Linear(128) $\to$ Tanh $\to$ Linear(1).

Update schedule (MuJoCo, GAE labels). Each iteration collects one rollout of 2048 steps from a single environment, computes GAE labels, and pushes the 2048 tuples $(s, a, \hat{A}, Z_{\text{old}})$ into the FIFO label buffer of capacity $W = 50{,}000$, which therefore holds the most recent 24 rollouts once full. The stored action is the unsquashed Gaussian sample. The environment receives that sample clipped to the action bounds, so the policy is a clipped Gaussian and the regression is fit on the pre-clip action. The dual regression then runs 10 epochs of 32 minibatches of 64 tuples, 320 gradient steps per iteration. The $20{,}480$ tuples are drawn uniformly with replacement from the buffer once per iteration, standardized as a batch, shuffled, and consumed sequentially, so a stored label is used about 0.4 times per iteration on average and about 10 times over its residence in the buffer. The value loss in the same gradient step uses the on-policy rollout minibatch of the same size. The temperature $\alpha_t = \lambda + \beta/t$ is computed once per rollout, so with $\lambda = 0.05$ and $\beta = 1$ it decays from 1.05 to $\lambda + \beta/488$ over the 488 rollouts that make up 1M steps.

Continuous control (MuJoCo, twin-Q critic labels). For the SAC comparison, both SAC and RDA2C with twin-Q labels use the standard CleanRL SAC backbone: two-layer MLPs with width 256 and ReLU activations, together with twin-Q critics. The SAC baseline uses the standard squashed Gaussian actor and default SAC hyperparameters. Our RDA2C with twin-Q labels variant reuses the same replay / critic backbone and the same width actor trunk, but replaces the SAC actor loss with regression onto the learned quadratic dual score from Section 4. Concretely, the RDA2C with twin-Q labels actor has mean, log standard deviation, and scalar bias heads, while each Q-network maps (s, a) through 256-256 ReLU layers to a scalar estimate.

The reported twin-Q critic RDA2C runs use *automatic temperature tuning* in place of the hand-designed RDA schedule used in the GAE experiments. Specifically, for RDA2C twin-Q we optimize a scalar $\log \alpha_t$ against target entropy $-|A|$. This learned temperature determines the sampling scale of the squashed Gaussian policy through $\sigma(s)\sqrt{\alpha_t}$ and enters both the soft critic bootstrap target and the actor regression labels. For $K = 2$ current-policy samples $a_1, \ldots, a_K$ at a replayed state s, we define

$$x_i(s) = \min(Q_1, Q_2)(s, a_i) - \alpha_t \log \pi_\theta(a_i \mid s), \qquad g_i(s) = x_i(s) - \frac{1}{K-1} \sum_{j \neq i} x_j(s).$$

The regression buffer stores (s, a_i, g_i), where a_i is the corresponding pre-Tanh Gaussian sample. We store raw labels in a FIFO label buffer of total size $W = 12{,}800$, standardize them within sampled regression minibatches, clip the standardized targets to $[-10, 10]$, and regress the quadratic dual score $Z_\theta(s, A_i)$ to these buffered targets. In the primary comparison both methods use batch size 512 and policy frequency 1. SAC at its CleanRL defaults, batch size 256 and policy frequency 2, is reported separately in Appendix B. Both methods use 5 parallel environments and deterministic evaluation over 20 episodes. Because autotuning is enabled throughout this comparison, the RDA schedule parameters (λ, β) are inactive.

All MuJoCo runs used 2 CPU cores and no GPUs. All Atari runs used a single H100 GPU.

A.2 Action classes and label settings

Handling discrete and continuous actions. For discrete actions, $Z_\theta(s)$ outputs one score per action, and the entropy readout is the corresponding softmax policy. For continuous control, we choose a quadratic dual score class, $Z_\theta(s,a) = -\frac{1}{2}\sum_i \frac{(a_i-\mu_{\theta,i}(s))^2}{\sigma^2_{\theta,i}(s)} + b_\theta(s)$. Under the same entropy mirror map, $\mathrm{softmax}(Z_\theta(s,\cdot)/\alpha_t)$ is the diagonal Gaussian $\mathcal{N}\left(\mu_\theta(s), \alpha_t\ \mathrm{diag}(\sigma^2_\theta(s))\right)$. The additive term $b_\theta(s)$ cancels in the normalization and does not affect the policy.

Labels, replay windows, and stabilization. The GAE label variants in MuJoCo and Atari compute advantage labels from a value critic using generalized advantage estimates (Schulman et al., 2016). The twin-Q label variant constructs labels from the two critics by sampling actions from the current policy at replayed states and taking the smaller of the resulting soft values. The full label construction is given above. All RDA2C variants use a fixed sliding replay window for the dual regression problem. This label window is separate from the transition replay buffer that the twin-Q backbone shares with SAC. Unless changed in the ablations, we use $W = 50{,}000$ label tuples for the GAE MuJoCo setting, $W = 12{,}800$ label tuples for the twin-Q critic MuJoCo setting, and $W = 500{,}000$ frames for Atari. The Atari GAE runs use clipped dual regression with clip radius 1.0, stabilizer $\beta = 0.6$, and entropy floor $\lambda = 0.005$. The clipped dual regression is described in Appendix A.4.

A.3 Hyperparameters

We provide the hyperparameter configurations for both domains. To ensure a fair comparison, we match the **Total Batch Size** between RDA2C and the PPO baseline. Method-specific learning-rate and stabilization settings are listed explicitly below.

Continuous control (MuJoCo, GAE labels). Table 1 lists the hyperparameters for the GAE MuJoCo benchmark.

Continuous control (MuJoCo, twin-Q critic labels). Table 2 lists the hyperparameters for the SAC comparison. Both methods use the standard SAC replay and twin-Q architecture, but RDA2C with twin-Q labels uses a tuned actor regression configuration in place of the default SAC update schedule. The table therefore separates common backbone/evaluation settings from the method specific batch, policy frequency, temperature, and label construction settings.

Discrete Control (Atari). For Atari, we compare against PPO using the same total batch size per update, the same CleanRL preprocessing and EnvPool execution stack, and the same 10M-frame budget. The remaining rollout and optimization settings follow each method's own defaults and are tabled side by side in Appendix A. Table 3 lists the hyperparameters used for the 10M frame benchmark. Note that while the number of environments and steps per environment differ, the total batch size is fixed at 1024 for both algorithms.

Table 1: **MuJoCo Hyperparameters (GAE labels).** Comparison of the GAE RDA2C configuration reported in Table 4 and the PPO baseline. RDA2C uses a replay buffer of size $W = 50{,}000$ and enables optional clipped prediction regression with $\epsilon_{\text{clip}} = 0.2$. The target-advantage clip is inactive for this configuration.

Parameter	RDA2C (Ours)	PPO (Baseline)
Shared Rollout / Optimization Settings		
Optimizer	Adam	Adam
Discount Factor (γ)	0.99	0.99
Total Timesteps	1,000,000	1,000,000
Total Batch Size	2048	2048
Minibatch Size	64	64
VF Coefficient	0.5	0.5
Update Epochs	10	10
Num. Environments	1	1
Steps per Env	2048	2048
Method Specific Settings		
Actor Trunk	2×128 Tanh MLP	2×128 Tanh MLP
Actor Heads	Mean / state LogStd / bias	Mean / global LogStd
Value Network	2×128 Tanh MLP	2×128 Tanh MLP
Learning Rate	2×10^{-4}	2×10^{-4}
Hidden Width	128	128
Max Gradient Norm	3.0	3.0
GAE Lambda (λ_{GAE})	0.80	0.80
PPO Policy Ratio Clip	–	0.2
Value Loss Clipping	Yes, $\epsilon = 0.2$	Yes, $\epsilon = 0.2$
RDA2C Dual Regression Settings		
Replay Window Size (W)	50,000	–
Regularizer (λ)	**0.05**	–
Stabilizer (β)	**1.0**	–
Dual Prediction Clip (ϵ_{clip})	**0.2**	–

Table 2: **MuJoCo Hyperparameters (twin-Q critic labels).** Comparison of the SAC baseline at matched batch size and policy update frequency and the RDA2C with twin-Q labels instantiation reported in Table 5.

Parameter	RDA2C with twin-Q labels (Ours)	SAC (Baseline)
Common Backbone / Evaluation Settings		
Actor Trunk	2×256 ReLU MLP	2×256 ReLU MLP
Actor Heads	Mean, LogStd, bias	Mean, LogStd
Twin-Q Networks	2×256 ReLU MLP	2×256 ReLU MLP
Total Timesteps	1,000,000	1,000,000
Num. Environments	5	5
Replay Buffer Size	1,000,000	1,000,000
Discount Factor (γ)	0.99	0.99
Target Smoothing (τ)	0.005	0.005
Policy Learning Rate	3×10^{-4}	3×10^{-4}
Q-Learning Rate	1×10^{-3}	1×10^{-3}
Learning Starts	5,000	5,000
Target Network Frequency	1	1
Temperature Tuning	Autotune	Autotune
Evaluation Policy	Deterministic mean action	Deterministic mean action
Eval Frequency / Episodes	50,000 / 20	50,000 / 20
Method Specific SAC-Loop Settings		
Batch Size	512	512
Policy Frequency	1	1
Target Entropy	$-\lvert A\rvert$	$-\lvert A\rvert$
RDA2C twin-Q Actor Regression Settings		
Label Samples (K)	2	–
Replay States per Actor Step (B)	512	–
Label Window Size (W)	12,800	–
Advantage Clip	10.0	–
Dual Prediction Clip (ϵ_{clip})	∞ (off)	–

Table 3: **Atari Hyperparameters.** Final clipped RDA2C setting compared with PPO.

Parameter	RDA2C (Ours)	PPO (Baseline)
Shared Settings		
CNN Backbone	Nature CNN, 512-unit output	Nature CNN, 512-unit output
Actor–Critic Trunks	Independent policy and value trunks	Shared trunk
Optimizer	Adam	Adam
Learning Rate	2.5×10^{-4}	2.5×10^{-4}
LR Schedule	Linear Decay	Linear Decay
Discount Factor (γ)	0.99	0.99
Total Batch Size	1024	1024
Minibatch Size	256	256
VF Coefficient	0.5	0.5
Max Gradient Norm	2.0	0.5
Sampling Strategy		
Num. Environments	4	8
Steps per Env	256	128
Algorithm Specific		
Update Epochs	2	4
GAE Lambda (λ_{GAE})	0.85	0.95
Window Size (W)	**500,000**	–
Entropy Coeff (λ)	**0.005**	0.01
Stabilizer (β)	**0.6**	–
Dual Prediction Clip (ϵ_{clip})	**1.0**	–
PPO Policy Ratio Clip	–	0.1

A.4 Optional clipped dual regression

The main algorithm uses the squared dual regression loss in (4). In the final GAE experiments, we also use a clipped prediction variant to prevent abrupt changes in the fitted dual score under repeated replay updates. When a transition enters the buffer, we store the old prediction $Z_{\mathrm{old}}(s,a)$. For clip radius $\epsilon_{\mathrm{clip}} > 0$, define

$$Z_\theta^{\mathrm{clip}}(s,a) = Z_{\mathrm{old}}(s,a) + \mathrm{clip}\big(Z_\theta(s,a) - Z_{\mathrm{old}}(s,a), -\epsilon_{\mathrm{clip}}, \epsilon_{\mathrm{clip}}\big),$$

$$\mathcal{L}_{\mathrm{clip}}(\theta,t) = \frac{1}{2}\,\mathbb{E}_{(s,a,\hat{A},Z_{\mathrm{old}})\sim\mathcal{B}_t}\left[\max\{(Z_\theta(s,a)-\hat{A})^2, (Z_\theta^{\mathrm{clip}}(s,a)-\hat{A})^2\}\right].$$

This is a regression stabilizer analogous in spirit to PPO value function clipping (Schulman et al., 2017). It is not a policy ratio clip and does not change the mirror map readout in (3). We use $\epsilon_{\mathrm{clip}} = 0.2$ for GAE MuJoCo and $\epsilon_{\mathrm{clip}} = 1.0$ for the final GAE Atari runs. We set $\epsilon_{\mathrm{clip}} = \infty$ when this stabilizer is disabled.

B Main-Text Numerical Result Tables

This appendix collects the exact numerical tables corresponding to the bar-plot summaries in Figures 2 and 3.

Evaluation conventions. The three comparisons use three return conventions, each the one used to evaluate the corresponding baseline. The PPO comparisons on MuJoCo and Atari report the online episodic return during training, the convention of the PPO implementation we match (Huang et al., 2022). The SAC comparison reports the best deterministic evaluation return within the 1M-step budget. Evaluating with the mean action is the SAC protocol (Haarnoja et al., 2018), and reporting the maximum evaluation return within a 1M-step budget is the convention of Fujimoto et al. (2018). The AAPDA comparison reports the mean of the final five deterministic evaluations, following the final-five-epoch average of Gao et al. (2026), with both methods evaluated on the same ten fixed evaluation seeds at checkpoints every $20{,}000$ environment steps, RDA2C's rounded up to its next rollout boundary. The best-within-budget convention is optimistic for both methods it is applied to, and the conventions are not comparable across comparisons, so no number is compared across them.

Aggregate bootstrap intervals. The main-text figures plot per-environment relative improvement. For each environment, the point estimate is computed from the seed-averaged returns,

$\Delta\% = 100 \cdot (\bar{R}_{\text{ours}} - \bar{R}_{\text{base}})/|\bar{R}_{\text{base}}|$. Percentages are computed from unrounded seed means, so recomputing them from the rounded table entries can differ in the last digit. To summarize uncertainty at the panel level, we report fixed suite paired bootstrap intervals: each bootstrap replicate resamples matched baseline/ours seed pairs within each environment, computes the environment level percent improvements using that environment's fixed baseline mean for normalization, averages these improvements over the fixed benchmark suite, and takes the 2.5 and 97.5 percentiles across bootstrap replicates.

MuJoCo GAE label comparison. Table 4 reports the running average episodic returns underlying the MuJoCo/PPO panel in Figure 2. RDA2C uses GAE labels and clipped dual regression in this setting, and the table shows both the mean return and the environment-level relative improvement over PPO.

Table 4: **MuJoCo Continuous Control Results.** Mean episodic return ($\pm$ std) over 10 seeds. PPO is matched to RDA2C's architecture and optimization settings (Appendix A); the CleanRL-default comparison appears in Appendix B. RDA2C uses clipped dual regression with $\epsilon_{\text{clip}} = 0.2$ and value-loss clipping enabled. Last column reports relative change $\Delta\% \triangleq 100 \cdot (\text{RDA2C} - \text{PPO})/(|\text{PPO}|)$ (higher is better).

Environment	**PPO (Baseline)**	**RDA2C (Ours)**	$\Delta\%$
Ant-v5	1953 ± 372	$\mathbf{2831 \pm 583}$	$\mathbf{+45.0\%}$
HalfCheetah-v5	1864 ± 78	$\mathbf{7099 \pm 645}$	$\mathbf{+280.9\%}$
Hopper-v5	2333 ± 590	$\mathbf{2607 \pm 272}$	$\mathbf{+11.7\%}$
Humanoid-v5	2370 ± 976	$\mathbf{3768 \pm 1290}$	$\mathbf{+59.0\%}$
HumanoidStandup-v5	$\mathbf{170412 \pm 8967}$	156068 ± 9614	-8.4%
InvertedPendulum-v5	989 ± 36	$\mathbf{999.2 \pm 2.6}$	$\mathbf{+1.1\%}$
Pusher-v5	-33.5 ± 2.2	$\mathbf{-30.6 \pm 3.0}$	$\mathbf{+8.9\%}$
Walker2d-v5	$\mathbf{3626 \pm 604}$	3546 ± 659	-2.2%

MuJoCo twin-Q critic comparison. Table 5 reports deterministic evaluation returns for the SAC-style twin-Q critic backbone. RDA2C and SAC share the replay and critic components, so the table isolates the effect of replacing the SAC actor improvement step with dual-score regression and entropy readout.

