# Actor-Accelerated Policy Dual Averaging for Reinforcement Learning in Continuous Action Spaces

**Ji Gao**[1]
jgao325@gatech.edu

**Caleb Ju**[2]
calebju4@gatech.edu

**Guanghui Lan**[2]
george.lan@isye.gatech.edu

**Zhaohui Tong**[1]
zt7@gatech.edu

[1]School of Chemical and Biomolecular Engineering
[2]H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology, Atlanta, GA, USA

## Abstract

Policy Dual Averaging (PDA) offers a principled Policy Mirror Descent (PMD) framework that more naturally admits value function approximation than standard PMD, enabling the use of approximate advantage (or Q-) functions while retaining strong convergence guarantees. However, applying PDA in continuous state and action spaces remains computationally challenging, since action selection involves solving an optimization subproblem at each decision step. In this paper, we propose *actor-accelerated PDA* (AA-PDA), which uses a learned policy network to approximate the solution of the optimization subproblems, significantly reducing runtime. We provide a theoretical analysis that quantifies how actor approximation error impacts the convergence of PDA under suitable assumptions. We then evaluate its performance on several benchmarks in robotics, control, and operations research problems. AA-PDA achieves better performance compared to on-policy baselines such as Proximal Policy Optimization (PPO). Overall, our results take a step toward bridging the gap between the theoretical advantages of PDA and its practical deployment in continuous-action problems with function approximation. Our code is available at `https://github.com/jgioa/pdarl`.

## 1 Introduction

Many advances in reinforcement learning (RL) use parameterized policy-gradient methods, such as Trust Region Policy Optimization (TRPO) [Schulman et al., 2015] and Proximal Policy Optimization (PPO) [Schulman et al., 2017]. These methods have achieved great success in continuous control tasks in various disciplines [Duan et al., 2016, Lillicrap et al., 2016]. Their practical performance has been partially supported by recent theoretical analysis. In particular, Policy Mirror Descent (PMD) serves as a unifying framework for first-order RL methods with convergence guarantees [Lan, 2023, Zhan et al., 2023, Tomar et al., 2021, Alfano et al., 2023, Lan et al., 2023, Li and Lan, 2025b, Li et al., 2024, Neu et al., 2017].

Despite this progress, theoretically principled policy updates may require solving optimization problems upon request to obtain actions in continuous action spaces, and neural-network-based parameterization can further complicate the problem. Within PMD-style methods for continuous actions, the theoretically defined update may require solving an optimization problem whenever a policy action is queried. Nonlinear value and policy approximations can make this action-generation

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

subproblem non-convex or poorly conditioned [Fan et al., 2020, Liu et al., 2019, Ju and Lan, 2026b, Alfano et al., 2023, Agarwal et al., 2021]. Large penalty coefficients used to limit policy changes can further increase the objective's Lipschitz constant, making repeated action solves slow or unreliable.

Recently, Policy Dual Averaging (PDA), inspired by Nesterov's Dual Averaging algorithm [Nesterov, 2009], was proposed as a promising alternative that matches PMD's convergence guarantees and offers a new perspective on policy optimization [Ju and Lan, 2026b]. Through careful algorithm design, PDA's optimization subproblems are made weakly convex while avoiding policy function approximation. As a result, locally (possibly globally) optimal solutions for the subproblems can be efficiently solved while evading explicit policy parameterization. While this leads to efficiently solvable policy update steps, the policy evaluation step becomes the main bottleneck. Each action query during policy evaluation requires solving a separate optimization subproblem, and this per-decision cost is often prohibitively expensive when implemented directly [Ju and Lan, 2026b, Lan, 2023]. See Section 3.1 for more details about PDA.