Table 5: **MuJoCo twin-Q critic results.** Best deterministic evaluation return ($\pm$ seed std) within the 1M-step budget over 10 seeds. Both methods use batch size 512, policy update frequency 1, 5 parallel environments, and 20 evaluation episodes; the CleanRL-default SAC comparison appears in Appendix B. Last column reports relative percent change against SAC (higher is better).

Environment	**SAC (Baseline)**	**RDA2C with twin-Q labels (Ours)**	$\Delta\%$
Ant-v5	$\mathbf{4389.60 \pm 1100.19}$	4240.94 ± 877.82	-3.4%
HalfCheetah-v5	$\mathbf{10118.79 \pm 894.32}$	9524.52 ± 1388.40	-5.9%
Hopper-v5	3345.29 ± 204.84	$\mathbf{3408.27 \pm 121.76}$	$\mathbf{+1.9\%}$
Humanoid-v5	$\mathbf{5376.44 \pm 175.72}$	5195.96 ± 314.71	-3.4%
HumanoidStandup-v5	158309.50 ± 4088.25	$\mathbf{158784.44 \pm 6868.90}$	$\mathbf{+0.3\%}$
InvertedPendulum-v5	$\mathbf{1000.00 \pm 0.00}$	$\mathbf{1000.00 \pm 0.00}$	0.0%
Pusher-v5	-29.58 ± 1.55	$\mathbf{-28.05 \pm 1.51}$	$\mathbf{+5.2\%}$
Walker2d-v5	4079.87 ± 371.70	$\mathbf{4504.91 \pm 788.33}$	$\mathbf{+10.4\%}$

AAPDA comparison. Table 6 reports the comparison with AAPDA described in Section 6: released defaults, 1M steps, ten paired training seeds, the same ten fixed evaluation seeds at every checkpoint, and the mean of the final five aligned deterministic checkpoints.

Table 6: **MuJoCo results against AAPDA.** Mean of the final five aligned deterministic checkpoints ($\pm$ seed std) over 10 paired seeds. Last column reports relative change $\Delta\% \triangleq 100 \cdot (\text{RDA2C} - \text{AAPDA})/(|\text{AAPDA}|)$.

Environment	AAPDA	RDA2C (Ours)	$\Delta\%$
Ant-v5	2562 ± 374	$\mathbf{4043 \pm 691}$	$\mathbf{+57.8\%}$
HalfCheetah-v5	4948 ± 705	$\mathbf{7808 \pm 629}$	$\mathbf{+57.8\%}$
Hopper-v5	2594 ± 794	$\mathbf{3218 \pm 196}$	$\mathbf{+24.0\%}$
Humanoid-v5	1209 ± 1335	$\mathbf{3653 \pm 1459}$	$\mathbf{+202.1\%}$
HumanoidStandup-v5	$\mathbf{162208 \pm 2025}$	156485 ± 9003	-3.5%
InvertedPendulum-v5	980.7 ± 49.6	$\mathbf{1000.0 \pm 0.0}$	$\mathbf{+2.0\%}$
Pusher-v5	-28.2 ± 1.1	$\mathbf{-27.1 \pm 1.7}$	$\mathbf{+3.6\%}$
Walker2d-v5	$\mathbf{4424 \pm 251}$	4001 ± 558	-9.6%

Baseline default configurations. The main-text comparisons use baselines matched to RDA2C's shared settings. For reference, this section reports both baselines at their CleanRL default configurations. For PPO the defaults are 2×64 trunks, learning rate 3×10^{-4}, gradient clip 0.5, and GAE $\lambda = 0.85$. For SAC they are batch size 256 and policy update frequency 2. Matching strengthens the baselines: the matched configuration improves PPO over its defaults on five of eight tasks, and the batch-512, frequency-1 SAC matches or improves the default SAC on seven of eight tasks.

Table 7: **MuJoCo results against CleanRL-default PPO.** Mean episodic return ($\pm$ std) over 10 seeds. Same RDA2C runs as Table 4.

Environment	PPO (default)	RDA2C (Ours)	$\Delta\%$
Ant-v5	1855 ± 519	$\mathbf{2831 \pm 583}$	$\mathbf{+52.6\%}$
HalfCheetah-v5	1884 ± 78	$\mathbf{7099 \pm 645}$	$\mathbf{+276.8\%}$
Hopper-v5	2588 ± 831	$\mathbf{2607 \pm 272}$	$\mathbf{+0.7\%}$
Humanoid-v5	1100 ± 388	$\mathbf{3768 \pm 1290}$	$\mathbf{+242.5\%}$
HumanoidStandup-v5	$\mathbf{158572 \pm 12341}$	156068 ± 9614	-1.6%
InvertedPendulum-v5	$\mathbf{1000.0 \pm 0.0}$	999.2 ± 2.6	-0.1%
Pusher-v5	-36.2 ± 2.0	$\mathbf{-30.6 \pm 3.0}$	$\mathbf{+15.6\%}$
Walker2d-v5	3450 ± 475	$\mathbf{3546 \pm 659}$	$\mathbf{+2.8\%}$

Table 8: **MuJoCo twin-Q results against CleanRL-default SAC** (batch 256, policy frequency 2). Best deterministic evaluation return ($\pm$ seed std) within the 1M-step budget over 10 seeds. Same RDA2C runs as Table 5.

Environment	SAC (default)	RDA2C with twin-Q labels (Ours)	$\Delta\%$
Ant-v5	4048.10 ± 859.12	$\mathbf{4240.94 \pm 877.82}$	$\mathbf{+4.8\%}$
HalfCheetah-v5	$\mathbf{9844.07 \pm 914.73}$	9524.52 ± 1388.40	-3.2%
Hopper-v5	3348.58 ± 157.32	$\mathbf{3408.27 \pm 121.76}$	$\mathbf{+1.8\%}$
Humanoid-v5	$\mathbf{5228.44 \pm 241.40}$	5195.96 ± 314.71	-0.6%
HumanoidStandup-v5	156360.89 ± 7197.51	$\mathbf{158784.44 \pm 6868.90}$	$\mathbf{+1.5\%}$
InvertedPendulum-v5	$\mathbf{1000.00 \pm 0.00}$	$\mathbf{1000.00 \pm 0.00}$	0.0%
Pusher-v5	-29.80 ± 1.55	$\mathbf{-28.05 \pm 1.51}$	$\mathbf{+5.9\%}$
Walker2d-v5	3871.58 ± 736.89	$\mathbf{4504.91 \pm 788.33}$	$\mathbf{+16.4\%}$

Atari GAE label comparison. Table 9 reports the twelve Atari games underlying Figure 3.

Table 9: **Atari results.** Mean running return ($\pm$ seed std) over 10 seeds on twelve Atari games with clipped dual regression, $\epsilon_{\text{clip}} = 1.0$, $\beta = 0.6$, and $\lambda = 0.005$. Last column reports relative change $\Delta\% \triangleq 100 \cdot (\text{RDA2C} - \text{PPO})/(|\text{PPO}|)$ when PPO $\neq 0$.

Environment	**PPO (Baseline)**	**RDA2C (Ours)**	$\Delta\%$
Asterix	4061 ± 1014	$\mathbf{6024 \pm 1336}$	$\mathbf{+48.3\%}$
BeamRider	2540 ± 316	$\mathbf{9142 \pm 786}$	$\mathbf{+259.9\%}$
Breakout	$\mathbf{422 \pm 24}$	408 ± 29	-3.5%
Enduro	980 ± 135	$\mathbf{5348 \pm 1720}$	$\mathbf{+445.6\%}$
Freeway	$\mathbf{32.7 \pm 0.6}$	23.6 ± 16.3	-27.8%
Gravitar	686 ± 187	$\mathbf{835 \pm 376}$	$\mathbf{+21.8\%}$
MsPacman	2085 ± 268	$\mathbf{3243 \pm 936}$	$\mathbf{+55.6\%}$
Pong	$\mathbf{20.6 \pm 0.7}$	20.5 ± 1.4	-0.6%
PrivateEye	70.1 ± 47.1	$\mathbf{89.8 \pm 31.6}$	$\mathbf{+28.2\%}$
Qbert	$\mathbf{15804 \pm 1993}$	14562 ± 1283	-7.9%
Seaquest	1541 ± 427	$\mathbf{9114 \pm 7642}$	$\mathbf{+491.4\%}$
SpaceInvaders	998 ± 187	$\mathbf{1373 \pm 220}$	$\mathbf{+37.6\%}$

C Additional Experimental Results

C.1 MuJoCo GAE label learning curves

Figure 4 displays the 10-seed training dynamics for the PPO versus clipped-dual-regression RDA2C comparison across the 8 continuous control environments. The curves highlight large gains on `HalfCheetah` and `Humanoid`, a clear improvement on `Ant`, and PPO-level behavior on the remaining tasks, with `HumanoidStandup` remaining slightly below PPO in the final aggregate table.

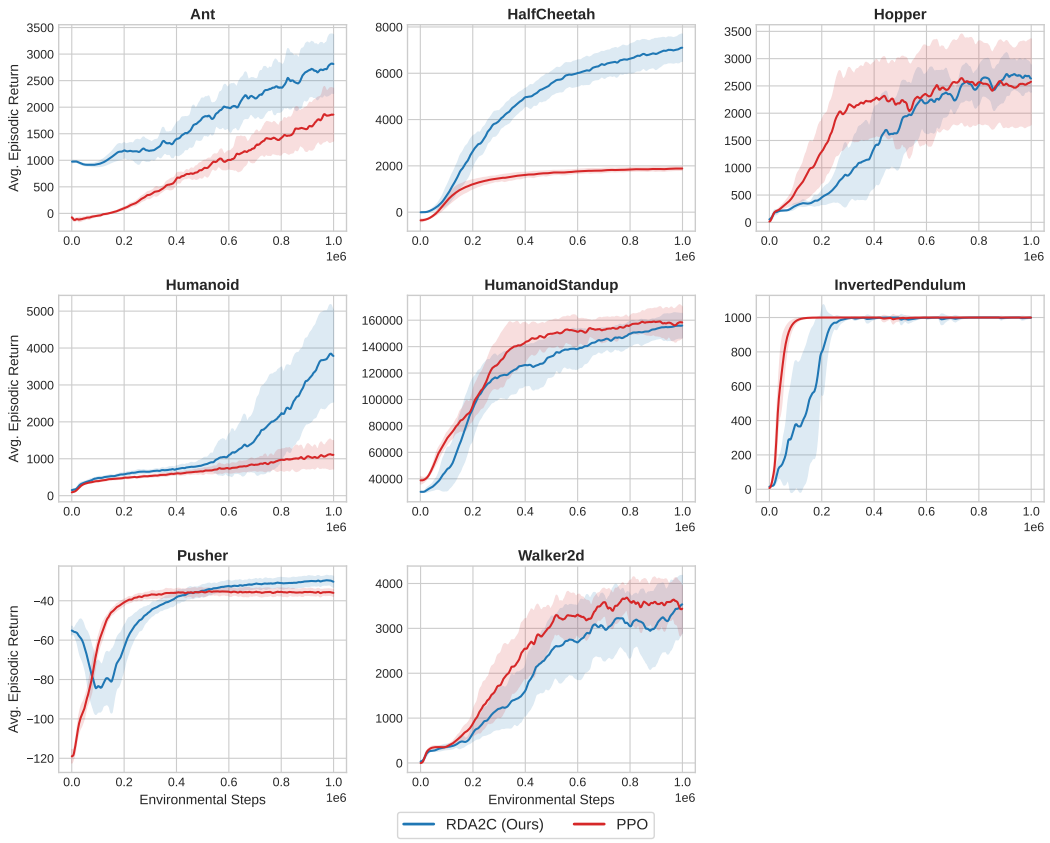


Figure 4: **PPO versus RDA2C MuJoCo learning curves.** Blue denotes RDA2C and red denotes PPO. Shaded regions show standard deviation over 10 seeds. The clipped-dual-regression configuration yields large gains on `HalfCheetah`, `Humanoid`, and `Ant`, while remaining close to PPO on the other environments.

C.2 MuJoCo twin-Q critic label learning curves

Figure 5 presents the 10-seed online training-return curves for the SAC versus RDA2C with twin-Q labels comparison. These curves correspond to the logged running reward summaries in Table 10, not to the deterministic evaluation returns in Table 5. Across environments, the off-policy RDA actor update with soft twin-Q RLOO labels is generally stronger during data collection: RDA2C with twin-Q labels has higher final running reward on seven of eight tasks and lower seed standard deviation on six of eight tasks. The largest online return gaps appear on `Hopper-v5` and `Walker2d-v5`. `HalfCheetah-v5` remains the main environment where SAC is clearly stronger.

Table 10: **MuJoCo online training return with twin-Q critic labels.** Final logged running reward ($\pm$ seed std) over 10 seeds at 1,000,000 environment steps. This is an online data collection metric, distinct from the deterministic evaluation return in Table 5. Last column reports relative percent change against SAC (higher is better).

Environment	SAC (Baseline)	RDA2C with twin-Q labels (Ours)	$\Delta\%$
Ant-v5	3414.91 ± 873.59	$\mathbf{3683.40 \pm 842.94}$	$\mathbf{+7.9\%}$
HalfCheetah-v5	$\mathbf{9182.51 \pm 888.07}$	8538.56 ± 1167.47	-7.0%
Hopper-v5	2289.03 ± 637.75	$\mathbf{2948.40 \pm 357.63}$	$\mathbf{+28.8\%}$
Humanoid-v5	4771.82 ± 320.91	$\mathbf{4979.12 \pm 215.64}$	$\mathbf{+4.3\%}$
HumanoidStandup-v5	152428.66 ± 7472.47	$\mathbf{155100.32 \pm 5994.39}$	$\mathbf{+1.8\%}$
InvertedPendulum-v5	935.64 ± 72.85	$\mathbf{969.61 \pm 33.70}$	$\mathbf{+3.6\%}$
Pusher-v5	-30.01 ± 1.24	$\mathbf{-28.87 \pm 1.50}$	$\mathbf{+3.8\%}$
Walker2d-v5	3124.79 ± 1354.77	$\mathbf{4067.66 \pm 1117.91}$	$\mathbf{+30.2\%}$

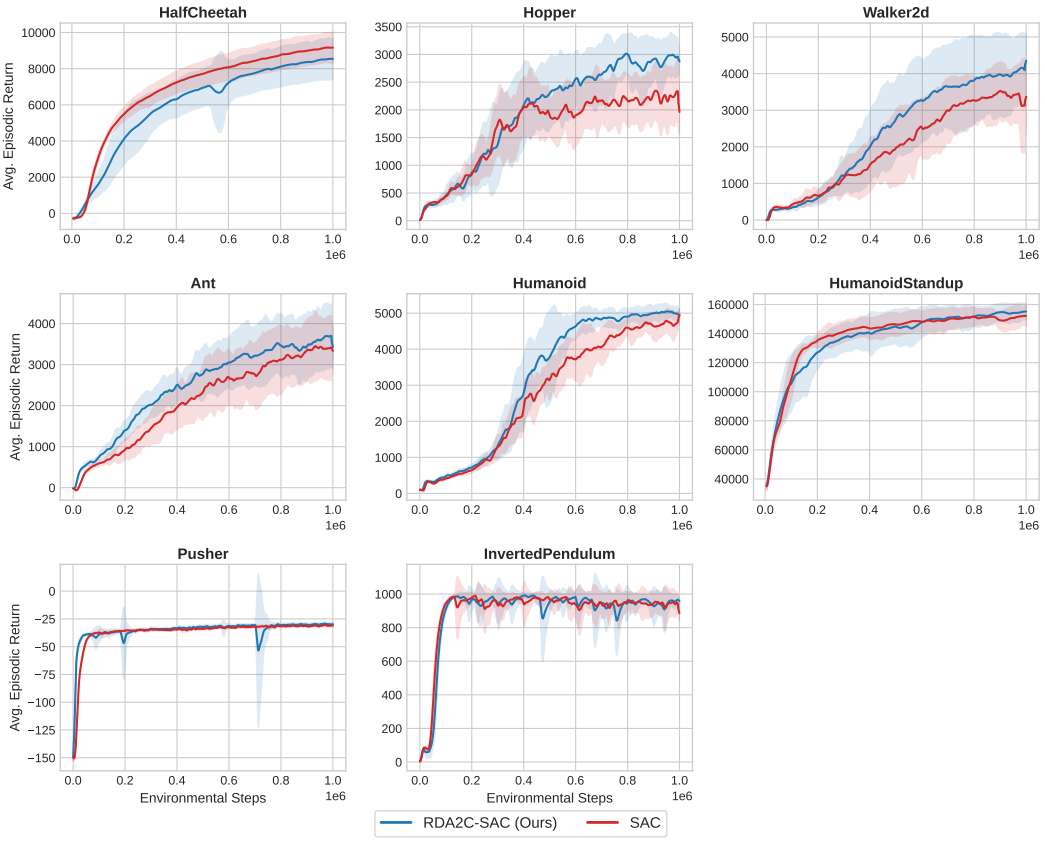


Figure 5: **MuJoCo online training curves with twin-Q critic labels.** Blue denotes RDA2C twin-Q and red denotes SAC. Shaded regions show standard deviation over 10 seeds. The curves report the online running reward metric logged during data collection. RDA2C twin-Q achieves higher final running reward on Ant, Hopper, Humanoid, HumanoidStandup, InvertedPendulum, Pusher, and Walker2d, while SAC remains stronger on HalfCheetah.

C.3 Atari Learning Curves

Figure 6 presents the learning dynamics for the twelve Atari games used in the main comparison. Curves represent the average episodic return over 10 random seeds with standard deviation shading.

Analysis. The curves give a more detailed view of the summary in the main text. RDA2C improves substantially over PPO on `Seaquest`, `BeamRider`, `Enduro`, and `Asterix`. In these runs, the replayed dual averaging update continues to improve after the PPO curve has largely flattened.

The remaining games include threshold and saturated regimes. `Pong` is near saturated for both methods, while `Freeway` remains a high variance threshold game where PPO is higher in the final 10-seed aggregate. RDA2C improves `PrivateEye`, but it does not add a directed exploration mechanism.

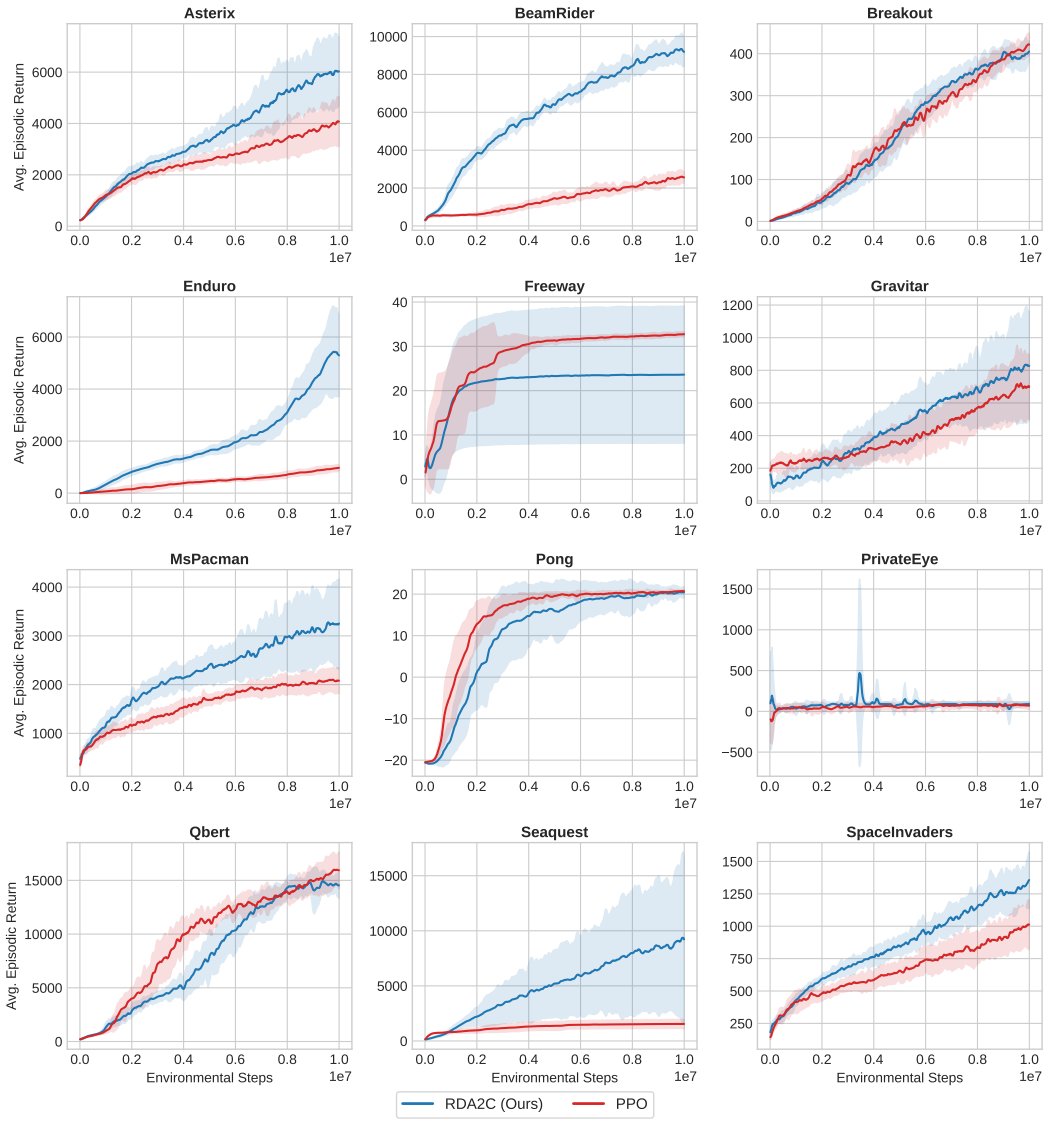


Figure 6: **Full Training Dynamics (10M Frames).** Comparison of RDA2C and PPO across the twelve Atari games used in the main comparison.

D Ablation Studies

This section reports the ablations: the replay window (Section D.1), the RDA stabilizer (Section D.2), the entropy floor (Section D.3), the advantage-label loss (Section D.4), the SAC replay buffer (Section D.5), and a PPO capacity control (Section D.6).

D.1 Replay Window Size

We evaluate sensitivity to the replay / label window size W in both the GAE Atari setting and the twin-Q critic label MuJoCo setting. The two sweeps differ because the Atari implementation stores replayed transitions, whereas the twin-Q critic implementation stores labeled (s, a, g) tuples in a dedicated FIFO actor regression buffer.

This sweep should be interpreted qualitatively. The theoretical discussion in Section 5 predicts a variance–staleness tradeoff and, in a simplified asymptotic bound, balances the replay error terms with $W_t \asymp t^{2/3}$. Our finite budget sweeps do not estimate this exponent: they vary fixed window sizes and report achieved return, and they do not measure score estimation error directly. The empirical question is therefore whether small windows are too noisy and whether very long memories can become stale.

The replay window controls how much past dual evidence defines the actor objective. Small windows use fresher labels but provide fewer samples for fitting the dual score, while large windows reduce label variance but mix in older policy-improvement signals. The Atari sweep in Appendix D.1 shows this finite-budget tradeoff, with intermediate windows often strongest. The best window varies across games and is not monotone in W. This behavior qualitatively matches the finite replay and local drift terms in Theorem 5.1. The twin-Q MuJoCo sweep (Appendix D.1.2) shows the same nonmonotone pattern, and the one-round current-label setting is never the best window. A current-label control for the GAE setting (Appendix D.1.3) isolates retention alone: the historical target is higher on seven of eight tasks.

D.1.1 Atari

We evaluate the sensitivity of RDA2C to the Atari replay window size W. Figure 7 shows the learning curves, and Table 11 provides the final asymptotic performance. The window sweep uses the plain squared dual regression loss, whereas the main Atari comparison in Table 9 uses the clipped loss of Appendix A.4, so the two tables are not comparable at $W = 500{,}000$. Only the comparison across windows within the sweep is meaningful.

Table 11: **Atari Window Size Sweep Results.** Mean episodic return at 10M frames, plain squared loss. **Bold** indicates the best performing configuration.

Environment	$W = 300,000$	$W = 500,000$	$W = 700,000$	$W = 1,000,000$
Asterix	$\mathbf{5448 \pm 1063}$	5305 ± 204	4758 ± 929	4122 ± 505
BeamRider	9768 ± 429	$\mathbf{10509 \pm 874}$	9823 ± 317	9831 ± 832
Breakout	$\mathbf{369 \pm 17}$	353 ± 6	319 ± 22	314 ± 31
Enduro	$\mathbf{3474 \pm 372}$	3374 ± 273	3077 ± 520	3076 ± 340
Freeway	33.8 ± 0.0	33.8 ± 0.0	33.8 ± 0.0	$\mathbf{33.9 \pm 0.0}$
Gravitar	690 ± 93	$\mathbf{830 \pm 169}$	675 ± 40	670 ± 65
MsPacman	2788 ± 576	$\mathbf{3974 \pm 669}$	3056 ± 802	3245 ± 428
Pong	20.9 ± 0.0	$\mathbf{21.0 \pm 0.0}$	$\mathbf{21.0 \pm 0.0}$	20.9 ± 0.1
PrivateEye	66.6 ± 47.1	66.6 ± 47.1	$\mathbf{100.0 \pm 0.0}$	99.9 ± 0.0
Qbert	15341 ± 392	15690 ± 309	14310 ± 1150	$\mathbf{15892 \pm 929}$
Seaquest	5484 ± 1819	$\mathbf{14158 \pm 5832}$	13335 ± 4911	13843 ± 4220
SpaceInvaders	1359 ± 44	$\mathbf{1457 \pm 237}$	1361 ± 188	1405 ± 172

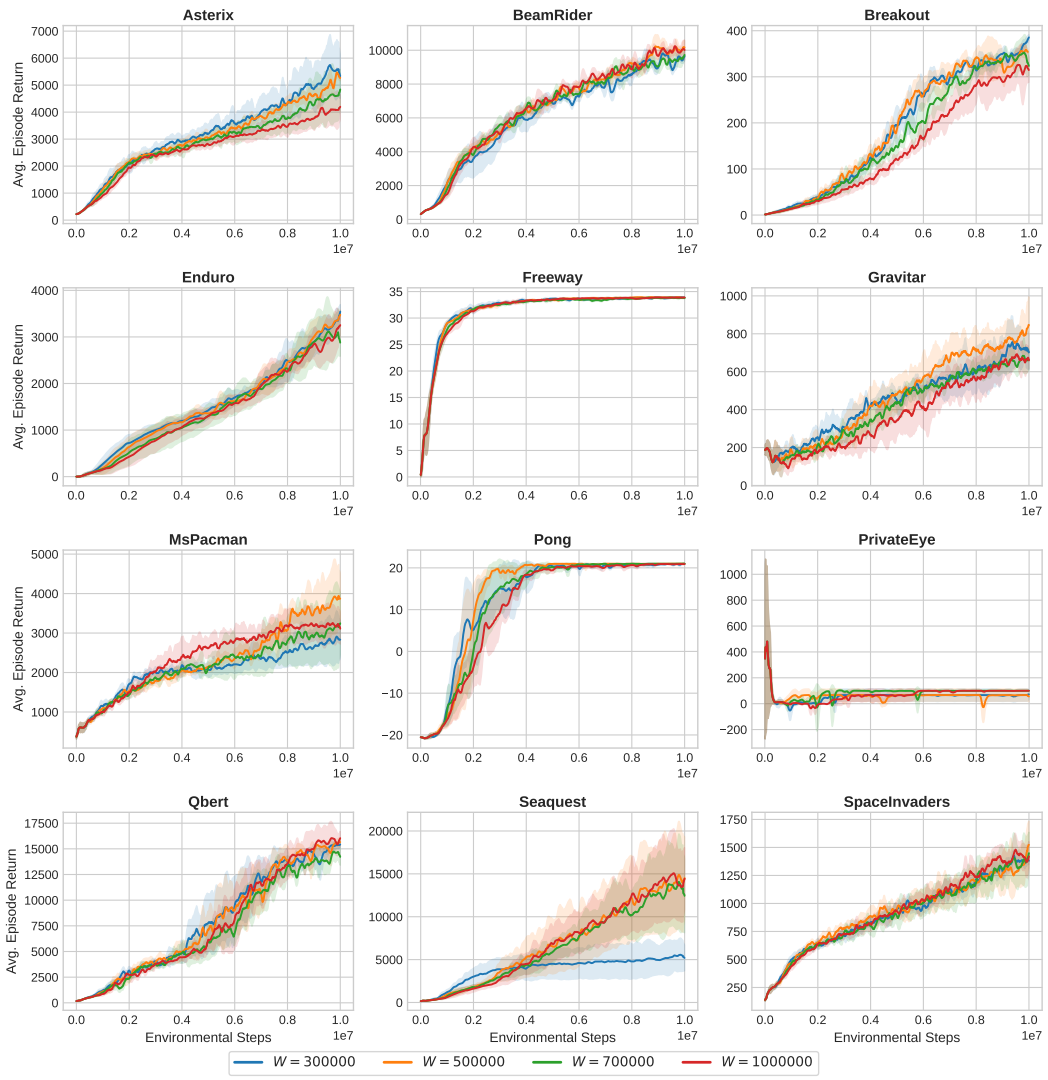


Figure 7: **Effect of Window Size (W) on Atari.** Training curves for $W \in \{300\text{k}, 500\text{k}, 700\text{k}, 1\text{M}\}$ on the twelve Atari games used in the main comparison. $W = 500\text{k}$ offers the most consistent performance across tasks.

D.1.2 MuJoCo with twin-Q Critics

We additionally sweep the RDA label buffer size on five representative MuJoCo tasks: `Ant-v5`, `HalfCheetah-v5`, `Hopper-v5`, `Humanoid-v5`, and `Walker2d-v5`. We summarize each setting by the best deterministic evaluation mean return achieved during training, averaged over 10 seeds. In this implementation, each actor trigger samples $B = 512$ replay states and generates $K = 2$ labels per state, so it inserts $KB = 1024$ fresh (s, a, g) tuples into the FIFO label buffer. Thus $W = 1024$ acts as a one round current-label control. Table 12 shows a nonmonotone mean return pattern: the current-label control is not the top mean setting on any of the five tasks, larger windows substantially improve `HalfCheetah-v5` and moderately improve `Walker2d-v5`, and `Ant-v5`, `Hopper-v5`, and `Humanoid-v5` are comparatively flat once $W > 1024$. This is consistent with the qualitative bias–variance/staleness tradeoff from Section 5: averaging recent actor-improvement labels can reduce noise, but overly long memories can become stale.

Table 12: **MuJoCo window size sweep with twin-Q critic labels.** Mean best deterministic evaluation return ($\pm$ seed std) over 10 seeds. Here W is the FIFO label buffer size in stored (s, a, g) tuples. With batch size 512 and $K = 2$ label samples, $W = 1024$ retains approximately one actor trigger's fresh labels and therefore acts as a current label control. **Bold** indicates the best window per environment.

Environment	$W = 1024$	$W = 2048$	$W = 4096$	$W = 8192$	$W = 12{,}800$	$W = 25{,}600$
Ant-v5	4755 ± 985	4008 ± 953	4314 ± 909	$\mathbf{4798 \pm 747}$	4241 ± 878	4672 ± 946
HalfCheetah-v5	7068 ± 3653	8800 ± 2565	9387 ± 1542	8081 ± 3410	9525 ± 1388	$\mathbf{9819 \pm 1029}$
Hopper-v5	3404 ± 195	3330 ± 111	3365 ± 207	3351 ± 133	3408 ± 122	$\mathbf{3430 \pm 150}$
Humanoid-v5	5280 ± 110	5237 ± 159	5279 ± 200	$\mathbf{5294 \pm 127}$	5196 ± 315	5218 ± 107
Walker2d-v5	4444 ± 459	3810 ± 1291	$\mathbf{4724 \pm 522}$	3566 ± 1302	4505 ± 788	4179 ± 977

D.1.3 CURRENT-LABEL CONTROL (GAE MUJOCO)

This control changes label retention and nothing else. The buffer size $W = 2048$ retains exactly one rollout of labels, so every regression target comes from the current policy. The main-results setting $W = 50{,}000$ retains about 24 rollouts. Architecture, optimizer state, and training budget are fixed. Table 13 shows the result over ten paired seeds: the historical target is higher on seven of eight tasks. The gains are large where they appear: `Humanoid-v5` goes from 500 to 3768 and `Ant-v5` from 884 to 2831. This is the most direct evidence that the improvement comes from aggregating labels across several recent policies. Score regression on current labels alone does not produce it.

Table 13: **Current-label control (GAE labels, MuJoCo).** Mean running episodic return ($\pm$ seed std) over 10 paired seeds. Last column reports relative change $\Delta\% \triangleq 100 \cdot (\text{history} - \text{current})/(|\text{current}|)$ (higher is better).

Environment	**current** ($W = 2048$)	**history** ($W = 50{,}000$)	$\Delta\%$
Ant-v5	884 ± 412	$\mathbf{2831 \pm 583}$	$\mathbf{+220.1\%}$
HalfCheetah-v5	5162 ± 1298	$\mathbf{7099 \pm 645}$	$\mathbf{+37.5\%}$
Hopper-v5	2112 ± 802	$\mathbf{2607 \pm 272}$	$\mathbf{+23.4\%}$
Humanoid-v5	500 ± 75	$\mathbf{3768 \pm 1290}$	$\mathbf{+654.0\%}$
HumanoidStandup-v5	94062 ± 24669	$\mathbf{156068 \pm 9614}$	$\mathbf{+65.9\%}$
InvertedPendulum-v5	$\mathbf{1000.0 \pm 0.0}$	999.2 ± 2.6	-0.1%
Pusher-v5	-31.9 ± 1.3	$\mathbf{-30.6 \pm 3.0}$	$\mathbf{+4.3\%}$
Walker2d-v5	2973 ± 707	$\mathbf{3546 \pm 659}$	$\mathbf{+19.3\%}$

D.2 RDA Stabilizer

Figure 8 varies the RDA stabilizer β while holding the entropy floor and replay window fixed. Larger values damp early policy changes more strongly. The aggregate results show that some transient stabilization is useful, with moderate values performing best, while the per-task heatmaps show that the best β is environment dependent. Very large β can make the early readout too conservative, whereas $\beta = 0$ removes this transient damping entirely.

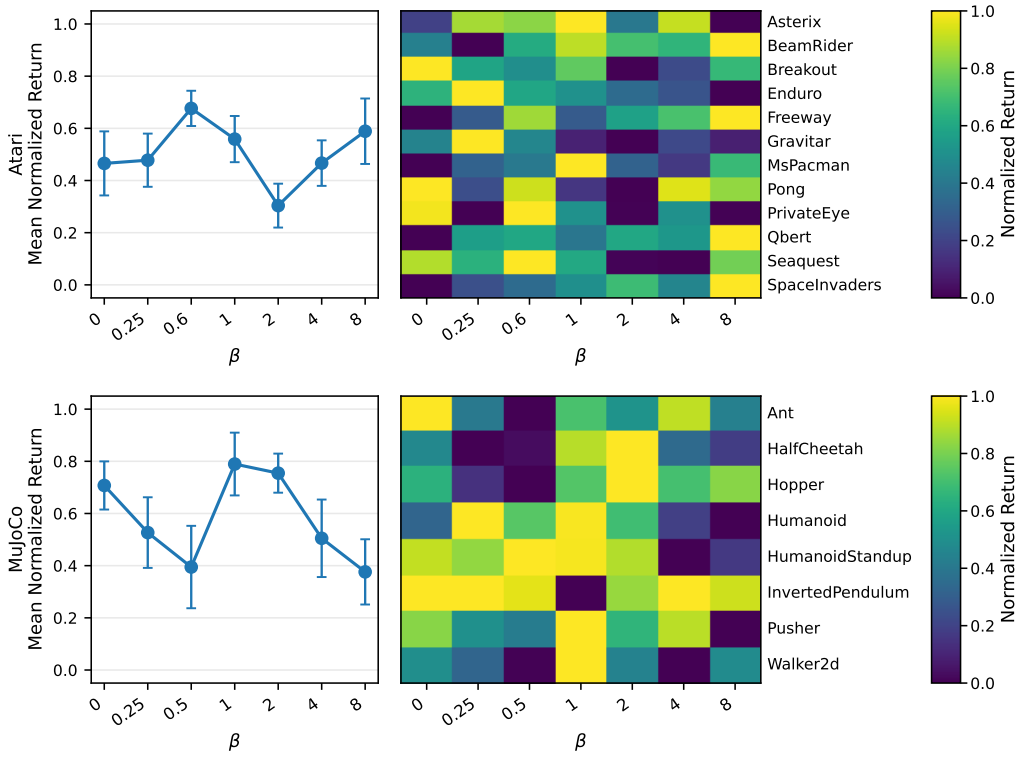


Figure 8: **Sensitivity to the RDA stabilizer β.** Returns are normalized independently within each environment across the swept β values before aggregation. Error bars in the aggregate panels denote task SEM. For the reason in footnote 1, they do not denote confidence intervals. They illustrate the scale of variation across tasks.

D.3 Entropy Floor Sensitivity

We sweep the persistent entropy floor λ in the RDA temperature $\alpha_t = \lambda + \beta/t$ while holding the replay window and β fixed. Figure 9 shows a broad small λ plateau on Atari and a moderate λ plateau on MuJoCo. Very large entropy floors keep the readout too diffuse and hurt return.

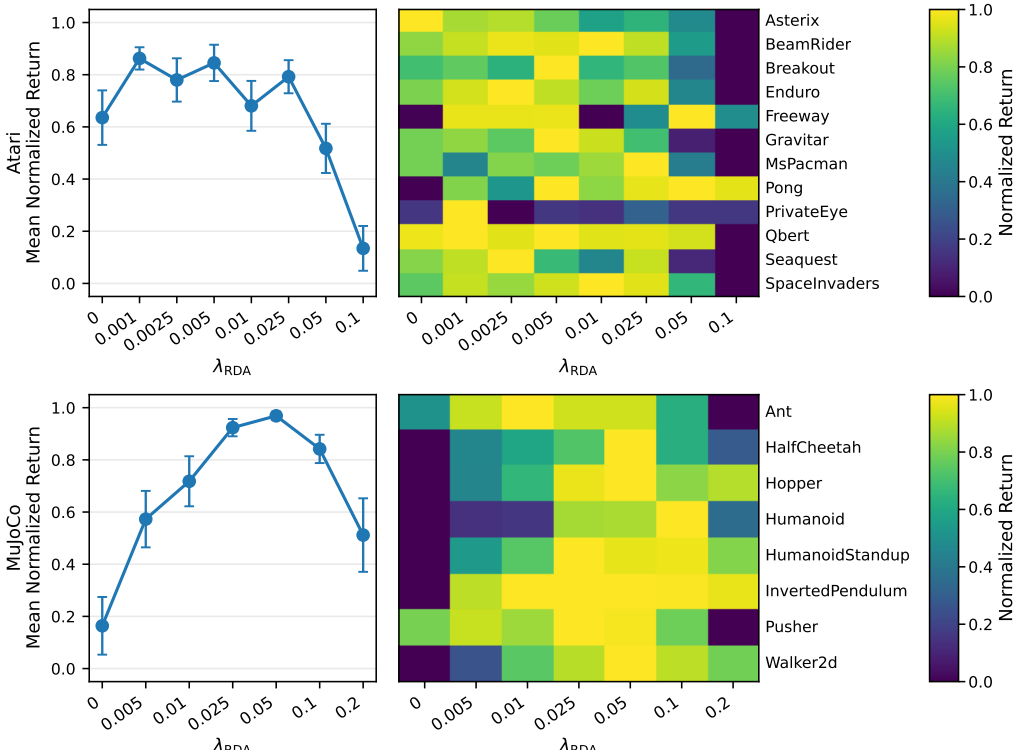


Figure 9: **Sensitivity to the entropy floor λ.** Returns are min-max normalized independently within each environment across the swept λ values before aggregation. Error bars in the aggregate panels denote task SEM. For the reason in footnote 1, they do not denote confidence intervals. They illustrate the scale of variation across tasks.

D.4 Advantage-Label Loss

We compare four dual regression losses for GAE advantage labels: plain MSE, Huber regression, normalized target clipping, and PPO style prediction clipping. Figure 10 reports Atari and MuJoCo separately. Prediction clipping is the strongest default, especially on MuJoCo. Target clipping is competitive on Atari but is less reliable in continuous control. Huber regression alone does not recover the performance of prediction clipping.

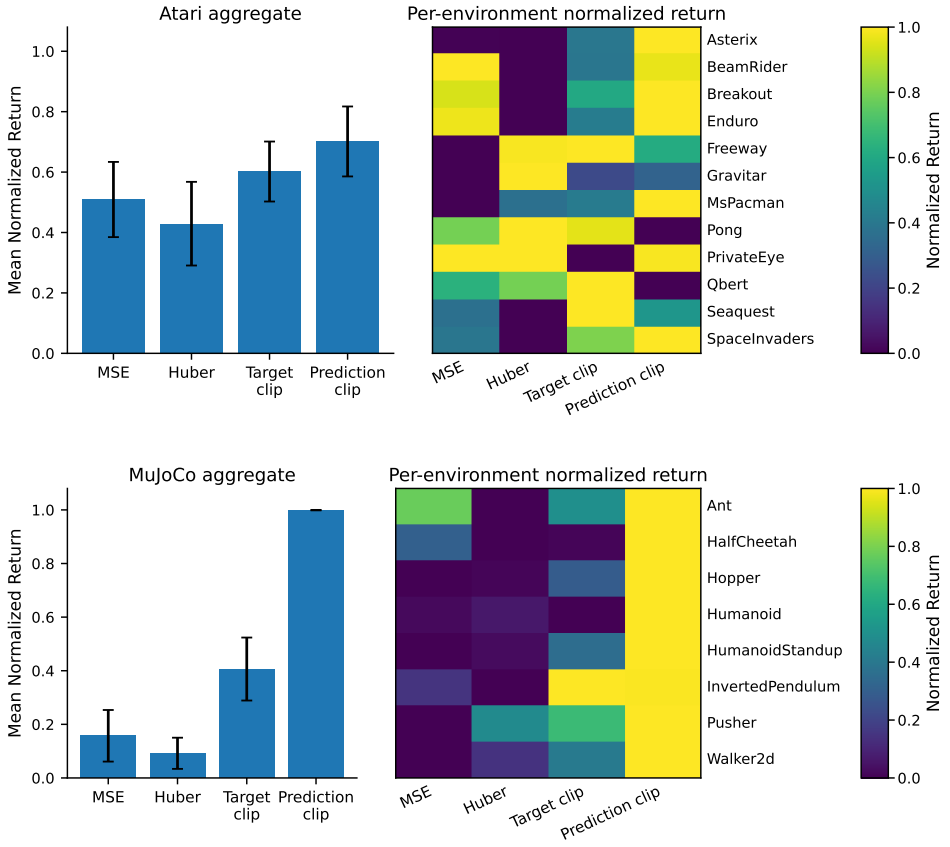


Figure 10: **Advantage label loss ablation.** Atari results are shown above and MuJoCo results below. Returns are min-max normalized within each environment across loss types before aggregation. Error bars in the aggregate panels denote task SEM. For the reason in footnote 1, they do not denote confidence intervals. They illustrate the scale of variation across tasks. Prediction clipping gives the best aggregate, while Huber alone is weaker.

D.5 SAC Replay Buffer Size

Sensitivity to the shared transition replay buffer. We examine sensitivity to the capacity of the ordinary transition replay buffer used by the shared SAC/twin-Q backbone. In the main comparison, we retain the 1M capacity from the default SAC configuration used in our implementation and apply the same capacity to both SAC and RDA2C. This setting preserves comparability with the standard baseline. It was not selected using the present sweep. For this diagnostic ablation, we vary the shared transition-buffer capacity over 50k, 100k, 250k, 500k, and 1M, while holding RDA2C's separate advantage-label buffer fixed at 12,800. We normalize jointly across both methods and all tested capacities within each environment, so the aggregate summarizes sensitivity to the shared replay capacity and the method gap on a common scale. Figure 11 shows that aggregate normalized performance peaks near 100k for both methods and decreases for larger capacities. The similar dependence across SAC and RDA2C suggests that this effect comes primarily from recency in the shared transition-replay backbone and not from the RDA-specific label buffer, which is held fixed. We retain 1M in the main comparison to report results under the default SAC configuration.

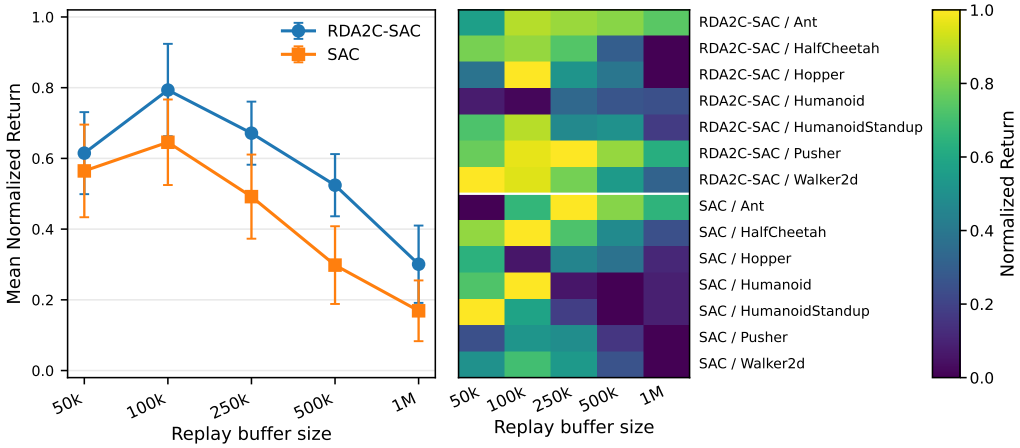


Figure 11: **Sensitivity to standard SAC replay buffer size.** We vary the ordinary SAC replay buffer and keep the RDA label window fixed at 12,800. Returns are min-max normalized independently within each environment across all method and buffer configurations before aggregation. Error bars denote task SEM. InvertedPendulum is omitted from the normalized aggregate because it is saturated for every setting.

D.6 PPO Network Capacity Control (MuJoCo, GAE)

The matched comparison in Table 4 sets PPO to RDA2C's width and learning rate. This ablation asks the complementary question: does capacity alone close the gap when PPO keeps its own defaults? Starting from the CleanRL default 2×64 trunk (Appendix B), we retrain PPO with a 2×128 trunk, keeping the global log-standard-deviation and the remaining default hyperparameters. We report both the default learning rate 3×10^{-4} and the RDA2C rate 2×10^{-4}. Table 14 and Figure 12 show that enlarging PPO does not close the gap: at 3×10^{-4} the wider PPO is often worse than the 2×64 baseline (especially on `Ant-v5` and `Hopper-v5`), and lowering the learning rate recovers some of that loss but still leaves RDA2C ahead on the tasks with the largest gains.

Table 14: **PPO network capacity control (MuJoCo, GAE labels).** Mean running episodic return ($\pm$ seed std) over 10 seeds. All PPO variants use global log-standard-deviation. **Bold** marks the best mean in each row.

Environment	**PPO** 2×64 **lr** 3×10^{-4}	**PPO** 2×128 **lr** 3×10^{-4}	**PPO** 2×128 **lr** 2×10^{-4}	**RDA2C** 2×128 **lr** 2×10^{-4}
Ant-v5	1855 ± 519	1260 ± 441	1537 ± 757	$\mathbf{2831 \pm 583}$
HalfCheetah-v5	1884 ± 78	1940 ± 69	2190 ± 1059	$\mathbf{7099 \pm 645}$
Hopper-v5	2588 ± 831	2154 ± 719	2363 ± 464	$\mathbf{2607 \pm 272}$
Humanoid-v5	1100 ± 388	1187 ± 529	1573 ± 901	$\mathbf{3768 \pm 1290}$
HumanoidStandup-v5	158572 ± 12341	$\mathbf{161394 \pm 6265}$	159009 ± 8458	156068 ± 9614
InvertedPendulum-v5	$\mathbf{1000.0 \pm 0.0}$	952.6 ± 149.9	997.3 ± 8.4	999.2 ± 2.6
Pusher-v5	-36.2 ± 2.0	-36.2 ± 2.1	-36.5 ± 2.4	$\mathbf{-30.6 \pm 3.0}$
Walker2d-v5	3450 ± 475	3262 ± 951	$\mathbf{3812 \pm 675}$	3546 ± 659

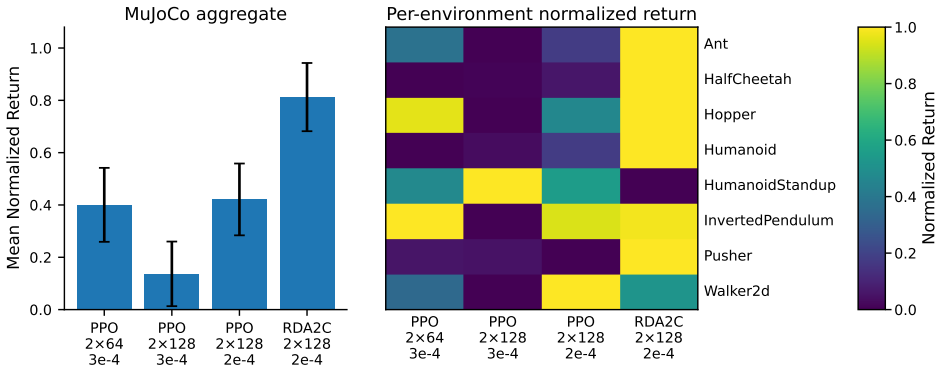


Figure 12: **PPO network capacity control on MuJoCo (GAE labels).** Returns are min–max normalized within each environment across the four configurations before aggregation. Error bars in the aggregate panel denote task SEM. Matching PPO's width to RDA2C does not reproduce the RDA2C gains.

E RDA2C Gradient Boosting with XGBoost

E.1 Overview

To complement our neural network instantiation of the dual predictor Z_θ, we performed a small study in which Z_θ is instantiated as a gradient boosted decision tree regressor (XGBoost). This setting makes the connection between dual averaging and additive function approximation particularly explicit, and it allows us to compare two concrete approximations of the running dual average: `xgb_boosting` (incremental boosting of the cumulative dual) and `xgb_dagger` (DAgger style dataset aggregation with a FIFO buffer).

Supervised representation. In both variants, we train a regressor on features $\phi(s, a) = [s, \text{onehot}(a)]$ and targets given by an advantage label derived from a learned value function. The

resulting predictor $\widehat{Z}(s,a)$ is converted into a policy by the closed-form mirror map:

$$\pi(a \mid s) \propto \exp\left(\frac{\widehat{Z}(s,a)}{\alpha}\right),$$

using the same entropy regularized Boltzmann form as in the main algorithm. In all runs reported here, we apply clipped importance sampling (IPS) by reweighting advantage targets by $1/\pi_{\mathrm{beh}}(a \mid s)$ with a maximum weight (`iw_max`). See the configuration tables below.

Incremental boosting (`xgb_boosting`). We maintain a single XGBoost booster whose prediction represents an *unnormalized* cumulative dual $U(s,a) \approx \sum_{k=1}^{K} g_k(s,a)$, where g_k is the (normalized and optionally IPS weighted) advantage signal from update k. Operationally, after each minibatch we add one tree (warm start) using a *dual target* $y = U_{\mathrm{prev}} + g$ so that the new tree fits the residual advantage. At action selection time, logits are formed as $U(s,a)/(K\lambda + \beta)$, where K is the current number of boosted rounds. This matches the scaling implied by the ratio Z_t/α_t in (3) when identifying K with the number of dual updates.

Dataset aggregation (`xgb_dagger`). We maintain a FIFO buffer of advantage labeled tuples $(s, a, \hat{A})$ collected from all past rollouts (up to a fixed capacity), and periodically continue boosting on the full buffer. This corresponds to fitting a single predictor to the pooled dataset, which approximates the average dual Z_t under the mixture of visited state–action distributions. The policy uses the standard temperature schedule $\alpha_t = \lambda + \beta/t$ and applies the same softmax mapping.

E.2 Hyperparameters

Experimental protocol. We ran both variants for 10^6 environment steps on `CartPole-v1` and `Acrobot-v1`, using 8 parallel environments with rollout length 256 (batch size 2048). We report mean $\pm$ standard deviation of the final episodic return over 5 random seeds.

Hyperparameters. Tables 15 and 16 list the configuration used for each environment.

Table 15: **XGBoost dataset aggregation (`xgb_dagger`) configurations.**

Parameter	CartPole-v1	Acrobot-v1
Total timesteps	1,000,000	1,000,000
Num envs / steps	8 / 256	8 / 256
Batch size	2048	2048
γ / GAE λ_{GAE}	0.99 / 0.8	0.99 / 0.8
Value LR / epochs	2.5×10^{-3} / 4	2.5×10^{-3} / 4
Anneal LR	`True`	`False`
Max grad norm	0.5	0.5
FIFO capacity	100,000	200,000
Fit every (iters)	4	2
Advantage clip	10	25
IPS / `iw_max`	`True` / 10	`True` / 5
α_t schedule	λ=0.15, β=2	λ=0.2, β=2
Trees per fit	1	2
Tree LR (`eta`)	0.05	0.05
Max depth	3	7
Min child weight	3	3
Subsample / colsample	1.0 / 1.0	1.0 / 1.0

Table 16: **XGBoost incremental boosting (`xgb_boosting`) configurations.**

Parameter	CartPole-v1	Acrobot-v1
Total timesteps	1,000,000	1,000,000
Num envs / steps	8 / 256	8 / 256
Batch size	2048	2048
γ / GAE λ_{GAE}	0.99 / 0.8	0.99 / 0.8
Value LR / epochs	3×10^{-3} / 1	10^{-2} / 1
Anneal LR	`True`	`True`
Max grad norm	10	10
Normalize adv	`True`	`True`
Advantage clip	5	5
IPS / `iw_max`	`True` / 10	`True` / 10
α_t schedule	λ=10^{-4}, β=1	λ=10^{-4}, β=1
Tree LR (`eta`)	0.5	0.45
Max depth	3	3
Min child weight	5	5
Subsample / colsample	1.0 / 1.0	1.0 / 1.0

E.3 Results

Table 17: **RDA2C gradient boosting results.** Final episodic return (mean $\pm$ std over 5 seeds).

Environment	Method	Final return
CartPole-v1	`xgb_dagger`	500.00 ± 0.00
CartPole-v1	`xgb_boosting`	500.00 ± 0.00
Acrobot-v1	`xgb_dagger`	-84.15 ± 4.57
Acrobot-v1	`xgb_boosting`	-88.98 ± 2.43

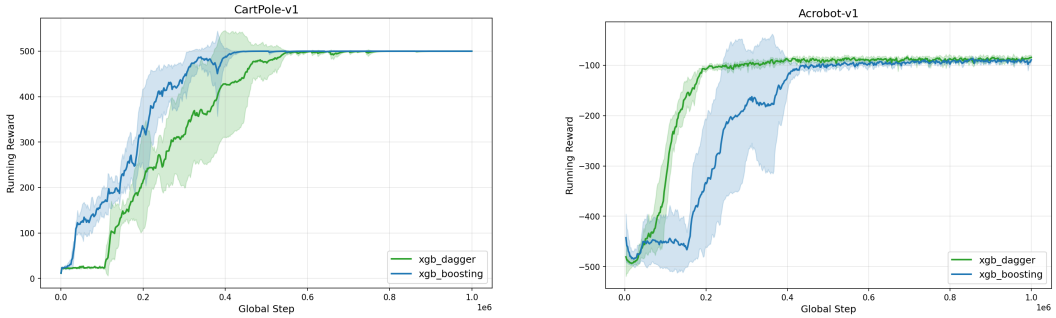


Figure 13: **RDA2C gradient boosting learning curves.** On CartPole-v1 (left), both `xgb_dagger` and `xgb_boosting` reach 500 within 10^6 steps. On Acrobot-v1 (right), `xgb_dagger` is slightly more sample efficient early, and both methods converge similarly by 10^6 steps.

F PROOFS AND IMPLICATIONS FOR POLICY PERFORMANCE

The proof below analyzes the learned replay actor through a direct regularized score to value argument.

The three layers are:

1. *Replay window Bayes target and local drift.* The replay regression Bayes target Z_t^A is a convex combination of recent advantages at each state action pair, so it tracks the current advantage A_t up to a local window drift term.
2. *Direct regularized learned readout bound.* Since the learned policy π_{θ_t} is the exact entropy regularized maximizer of the learned score $Z_{\theta_{t-1}}$, we prove regularized approximate optimality for π_{θ_t}. The residual $Z_{\theta_{t-1}} - Z_t^A$ against the regression Bayes target Z_t^A enters through a direct inner product error under the comparator occupancy.
3. *Replay decomposition.* The learned score error $\|Z_{\theta_{t-1}} - Z_t^A\|_{2,\mu_t}$ is decomposed into supervised fitting error, finite replay concentration, and critic label bias.

Estimation chain. The estimation chain summarizes the construction, and each arrow is one estimation step:

$$Z_t \xrightarrow{\text{replay weighting}} \widehat{Z}_t \xrightarrow{\text{regression}} Z_\theta \xrightarrow{\text{mirror map}} \pi_{\theta_{t+1}}.$$

The exact score Z_t is the state-weighted average advantage. Replay sampling replaces it with the Bayes target $\widehat{Z}_t$, which carries the behavior action weights of the stored tuples. Fitting gives Z_θ, and the entropy mirror map turns Z_θ into the next policy. Section 5 bounds each gap in this chain. For states outside the support of $\bar{d}_t$, Z_t can be assigned any fixed value, since those states have zero weight in the objective in (1) until later data cover them.

F.1 PERFORMANCE DIFFERENCE LEMMA

With normalized discounted occupancies, the following lemma converts policy space advantage feedback into value differences.

Lemma F.1 (Performance difference lemma (Kakade & Langford, 2002))**.** *For any stationary comparator $\pi^\star$ and any policy π_{θ_t}, with $A_t = A^{\pi_{\theta_t}}$,*

$$J(\pi^\star) - J(\pi_{\theta_t}) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi^\star}}\left[\langle \pi^\star(\cdot \mid s) - \pi_{\theta_t}(\cdot \mid s), A_t(s,\cdot)\rangle\right]. \tag{6}$$

F.2 WINDOW NOTATION AND REPLAY BAYES TARGET

Fix an exponent $\zeta \in (0,1)$. For every $t \geq 2$, let

$$W_t = \max\{1, \lfloor t^\zeta \rfloor\}$$

be the number of policy iterations represented in the replay window, and define

$$\mathcal{W}_t = \{t - W_t, \ldots, t-1\}.$$

Thus $|\mathcal{W}_t| = W_t$ and $W_t \leq t^\zeta$. Replacing $\lfloor t^\zeta \rfloor$ by $\lceil t^\zeta \rceil$ only changes constants and has no effect on the rate. If the implementation measures replay capacity in samples rather than iterations, a fixed batch size converts sample capacity into a constant multiple of W_t, and that constant is absorbed into C_{var} below.

Throughout this section,

$$J(\pi) = J_\rho(\pi) := V^\pi(\rho) = \mathbb{E}_{s_0\sim\rho}[V^\pi(s_0)]$$

denotes the scalar start distribution value from the fixed initial distribution ρ. We also assume that rewards are uniformly bounded by a global constant $R_{\max} < \infty$:

$$|r(s,a)| \leq R_{\max} \qquad \text{for all } (s,a).$$

We write G_A for a uniform advantage bound. Under this two sided reward bound one may take $G_A = 2R_{\max}/(1-\gamma)$, although the analysis below only uses $\|A_t\|_\infty \leq G_A$.

Write

$$A_k(s,a) = A^{\pi_{\theta_k}}(s,a)$$

for the advantage of the learned policy at round k.

When we need the population replay distribution explicitly, we reuse the occupancy notation from Section 2:

$$\mu_k(s,a) = \mu^{\pi_{\theta_k}}(s,a) = d^{\pi_{\theta_k}}(s)\pi_{\theta_k}(a \mid s)$$

for the state action occupancy of the data generated at round k, and define the window replay marginal

$$\mu_t(s,a) = \frac{1}{W_t} \sum_{k \in \mathcal{W}_t} \mu_k(s,a).$$

Thus μ_t is the population state action distribution represented by the current replay window. Ratios involving $\mu_t(s,a)$ are always understood on the support where $\mu_t(s,a) > 0$.

The score learned by the squared loss replay regression has, with true advantage labels, the population conditional mean target

$$Z_t^A(s,a) = \begin{cases} \dfrac{\sum_{k \in \mathcal{W}_t} \mu_k(s,a) A_k(s,a)}{\sum_{k \in \mathcal{W}_t} \mu_k(s,a)}, & \displaystyle\sum_{k \in \mathcal{W}_t} \mu_k(s,a) > 0, \\ \dfrac{1}{W_t} \displaystyle\sum_{k \in \mathcal{W}_t} A_k(s,a), & \displaystyle\sum_{k \in \mathcal{W}_t} \mu_k(s,a) = 0. \end{cases}$$

On the support of μ_t this is the replay regression Bayes target. The second line is a harmless tie breaker on state action pairs with zero window replay mass. It is included only so that the pointwise supremum $\|A_t - Z_t^A\|_\infty$ used in Layer 1 is defined on every (s,a), and it does not affect the $L_2(\mu_t)$ score error term, which never sees the off support branch. At every (s,a), $Z_t^A(s,a)$ is therefore a convex combination of the recent advantages $\{A_k(s,a) : k \in \mathcal{W}_t\}$, with coefficients proportional to the joint occupancy $\mu_k(s,a)$ on the support and uniform off it. This convex combination property is all that the local drift bound uses.

The algorithm uses a fixed stabilizer $\beta \geq 0$. The temperature of the readout that produces the round-t policy is that of (3),

$$\alpha_{t-1} = \lambda + \frac{\beta}{t-1}, \qquad \lambda > 0.$$

For $t \geq 2$, define the learned readout by

$$\pi_{\theta_t}(\cdot \mid s) = \mathrm{softmax}\left(\frac{Z_{\theta_{t-1}}(s,\cdot)}{\alpha_{t-1}} \right).$$

Here $Z_{\theta_{t-1}}$ denotes the score learned using data available before round t. The proof is stated for the average over $t = 2, \ldots, T$, so every policy in the average is read out from a nonempty past window.

F.3 Layer 1: replay window Bayes target and local drift

The role of Layer 1 is now only to connect the window target appearing in the score residual to the current advantage in the performance difference lemma. The main bridge is the regression Bayes target Z_t^A. Since $Z_t^A(s,a)$ is a convex combination of recent advantages at each fixed (s,a), the local policy path stability assumption implies $\|A_t - Z_t^A\|_\infty = O(t^{\zeta-1})$.

Assumption F.2 (Learned policy path stability)**.** There exists $C_\pi > 0$ such that the learned policies satisfy

$$\sup_s \left\| \pi_{\theta_{k+1}}(\cdot \mid s) - \pi_{\theta_k}(\cdot \mid s) \right\|_1 \leq \frac{C_\pi}{k}$$

for every $k \geq 1$.

This is the algorithm side stability condition needed because the replay buffer is generated by the learned policies π_{θ_k}, not by an auxiliary exact windowed readout. For learned policies, this need not follow from average fitting error alone: consecutive learned policies can move by a constant amount while maintaining small average replay error. We therefore state this assumption explicitly. It is natural for warm started actor updates with $1/k$-scale step sizes, trust region constraints, clipping, or projection onto a path stable policy class.

Lemma F.3 (Value and advantage Lipschitzness in policy)**.** *For any two stationary policies π and π',*

$$\left\|V^{\pi} - V^{\pi'}\right\|_{\infty} \le \frac{R_{\max}}{(1-\gamma)^2} \sup_s \|\pi(\cdot \mid s) - \pi'(\cdot \mid s)\|_1 .$$

Moreover,

$$\left\|A^{\pi} - A^{\pi'}\right\|_{\infty} \le \frac{(1+\gamma)R_{\max}}{(1-\gamma)^2} \sup_s \|\pi(\cdot \mid s) - \pi'(\cdot \mid s)\|_1 .$$

Proof. For any state s,

$$V^{\pi}(s) - V^{\pi'}(s) = \left\langle \pi(\cdot \mid s) - \pi'(\cdot \mid s), Q^{\pi'}(s,\cdot) \right\rangle + \mathbb{E}_{a\sim\pi(\cdot|s)}\left[Q^{\pi}(s,a) - Q^{\pi'}(s,a)\right].$$

By the Bellman relation,

$$Q^{\pi}(s,a) - Q^{\pi'}(s,a) = \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}\left[V^{\pi}(s') - V^{\pi'}(s')\right].$$

Therefore, by Hölder's inequality,

$$\left|V^{\pi}(s) - V^{\pi'}(s)\right| \le \|\pi(\cdot \mid s) - \pi'(\cdot \mid s)\|_1 \left\|Q^{\pi'}\right\|_{\infty} + \gamma \left\|V^{\pi} - V^{\pi'}\right\|_{\infty}.$$

Taking the supremum over s and using $\left\|Q^{\pi'}\right\|_{\infty} \le R_{\max}/(1-\gamma)$ gives

$$\left\|V^{\pi} - V^{\pi'}\right\|_{\infty} \le \frac{R_{\max}}{1-\gamma} \sup_s \|\pi(\cdot \mid s) - \pi'(\cdot \mid s)\|_1 + \gamma \left\|V^{\pi} - V^{\pi'}\right\|_{\infty}.$$

Rearranging proves the value bound. For the advantage bound,

$$\left\|Q^{\pi} - Q^{\pi'}\right\|_{\infty} \le \gamma \left\|V^{\pi} - V^{\pi'}\right\|_{\infty},$$

and since $A^{\pi} = Q^{\pi} - V^{\pi}$,

$$\left\|A^{\pi} - A^{\pi'}\right\|_{\infty} \le \left\|Q^{\pi} - Q^{\pi'}\right\|_{\infty} + \left\|V^{\pi} - V^{\pi'}\right\|_{\infty} \le (1+\gamma)\left\|V^{\pi} - V^{\pi'}\right\|_{\infty}.$$

Substituting the value bound completes the proof. □

Corollary F.4 (Local advantage drift)**.** *Suppose Assumption F.2 holds. If $A_k = A^{\pi_{\theta_k}}$, then for every $k \ge 1$,*

$$\|A_{k+1} - A_k\|_{\infty} \le \frac{L_A}{k}, \qquad L_A = \frac{(1+\gamma)R_{\max}C_{\pi}}{(1-\gamma)^2}.$$

Proof. By Lemma F.3,

$$\|A_{k+1} - A_k\|_{\infty} \le \frac{(1+\gamma)R_{\max}}{(1-\gamma)^2} \sup_s \|\pi_{\theta_{k+1}}(\cdot \mid s) - \pi_{\theta_k}(\cdot \mid s)\|_1.$$

Applying Assumption F.2 to the right hand side gives the claim with $L_A = (1+\gamma)R_{\max}C_{\pi}/(1-\gamma)^2$. □

Lemma F.5 (The recent window tracks the current advantage)**.** *Under Corollary F.4, for every $t \ge 2$,*

$$\left\|A_t - Z_t^A\right\|_{\infty} \le C_{\mathrm{loc}} t^{\zeta-1},$$

where one may take

$$C_{\mathrm{loc}} = \frac{L_A}{1 - 2^{\zeta-1}}.$$

Consequently, for every $T \ge 2$,

$$\frac{2}{T-1}\sum_{t=2}^{T} \left\|A_t - Z_t^A\right\|_{\infty} \le \frac{2C_{\mathrm{loc}}}{\zeta}\frac{T^{\zeta}}{T-1}.$$

Proof. For $k \in \mathcal{W}_t$, Corollary F.4 gives

$$\begin{aligned}\|A_t - A_k\|_\infty &\le \sum_{j=k}^{t-1} \|A_{j+1} - A_j\|_\infty \\ &\le L_A \sum_{j=k}^{t-1} \frac{1}{j}.\end{aligned}$$

Since $k \ge t - W_t$ and $W_t \le t^\zeta$, we have $k \ge t - t^\zeta$. Also, since $t^\zeta/t = t^{\zeta-1} \le 2^{\zeta-1}$ for every $t \ge 2$,

$$t - W_t \ge t - t^\zeta \ge \left(1 - 2^{\zeta-1}\right) t.$$

Therefore,

$$\sum_{j=k}^{t-1} \frac{1}{j} \le \frac{t-k}{t-W_t} \le \frac{W_t}{(1-2^{\zeta-1})t} \le \frac{t^{\zeta-1}}{1-2^{\zeta-1}}.$$

Thus, uniformly over $k \in \mathcal{W}_t$,

$$\|A_t - A_k\|_\infty \le C_{\mathrm{loc}} t^{\zeta-1}.$$

For each fixed (s,a), $Z_t^A(s,a)$ is a convex combination of the values $\{A_k(s,a) : k \in \mathcal{W}_t\}$ (coefficients proportional to $\mu_k(s,a)$ on the support of μ_t, and uniform off it). Hence

$$\begin{aligned}\left|A_t(s,a) - Z_t^A(s,a)\right| &\le \max_{k \in \mathcal{W}_t} |A_t(s,a) - A_k(s,a)| \\ &\le C_{\mathrm{loc}} t^{\zeta-1}.\end{aligned}$$

Taking the supremum over (s,a) proves the pointwise bound. For the average, multiply the pointwise bound by 2 and use

$$\sum_{t=2}^{T} t^{\zeta-1} \le \int_1^T x^{\zeta-1}\, dx = \frac{T^\zeta - 1}{\zeta} \le \frac{T^\zeta}{\zeta}.$$

□

F.4 Layer 2: direct regularized learned readout value transfer

Layer 2 proves regularized approximate optimality directly for the learned policy π_{θ_t}. The only transfer needed is from an $L_2(\mu_t)$ score error bound to a comparator state expectation. This is where the quadratic coverage coefficient enters.

Where the action sampling mismatch is paid. The replay Bayes target weights each action by how often the retained behavior policies took it, so the fitted score is accurate where replay puts probability mass and is unconstrained elsewhere. The analysis does not estimate this mismatch. It bounds it through the assumption that the coverage coefficient below is finite, which holds exactly when the replay occupancy μ_t reaches every action at every state the comparator visits. The burden of the mismatch therefore sits on this coefficient. It grows when the comparator prefers actions the replayed policies rarely take, and the guarantee weakens accordingly. Removing the assumption, for example through importance weighting on the stored labels or through a drift argument, is open.

For a comparator $\pi^\star$, define the quadratic replay to comparator coverage coefficient

$$C_t^{\mathrm{cov}}(\pi^\star) = \operatorname*{ess\,sup}_{s,a} \frac{d^{\pi^\star}(s)}{\mu_t(s,a)},$$

with the convention that the ratio is $+\infty$ whenever $d^{\pi^\star}(s) > 0$ and $\mu_t(s,a) = 0$. Let

$$\mu_t^S(s) = \sum_a \mu_t(s,a)$$

be the replay state marginal. In finite action spaces this coefficient is easy to interpret. If the replay conditional action distribution satisfies

$$\rho_t(a \mid s) = \frac{\mu_t(s,a)}{\mu_t^S(s)} \ge \eta$$

whenever $\mu_t^S(s) > 0$, then

$$C_t^{\mathrm{cov}}(\pi^\star) \le \frac{1}{\eta} \operatorname*{ess\,sup}_s \frac{d^{\pi^\star}(s)}{\mu_t^S(s)}.$$

Thus action exploration still enters the theory through the coverage transfer, even though it no longer enters the replay reweighting lemma.

Lemma F.6 (Direct regularized optimality of the learned readout)**.** *For every comparator policy $\pi^\star$, every state s, and every $t \ge 2$,*

$$\begin{aligned}&\langle \pi^\star(\cdot \mid s) - \pi_{\theta_t}(\cdot \mid s), A_t(s,\cdot)\rangle + \lambda\left(\varphi(\pi_{\theta_t}(\cdot \mid s)) - \varphi(\pi^\star(\cdot \mid s))\right)\\ &\quad\le \frac{\beta}{t-1}\ln|\mathcal{A}| + 2\left\|A_t(s,\cdot) - Z_t^A(s,\cdot)\right\|_\infty + 2\left\|Z_{\theta_{t-1}}(s,\cdot) - Z_t^A(s,\cdot)\right\|_2.\end{aligned}$$

Proof. Fix s and $t \ge 2$. By definition of the readout, with $\alpha_{t-1} = \lambda + \beta/(t-1)$,

$$\pi_{\theta_t}(\cdot \mid s) \in \arg\max_{p \in \Delta(\mathcal{A})} \left\{\left\langle p, Z_{\theta_{t-1}}(s,\cdot)\right\rangle - \alpha_{t-1}\varphi(p)\right\}.$$

Since this objective is concave, evaluating it at $\pi_{\theta_t}(\cdot \mid s)$ and at $\pi^\star(\cdot \mid s)$ and using maximality gives

$$\left\langle \pi^\star(\cdot \mid s) - \pi_{\theta_t}(\cdot \mid s), Z_{\theta_{t-1}}(s,\cdot)\right\rangle + \alpha_{t-1}\left(\varphi(\pi_{\theta_t}(\cdot \mid s)) - \varphi(\pi^\star(\cdot \mid s))\right) \le 0.$$

Writing $\alpha_{t-1} = \lambda + \beta/(t-1)$ and carrying the score from $Z_{\theta_{t-1}}$ to A_t through the regression Bayes target Z_t^A (all quantities evaluated at s),

$$\begin{aligned}&\langle \pi^\star - \pi_{\theta_t}, A_t\rangle + \lambda\left(\varphi(\pi_{\theta_t}) - \varphi(\pi^\star)\right)\\ &= \underbrace{\left\langle \pi^\star - \pi_{\theta_t}, Z_{\theta_{t-1}}\right\rangle + \alpha_{t-1}\left(\varphi(\pi_{\theta_t}) - \varphi(\pi^\star)\right)}_{\le 0} - \frac{\beta}{t-1}\left(\varphi(\pi_{\theta_t}) - \varphi(\pi^\star)\right)\\ &\qquad + \left\langle \pi^\star - \pi_{\theta_t}, A_t - Z_t^A\right\rangle + \left\langle \pi^\star - \pi_{\theta_t}, Z_t^A - Z_{\theta_{t-1}}\right\rangle.\end{aligned}$$

We bound the three remaining terms. Since $\varphi(p) = \sum_a p(a)\ln p(a) \in [-\ln|\mathcal{A}|, 0]$,

$$-\frac{\beta}{t-1}\left(\varphi(\pi_{\theta_t}) - \varphi(\pi^\star)\right) \le \frac{\beta}{t-1}\ln|\mathcal{A}|.$$

By Hölder's inequality with $\|\pi^\star - \pi_{\theta_t}\|_1 \le 2$,

$$\left\langle \pi^\star - \pi_{\theta_t}, A_t - Z_t^A\right\rangle \le 2\left\|A_t(s,\cdot) - Z_t^A(s,\cdot)\right\|_\infty.$$

By the Cauchy–Schwarz inequality with $\|\pi^\star - \pi_{\theta_t}\|_2 \le \|\pi^\star\|_2 + \|\pi_{\theta_t}\|_2 \le 2$ (each distribution has ℓ_2 norm at most 1),

$$\left\langle \pi^\star - \pi_{\theta_t}, Z_t^A - Z_{\theta_{t-1}}\right\rangle \le 2\left\|Z_{\theta_{t-1}}(s,\cdot) - Z_t^A(s,\cdot)\right\|_2.$$

Summing the three bounds proves the claim. □

Proposition F.7 (Direct regularized windowed value transfer)**.** *For every comparator policy $\pi^\star$ and every $T \ge 2$,*

$$\begin{aligned}&\frac{1}{T-1}\sum_{t=2}^{T}\left[\left(J(\pi^\star) - J(\pi_{\theta_t})\right) + \frac{\lambda}{1-\gamma}\mathbb{E}_{s\sim d^{\pi^\star}}\left(\varphi(\pi_{\theta_t}(\cdot \mid s)) - \varphi(\pi^\star(\cdot \mid s))\right)\right]\\ &\quad\le \frac{1}{1-\gamma}\left[\frac{\beta\ln|\mathcal{A}|}{T-1}\sum_{t=2}^{T}\frac{1}{t-1} + \frac{2}{T-1}\sum_{t=2}^{T}\left\|A_t - Z_t^A\right\|_\infty\right.\\ &\qquad\left. + \frac{2}{T-1}\sum_{t=2}^{T}\sqrt{C_t^{\mathrm{cov}}(\pi^\star)}\left\|Z_{\theta_{t-1}} - Z_t^A\right\|_{2,\mu_t}\right].\end{aligned}$$

Consequently, if $C_t^{\mathrm{cov}}(\pi^\star) \le C_{\mathrm{cov}}(\pi^\star)$ for all $t = 2,\dots,T$, the last sum is bounded by $2\sqrt{C_{\mathrm{cov}}(\pi^\star)}\cdot\frac{1}{T-1}\sum_{t=2}^{T}\|Z_{\theta_{t-1}} - Z_t^A\|_{2,\mu_t}$.

Proof. By the performance difference lemma,

$$J(\pi^\star) - J(\pi_{\theta_t}) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi^\star}}\left[\langle \pi^\star(\cdot \mid s) - \pi_{\theta_t}(\cdot \mid s), A_t(s,\cdot)\rangle\right],$$

where $A_t = A^{\pi_{\theta_t}}$. Apply Lemma F.6 pointwise in s and take $\mathbb{E}_{s\sim d^{\pi^\star}}$:

$$\begin{aligned} &J(\pi^\star) - J(\pi_{\theta_t}) + \frac{\lambda}{1-\gamma}\mathbb{E}_{s\sim d^{\pi^\star}}\left(\varphi(\pi_{\theta_t}) - \varphi(\pi^\star)\right) \\ &\quad\le \frac{1}{1-\gamma}\left[\frac{\beta}{t}\ln|\mathcal{A}| + 2\left\|A_t - Z_t^A\right\|_\infty + 2\,\mathbb{E}_{s\sim d^{\pi^\star}}\left\|Z_{\theta_{t-1}}(s,\cdot) - Z_t^A(s,\cdot)\right\|_2\right]. \end{aligned}$$

For the last term, by Jensen's inequality and the change of measure bound (as in the coverage discussion above),

$$\mathbb{E}_{s\sim d^{\pi^\star}}\left\|Z_{\theta_{t-1}}(s,\cdot) - Z_t^A(s,\cdot)\right\|_2 \le \left(\sum_{s,a} d^{\pi^\star}(s)\left(Z_{\theta_{t-1}}(s,a) - Z_t^A(s,a)\right)^2\right)^{1/2} \le \sqrt{C_t^{\mathrm{cov}}(\pi^\star)}\left\|Z_{\theta_{t-1}} - Z_t^A\right\|_{2,\mu_t}.$$

Averaging over $t = 2,\dots,T$ gives the display. The final assertion uses $C_t^{\mathrm{cov}}(\pi^\star) \le C_{\mathrm{cov}}(\pi^\star)$. □

F.5 Layer 3: replay regression error decomposition

Layer 2 reduces the learned policy value bound to the score error

$$\left\|Z_{\theta_{t-1}} - Z_t^A\right\|_{2,\mu_t},$$

the residual of the learned score against the regression Bayes target Z_t^A. The norm is taken under the window replay marginal μ_t defined above.

We recall the population replay target with true advantage labels,

$$Z_t^A(s,a) = \frac{\sum_{k\in\mathcal{W}_t}\mu_k(s,a)A_k(s,a)}{\sum_{k\in\mathcal{W}_t}\mu_k(s,a)}$$

on the support of μ_t. Its critic label analogue is

$$\widehat{Z}_t(s,a) = \frac{\sum_{k\in\mathcal{W}_t}\mu_k(s,a)\widehat{A}_k(s,a)}{\sum_{k\in\mathcal{W}_t}\mu_k(s,a)}.$$

Let

$$\widetilde{Z}_t$$

be the finite buffer empirical target represented by the actual replay sample. In the infinite buffer limit,

$$\widetilde{Z}_t = \widehat{Z}_t.$$

The score error chain is

$$Z_{\theta_{t-1}} \quad\longrightarrow\quad \widetilde{Z}_t \quad\longrightarrow\quad \widehat{Z}_t \quad\longrightarrow\quad Z_t^A.$$

Assumption F.8 (Primitive windowed replay error bounds)**.** For every $T \ge 2$, the following bounds hold.

1. Fitting error:
$$\frac{1}{T-1}\sum_{t=2}^{T}\left\|Z_{\theta_{t-1}} - \widetilde{Z}_t\right\|_{2,\mu_t} \le \bar{\epsilon}_{\mathrm{fit}}.$$
That is, the fitting error is not growing on average.

2. Finite replay concentration:
$$\left\|\widetilde{Z}_t - \widehat{Z}_t\right\|_{2,\mu_t} \le C_{\mathrm{var}}t^{-\zeta/2} \qquad \text{for every } t \ge 2.$$
This rate is the standard $N^{-1/2}$ concentration for the empirical mean of $N_t \asymp t^\zeta$ bounded labels around its conditional expectation, with C_{var} depending on the label range G_A, the regression function class capacity, and a logarithmic factor from a union bound over t.

3. Critic label bias:

$$\frac{1}{T-1}\sum_{t=2}^{T}\left\|\widehat{Z}_t - Z_t^A\right\|_{2,\mu_t} \le \epsilon_0.$$

That is, the critic bias is not growing.

Lemma F.9 (Averaged windowed score error)**.** *Under Assumption F.8, for every $T \ge 2$,*

$$\frac{1}{T-1}\sum_{t=2}^{T}\left\|Z_{\theta_{t-1}} - Z_t^A\right\|_{2,\mu_t} \le \bar{\epsilon}_{\text{fit}} + \epsilon_0 + \frac{C_{\text{var}}}{1-\zeta/2}\frac{T^{1-\zeta/2}}{T-1}.$$

Proof. For each t, apply the triangle inequality along the chain of replay targets ending at Z_t^A:

$$\left\|Z_{\theta_{t-1}} - Z_t^A\right\|_{2,\mu_t} \le \left\|Z_{\theta_{t-1}} - \widetilde{Z}_t\right\|_{2,\mu_t} + \left\|\widetilde{Z}_t - \widehat{Z}_t\right\|_{2,\mu_t} + \left\|\widehat{Z}_t - Z_t^A\right\|_{2,\mu_t}.$$

Average over $t = 2, \dots, T$. The fitting and critic terms are controlled by the averaged bounds in Assumption F.8. For finite replay concentration,

$$\frac{1}{T-1}\sum_{t=2}^{T} C_{\text{var}} t^{-\zeta/2} \le \frac{C_{\text{var}}}{T-1}\sum_{t=2}^{T} t^{-\zeta/2} \le \frac{C_{\text{var}}}{1-\zeta/2}\frac{T^{1-\zeta/2}}{T-1}.$$

Combining these estimates proves the claim. □

F.6 Final windowed value bound

This subsection contains the proofs of Theorem 5.1 and Corollaries 5.2 and F.11 from the main text.

Define the learned policy regularized gap

$$\mathcal{G}_T^{\text{reg}}(\pi^\star) := \frac{1}{T-1}\sum_{t=2}^{T}\left[(J(\pi^\star) - J(\pi_{\theta_t})) + \frac{\lambda}{1-\gamma}\mathbb{E}_{s\sim d^{\pi^\star}}\left(\varphi(\pi_{\theta_t}(\cdot \mid s)) - \varphi(\pi^\star(\cdot \mid s))\right)\right].$$

Proof of Theorem 5.1. Start from Proposition F.7, whose left hand side is $\mathcal{G}_T^{\text{reg}}(\pi^\star)$. Bound the local drift term $\|A_t - Z_t^A\|_\infty$ using Lemma F.5 and the score error term using Lemma F.9. Under the reward bound $|r(s,a)| \le R_{\max}$,

$$L_A = \frac{(1+\gamma)R_{\max}C_\pi}{(1-\gamma)^2}, \qquad C_{\text{loc}} = \frac{(1+\gamma)R_{\max}C_\pi}{(1-\gamma)^2(1-2^{\zeta-1})}.$$

The stabilizer contribution satisfies

$$\frac{\beta \ln|\mathcal{A}|}{T-1}\sum_{t=2}^{T}\frac{1}{t-1} \le \frac{\beta \ln|\mathcal{A}|(1+\ln T)}{T-1}.$$

Substituting these estimates gives the bound in Theorem 5.1. Equivalently, after expanding C_{loc},

$$\mathcal{G}_T^{\text{reg}}(\pi^\star) \le \frac{1}{1-\gamma}\left[\frac{\beta \ln|\mathcal{A}|(1+\ln T)}{T-1} + \frac{2(1+\gamma)R_{\max}C_\pi}{\zeta(1-\gamma)^2(1-2^{\zeta-1})}\frac{T^\zeta}{T-1} + 2\sqrt{C_{\text{cov}}(\pi^\star)}\left(\bar{\epsilon}_{\text{fit}} + \epsilon_0 + \frac{C_{\text{var}}}{1-\zeta/2}\frac{T^{1-\zeta/2}}{T-1}\right)\right].$$

For the balanced choice $\zeta = 2/3$,

$$\frac{2C_{\mathrm{loc}}}{\zeta} = \frac{3(1+\gamma)R_{\max}C_\pi}{(1-\gamma)^2(1-2^{-1/3})}, \qquad \frac{C_{\mathrm{var}}}{1-\zeta/2} = \frac{3}{2}C_{\mathrm{var}},$$

so

$$\mathcal{G}_T^{\mathrm{reg}}(\pi^\star) \le \frac{1}{1-\gamma}\left[\frac{\beta \ln|\mathcal{A}|(1+\ln T)}{T-1} + \frac{3(1+\gamma)R_{\max}C_\pi}{(1-\gamma)^2(1-2^{-1/3})}\frac{T^{2/3}}{T-1} + 2\sqrt{C_{\mathrm{cov}}(\pi^\star)}\left(\bar{\epsilon}_{\mathrm{fit}} + \epsilon_0 + \frac{3}{2}C_{\mathrm{var}}\frac{T^{2/3}}{T-1}\right)\right].$$

Thus, for fixed $\lambda > 0$, bounded coverage, and vanishing fitting and critic bias floors, $\mathcal{G}_T^{\mathrm{reg}}(\pi^\star) = O(T^{-1/3})$. □

Remark F.10 (Reading the bound). The regularized gap is the natural goal for an entropy-regularized method. It is the quantity the RDA iterates optimize, the entropy floor and the stabilizer control it directly, and it converts to the ordinary value gap at the cost of one explicit entropy bias term (Corollary 5.2).

The bound has three groups of terms: the dual averaging stabilizer (optimization) term, the local replay window drift, and the replay regression error, namely supervised fit $\bar{\epsilon}_{\mathrm{fit}}$, critic label bias ϵ_0, and finite buffer variance C_{var}, transferred to the comparator through coverage. The proof is given in Appendix F.

The replay terms expose the window length tradeoff. With $W_t \asymp t^\zeta$, finite buffer variance scales as $T^{-\zeta/2}$ and replay drift as $T^{\zeta-1}$. Balancing the two gives $\zeta = 2/3$ and a $T^{-1/3}$ rate. Smaller ζ uses fresher but noisier data, while larger ζ averages more data but increases stale replay drift.

Proof of Corollary 5.2. Starting from Theorem 5.1, add

$$\frac{\lambda}{1-\gamma}\mathbb{E}_{s\sim d^{\pi^\star}}\left[\varphi(\pi^\star(\cdot \mid s)) - \varphi(\pi_{\theta_t}(\cdot \mid s))\right]$$

to the regularized gap. Since $\varphi(p) \in [-\ln|\mathcal{A}|, 0]$,

$$\varphi(\pi^\star(\cdot \mid s)) - \varphi(\pi_{\theta_t}(\cdot \mid s)) \le \ln|\mathcal{A}|.$$

This gives the exact bound stated in Corollary 5.2. For $\zeta = 2/3$ and fixed $\lambda > 0$, collecting the nonvanishing fitting and critic terms separately from the stabilizer, drift, and variance terms gives

$$\frac{1}{T-1}\sum_{t=2}^{T}(J(\pi^\star) - J(\pi_{\theta_t})) \le \frac{\lambda \ln|\mathcal{A}|}{1-\gamma} + \frac{2\sqrt{C_{\mathrm{cov}}(\pi^\star)}}{1-\gamma}(\bar{\epsilon}_{\mathrm{fit}} + \epsilon_0) + O\Big(T^{-1/3}\Big).$$

The $O(T^{-1/3})$ coefficient is free of any explicit $1/\lambda$ factor. The primitive constants C_π, C_{var}, $C_{\mathrm{cov}}(\pi^\star)$, and constants hidden in $\bar{\epsilon}_{\mathrm{fit}}$ and ϵ_0 may still depend on λ through the learned policy path, replay coverage, and regression difficulty. □

Annealed-λ remark. Because λ now appears *only* in the entropy bias term (the score error coefficient is λ-free), annealing λ no longer inflates the replay terms. This is the improvement over the old softmax stability route, where the G_A/λ factor forced $\lambda_T \asymp T^{-1/6}$ and an unregularized $T^{-1/6}$ rate.

Corollary F.11 (Annealed unregularized rate)**.** *In the setting of Corollary 5.2, suppose the fitting and critic floors vanish, $\bar{\epsilon}_{\mathrm{fit}} = \epsilon_0 = 0$, and $C_\pi, C_{\mathrm{var}}, C_{\mathrm{cov}}(\pi^\star)$ remain uniformly bounded as $\lambda \to 0$. Then $\zeta = 2/3$ and $\lambda_T \asymp T^{-1/3}$ give $\frac{1}{T-1}\sum_{t=2}^{T}(J(\pi^\star) - J(\pi_{\theta_t})) = O(T^{-1/3})$.*

Proof of Corollary F.11. Under the hypotheses of Corollary F.11, set $\bar{\epsilon}_{\mathrm{fit}} = \epsilon_0 = 0$. In Corollary 5.2, the only term with explicit dependence on λ is then the entropy bias

$$\frac{\lambda \ln|\mathcal{A}|}{1-\gamma}.$$

The remaining terms are $O(T^{-1/3})$ and contain no explicit $1/\lambda$ factor. Under the stated uniform boundedness assumptions on C_π, C_{var}, $C_{\mathrm{cov}}(\pi^\star)$, and the constants in the replay error bounds, their coefficients remain bounded along the schedule $\lambda = \lambda_T \to 0$. Taking $\lambda_T \asymp T^{-1/3}$ makes the entropy bias $O(T^{-1/3})$, which proves

$$\frac{1}{T-1}\sum_{t=2}^{T}\left(J(\pi^\star) - J(\pi_{\theta_t})\right) = O\Big(T^{-1/3}\Big).$$

□

Because the score error coefficient $2\sqrt{C_{\mathrm{cov}}(\pi^\star)}$ is free of any explicit $1/\lambda$ factor, annealing the entropy weight leaves the replay terms at $T^{-1/3}$, so the optimization and replay contributions match. The condition $\lambda > 0$ controls the optimization term. The unregularized ablation $\lambda = 0$ (Appendix D) is an empirical stress test. The hypothesis of Corollary F.11 that C_π, C_{var}, and $C_{\mathrm{cov}}(\pi^\star)$ stay bounded as $\lambda \to 0$ is a genuine restriction, since a vanishing entropy floor sharpens the readout and can inflate both the path constant and the coverage coefficient. Corollary 5.2 at fixed λ is therefore the primary statement, and the annealed rate is conditional on that hypothesis.

F.7 Density weighted surrogate and the action sampling mismatch

This subsection connects the direct proof to the main text product space RDA surrogate. The main proof bridges through the replay regression Bayes target Z_t^A, which is exactly the object fit by squared loss. The main text product space surrogate instead induces a state density weighted window score $\bar{Z}_t^W$.

Define the average state occupancy in the current window by

$$\bar{d}_t^W(s) = \frac{1}{W_t}\sum_{k\in\mathcal{W}_t} d^{\pi_{\theta_k}}(s).$$

The density weighted windowed score is

$$\bar{Z}_t^W(s,a) = \begin{cases} \dfrac{\sum_{k\in\mathcal{W}_t} d^{\pi_{\theta_k}}(s) A_k(s,a)}{\sum_{k\in\mathcal{W}_t} d^{\pi_{\theta_k}}(s)}, & \displaystyle\sum_{k\in\mathcal{W}_t} d^{\pi_{\theta_k}}(s) > 0, \\ \dfrac{1}{W_t}\displaystyle\sum_{k\in\mathcal{W}_t} A_k(s,a), & \displaystyle\sum_{k\in\mathcal{W}_t} d^{\pi_{\theta_k}}(s) = 0. \end{cases}$$

The second line is only a tie breaker on states with zero window state mass. It does not change the density weighted surrogate, but it makes $\bar{Z}_t^W$ pointwise defined everywhere.

The exact windowed product space surrogate corresponding to the main text is

$$\arg\max_{\pi\in\Pi}\left\{\frac{1}{W_t}\sum_{k\in\mathcal{W}_t}\langle A_k,\pi\rangle_{d^{\pi_{\theta_k}}} - \alpha_{t-1}\varphi_{\bar{d}_t^W}(\pi)\right\},$$

where $\varphi(p) = \sum_a p(a)\log p(a) = -H(p)$. Expanding the state expectation gives

$$\begin{aligned} &\frac{1}{W_t}\sum_{k\in\mathcal{W}_t}\langle A_k,\pi\rangle_{d^{\pi_{\theta_k}}} - \alpha_{t-1}\varphi_{\bar{d}_t^W}(\pi) \\ &= \frac{1}{W_t}\sum_{k\in\mathcal{W}_t}\sum_s d^{\pi_{\theta_k}}(s)\,\langle\pi(\cdot\mid s), A_k(s,\cdot)\rangle - \alpha_{t-1}\sum_s \bar{d}_t^W(s)\varphi(\pi(\cdot\mid s)) \\ &= \sum_s \bar{d}_t^W(s)\left[\left\langle\pi(\cdot\mid s), \bar{Z}_t^W(s,\cdot)\right\rangle - \alpha_{t-1}\varphi(\pi(\cdot\mid s))\right] \\ &= \left\langle \bar{Z}_t^W, \pi\right\rangle_{\bar{d}_t^W} - \alpha_{t-1}\varphi_{\bar{d}_t^W}(\pi), \end{aligned}$$

where the equality uses the density weighted branch on states with $\bar{d}_t^W(s) > 0$, and states with $\bar{d}_t^W(s) = 0$ carry zero mass in the display. Thus the state density factor multiplies both the linear advantage term and the entropy term, and the statewise optimizer on the support of $\bar{d}_t^W$ is $\mathrm{softmax}(\bar{Z}_t^W(s,\cdot)/\alpha_{t-1})$.

The gap between the replay Bayes target Z_t^A and the density weighted surrogate $\bar{Z}_t^W$ is the action sampling mismatch. The next corollary bounds it and shows that a value guarantee phrased against the density weighted surrogate holds at the same rate.

Corollary F.12 (Action sampling mismatch and the density weighted surrogate)**.** *Under the assumptions of Theorem 5.1, for every $t \geq 2$,*

$$\left\| Z_t^A - \bar{Z}_t^W \right\|_{2,\mu_t} \leq 2C_{\mathrm{loc}} t^{\zeta-1}.$$

Consequently, replacing the replay Bayes target Z_t^A by the density weighted surrogate $\bar{Z}_t^W$ in the score residual adds at most $\frac{2\sqrt{C_{\mathrm{cov}}(\pi^\star)}}{1-\gamma} \cdot \frac{2C_{\mathrm{loc}}}{\zeta} \frac{T^\zeta}{T-1} = O(T^{\zeta-1})$ to the bound of Theorem 5.1, the same order as the local drift term, and does not change the $T^{-1/3}$ rate.

Proof. On the support of μ_t, both $Z_t^A(s,a)$ and $\bar{Z}_t^W(s,a)$ are convex combinations of $\{A_k(s,a) : k \in \mathcal{W}_t\}$, so

$$\begin{aligned}\left\| Z_t^A - \bar{Z}_t^W \right\|_{2,\mu_t} &\leq \sup_{s,a} \left| Z_t^A(s,a) - \bar{Z}_t^W(s,a) \right| \\ &\leq 2 \max_{k \in \mathcal{W}_t} \|A_k - A_t\|_\infty \leq 2C_{\mathrm{loc}} t^{\zeta-1},\end{aligned}$$

where the last inequality is the convex combination stability bound from the proof of Lemma F.5. For the second claim, the triangle inequality gives

$$\left\| Z_{\theta_{t-1}} - \bar{Z}_t^W \right\|_{2,\mu_t} \leq \left\| Z_{\theta_{t-1}} - Z_t^A \right\|_{2,\mu_t} + 2C_{\mathrm{loc}} t^{\zeta-1},$$

and averaging the extra term over $t = 2, \ldots, T$ contributes at most $\frac{2C_{\mathrm{loc}}}{\zeta} \frac{T^\zeta}{T-1}$ to the averaged score residual, which enters the value bound through the prefactor $\frac{2\sqrt{C_{\mathrm{cov}}(\pi^\star)}}{1-\gamma}$ of Theorem 5.1. The addition is of the same order as the local drift term already present in Theorem 5.1. □

The main statements are kept against Z_t^A because this removes the action sampling mismatch from the critical path while retaining the transparent connection to the main text surrogate written here.

F.8 Scope of the guarantee

The theorem provides the design rationale for the estimator and window schedule and analyzes the core algorithmic procedure implemented in code. Its objects are the stored labels, the finite window, and the fitted score. Two assumptions hold as stated in the implementation, with bounded rewards on every benchmark and an exact readout within the chosen score class, using a softmax on Atari and a diagonal Gaussian induced by the quadratic score in MuJoCo. The temperature schedule is the analyzed $\alpha_t = \lambda + \beta/t$. The analyzed window grows as $t^{2/3}$ to balance replay concentration against drift as $t \to \infty$. At a fixed budget the same tradeoff selects a fixed window, and the window ablation (Appendix D.1) confirms that the optimum is interior.

The remaining conditions concern the learned path, and each has a mechanism behind it. Path stability first. In full-history RDA the dual score is a running average of k rounds of feedback, so consecutive scores differ by $O(1/k)$ and the readout inherits that rate. This is where Assumption F.2 originates. A sliding window gives up that mechanism in exchange for faster adaptation, and the implementation supplies stability by other means: the regression is warm-started from the previous round, each round replaces a small fraction of the window (one of 24 rollouts in the GAE MuJoCo setting), the clipped dual regression bounds the movement of every stored prediction by ϵ_{clip} per update, and the temperature decays. We do not verify the C_π/k rate, and the reported runs did not need a step-size schedule, but a schedule gives a direct handle on it. With gradient clipping, the parameter movement per round is bounded by the step size times a constant, so a $1/k$ step size enforces $1/k$ policy movement whenever the readout is Lipschitz in the parameters uniformly over states. Coverage next. The coverage coefficient is finite exactly when replay puts positive mass on every action at every state the comparator visits. The entropy floor supplies the action half: with $\lambda > 0$ every readout policy has full support on a finite action set, so $\mu_t(s,a) > 0$ at every visited state. What remains is state coverage, that replay visits the states the comparator visits, the same exploration requirement that every comparator-based policy improvement guarantee carries. Fitting

and concentration last. The fitting error enters as a per-round average of the error of the score the optimizer returns, which the theorem keeps as a floor. The finite replay concentration rate $C_{\mathrm{var}} t^{-\zeta/2}$ is the variance scaling of an average over $W_t \asymp t^{\zeta}$ samples.

Only two implementation heuristics lie outside the analyzed update: the clipped dual regression of Appendix A.4, and the per-batch normalization and clipping of advantage labels. Both act in the direction of the assumptions. The clip bounds per-update movement, which is what path stability asks for, and the normalization is an affine rescaling of the labels, which changes the effective temperature scale and nothing else in the readout. The twin-Q runs autotune α_t as SAC does, which sits outside the analyzed schedule, and MuJoCo actions are continuous while the proof assumes finite actions. The Gaussian readout is the continuous instance of the same mirror map, and the experiments test whether the finite-action guarantee transfers.

G Related Work

This appendix reviews the work closest to RDA2C. The organizing distinction is between (i) last iterate policy space mirror updates, (ii) cumulative FTRL/RDA updates, and (iii) practical regression or replay mechanisms used to instantiate these ideas with function approximation. RDA2C sits at the intersection of the last two: it uses a dual averaging actor objective, but estimates the cumulative dual object through replayed advantage labels in a deep actor–critic loop.

G.1 Policy mirror descent, natural gradients, and trust region methods

A long line of work interprets modern policy optimization as approximate mirror descent in policy space, with KL divergence as the Bregman divergence induced by negative entropy. Natural policy gradient introduced the use of the Fisher geometry for policy search (Kakade, 2001), and TRPO later operationalized a trust region version of policy improvement with a KL constraint (Schulman et al., 2015). PPO replaces the constrained update by a clipped surrogate that approximately limits deviation from the behavior policy (Schulman et al., 2017), while MPO uses an EM-like KL regularized policy improvement step (Abdolmaleki et al., 2018). Maximum entropy methods such as SAC optimize entropy regularized objectives and can be interpreted within the broader regularized MDP and KL regularized policy optimization family (Haarnoja et al., 2018; Geist et al., 2019).

Theoretical work has made this mirror descent view increasingly explicit. Convex analytic treatments of entropy regularized MDPs relate exact trust region and regularized policy improvement updates to mirror descent and duality (Neu et al., 2017; Geist et al., 2019). Adaptive trust region and policy mirror descent analyses give global convergence and sampling complexity guarantees for exact or near exact policy space updates (Shani et al., 2020; Lan, 2023). Functional mirror ascent and surrogate optimization analyses further clarify how policy improvement objectives can be derived independently of a particular parameterization (Vaswani et al., 2022). Empirically motivated mirror descent actor–critic methods, including mirror descent policy optimization (MDPO), exploratory conservative policy optimization (ECPO), and mirror descent actor–critic (MDAC), instantiate related KL/entropy regularized updates in deep RL systems (Tomar et al., 2022; Mei et al., 2019; Iwaki, 2025).

Q–NPG is relevant to RDA2C because it connects value function regression, compatible function approximation, and natural gradient policy updates. In the policy gradient theory of Agarwal et al. (2019), NPG and Q–NPG updates are written through regression problems whose solutions define a natural policy gradient step. This is close in spirit to our use of advantage labels, but the update remains a last iterate PMD/NPG step: the fitted object determines a local update from the current policy. RDA2C instead fits a replay estimated state weighted average advantage score and reads the policy from that cumulative object through the entropy mirror map.

G.2 FTRL, regularized dual averaging, and policy dual averaging

Regularized dual averaging originates in online convex optimization and stochastic optimization, where the algorithm maintains an average of past subgradients in the dual space and maps back to the primal variable through a regularized mirror map. Foundational treatments include Nesterov's primal-dual subgradient schemes (Nesterov, 2009) and Xiao's regularized dual-averaging method (Xiao, 2010). RDA is closely related to follow-the-regularized-leader: McMahan (2011) establishes equivalence results connecting mirror descent and FTRL updates under suitable choices of regularization and linearization.

This equivalence is important for positioning RDA2C. We do not claim that FTRL/RDA generically improves the worst case online learning rate over mirror descent. Our use is representational: FTRL/RDA exposes the cumulative feedback as a single object. PMD/OMD writes the next update as a local step around the last policy. RDA/FTRL writes the next update as a regularized optimization over the cumulative linearized feedback. In reinforcement learning, that cumulative feedback is naturally represented by accumulated advantage or Q labels. This is what makes replay useful as an actor representation: the buffer approximates the object the RDA update needs.

The closest precedent for the update itself is POLITEX (Abbasi-Yadkori et al., 2019), which sets the next policy to the softmax of the cumulative sum of past action-value estimates and proves regret

bounds through the reduction to expert prediction. RDA2C keeps that cumulative-softmax form but replaces the explicit sum of stored critics by one regression on replayed advantage labels, adds the persistent entropy floor and the decaying stabilizer of RDA, and analyzes a finite replay window in place of the full history. In the same direction, Vieillard et al. (2020a) show that KL-regularized policy iteration implicitly averages past Q estimates, and Deep CPI (Vieillard et al., 2020b) makes the related conservative mixture update practical with function approximation. Both hold the cumulative object recursively, in the current parameters. RDA2C holds it in the label buffer, which is what makes the window an explicit design variable.

Within RL, a close theoretical precursor is policy dual averaging. Ju & Lan (2022) study policy optimization over general state and action spaces and introduce PDA methods designed to work with function approximation more directly than exact policy mirror descent. That line studies cleaner regimes with exact or bounded error evaluation, and the original continuous action update is defined by a statewise optimization subproblem. The actor accelerated PDA follow up (Gao et al., 2026) addresses this computational issue by learning an actor that approximates the PDA target, while DAPO studies dual-side approximation error through the Bregman divergence induced by the mirror potential (Xiong et al., 2024). RDA2C differs by eliminating the separate projection tracking actor: the learned dual score model Z_θ is the policy's natural coordinate representation, and the current policy is obtained directly through the entropy readout $\nabla\varphi^*(Z_\theta/\alpha_t)$.

Two further contrasts follow. PDA solves the readout statewise and numerically, while RDA2C replaces that solve with one regression and an exact mirror map. DAPO measures the error of a given dual approximation, while RDA2C specifies how to construct the dual object from replayed labels. The analyses differ the same way: prior guarantees assume exact or bounded-error policy evaluation, while our finite-time analysis charges the update for critic bias, finite-buffer variance, and replay-induced distribution shift.

G.3 Replay, off policy actor–critic learning, and stale data

Experience replay is a standard mechanism for improving sample efficiency in deep RL. In off-policy actor–critic methods, replay usually provides data for Bellman regression and actor-gradient estimates, as in SAC (Haarnoja et al., 2018). In on-policy or near on-policy methods such as PPO, data are typically reused only for a small number of epochs, with clipping or KL control limiting the effect of policy drift (Schulman et al., 2017). In both cases, stale data are primarily handled as an off-policy correction, stabilization, or variance-reduction issue.

RDA2C changes the role of replay on the actor side. Because the ideal RDA update depends on a historical average of policy space feedback, past advantage labeled samples are not merely stale approximations to the newest gradient. They are samples from the empirical dual objective that defines the next actor. This distinction also explains why buffer size appears explicitly in our theory. A short window reduces distribution shift but raises estimator variance, while a long window averages more labels but can bias the dual estimate toward older policies. The main theorem formalizes this replay-specific decomposition through critic bias, finite buffer variance, and replay induced distribution shift.

G.4 Reduction and regression views of policy improvement

RDA2C is also related to reductions that cast sequential decision making as iterative supervised learning on aggregated data. DAgger aggregates expert-labeled states to reduce imitation learning to no-regret online learning (Ross et al., 2011). NRPI extends this perspective to reinforcement learning by aggregating cost-to-go examples under an exploration distribution and reducing approximate policy iteration to no regret learning (Ross & Bagnell, 2014). These methods are close in spirit to RDA2C because the dataset is not an incidental implementation detail: it is the object through which the next policy improvement problem is specified.

Recent regression based policy optimization methods make a related move in modern RL and RLHF settings. REBEL reduces reinforcement learning to squared loss regression on relative rewards, REFUEL extends a regression and aggregation viewpoint to multi-turn RLHF under self-generated data, and A*-PO estimates an optimal-advantage-like object before applying simple regression policy updates (Gao et al., 2024a;b; Brantley et al., 2025). RDA2C follows the same broad reduction to

regression philosophy, but its regression target is derived from regularized dual averaging: the model Z_θ approximates a windowed average of policy space improvement feedback, and the actor update is the exact entropy mirror map readout from that learned dual score.

Advantage-weighted methods also turn policy improvement into supervised learning. AWR (Peng et al., 2019), AWAC (Nair et al., 2020), and CRR (Wang et al., 2020) fit the policy by weighted maximum likelihood on replayed actions, with weights that increase with the estimated advantage. That regression is in the primal: the target is a reweighted behavior distribution, and the policy is the fitted likelihood model. RDA2C regresses in the dual: the target is the advantage label itself, the fitted object is a score, and the policy is its mirror-map readout. The two differ in what the buffer represents. For advantage weighting, replay supplies actions to imitate. For RDA2C, replay supplies the cumulative feedback that defines the next policy.

Empirical studies of example forgetting in neural networks provide an additional motivation for keeping policy improvement history in an explicit buffer (Toneva et al., 2019). Recursive updates keep that history only in the actor weights. This motivation is secondary. The primary distinction is algorithmic: RDA2C's replay buffer estimates the dual object required by the RDA update, and recursive actor updates keep past feedback only in the current parameter vector.