Building upon PDA, we introduce *actor-accelerated PDA* (AA-PDA). AA-PDA changes only action generation by using a learned policy network to approximate the solution of the expensive optimization subproblem, accelerating the action computation during policy evaluation. The main contributions of this paper consist of the following aspects: **1) Practical Framework:** AA-PDA has a simple practical implementation, with only two algorithm-specific hyperparameters for regularization and exploration in addition to the standard hyperparameters for deep RL. It uses one actor forward pass at decision time and standard first-order minibatch training, and preserves a direct link to the PDA objective and convergence analysis; **2) Convergence and Error Analysis:** We conduct theoretical analysis that characterizes how approximation errors from actor-based subproblem solutions impact the overall convergence and optimality; **3) Empirical Performance:** Across continuous-control benchmarks, AA-PDA outperforms on-policy methods such as PPO in aggregate normalized performance at comparable training cost, and approaches the performance of the established off-policy method TD3.

## 2 Related Work

**Policy Mirror Descent.** PMD provides a unifying framework for policy optimization, connecting policy-gradient, trust-region, and proximal methods under a common theoretical perspective. Existing work establishes convergence guarantees under suitable regularity conditions, typically assuming exact or sufficiently accurate policy evaluation [Lan, 2023, Zhan et al., 2023, Alfano et al., 2023]. PMD and PDA are closely related first-order frameworks, but differ in how they aggregate and regularize updates: PMD uses the current advantage estimate with a regularizer centered at the current policy, whereas PDA accumulates advantage information across iterations and regularizes around a fixed initial policy or prox-center.

This distinction also affects implementation in continuous spaces. In PMD, the prox-center changes at every iteration, so with parameterized policies and value functions the algorithm must repeatedly represent and optimize a new objective tied to the current policy. In PDA, the reference policy remains fixed and the cumulative objective can be maintained explicitly. Both updates still require solving statewise optimization subproblems in continuous action spaces, which motivates the action approximation developed in AA-PDA.

**Regularized and Trust-Region Policy Optimization.** Regularization is central to many practical policy optimization algorithms. TRPO [Schulman et al., 2015] formulates policy updates as a constrained or penalized optimization problem to guarantee monotonic improvement. PPO [Schulman et al., 2017] applies clipping to a surrogate objective as an approximation. These methods rely on a linear approximation of the cost function in the PMD framework, where the approximation error is bounded by the Kullback–Leibler (KL) divergence between the old and new policies. To ensure monotonic improvement, the update step can be formulated as minimizing the objective subject to a penalty on the maximum statewise KL divergence. TRPO relaxes the impractical maximum KL constraint to an average KL constraint. PPO introduces a ratio clipping mechanism to bound the likelihood ratio $\pi(a|s)/\pi_k(a|s)$, which helps mitigate the impact of heavy-tailed noise in stochastic advantage estimates. Furthermore, the PMD framework generalizes these formulations by allowing for arbitrary Bregman divergences and strongly convex regularizers.

## 3 Preliminaries

We study RL problems with continuous state and action spaces, modeled as infinite-horizon discounted Markov decision processes (MDPs) defined by a five-tuple $(\mathbb{S}, \mathcal{A}, \mathbb{P}, c, \gamma)$. $\mathbb{S} \subseteq \mathbb{R}^{n_s}$ denotes the state space and $\mathcal{A} \subseteq \mathbb{R}^{n_u}$ denotes the closed and convex action space. The transition dynamics are specified by a stochastic transition model $\mathbb{P} : \mathfrak{B}(\mathbb{S}) \times \mathbb{S} \times \mathcal{A} \to [0, 1]$, which assigns a probability measure over next states for each state-action pair. A deterministic policy $\pi : \mathbb{S} \to \mathcal{A}$ selects a feasible action for a given state. $\gamma \in [0, 1)$ is the discount factor. $c : \mathbb{S} \times \mathcal{A} \to \mathbb{R}$ is the cost function. We keep the cost-minimization convention of the original PDA analysis [Ju and Lan, 2026b]. Reward maximization is the equivalent problem with $c = -r$ for a reward $r$, and our experiments use the environments' rewards, which corresponds to this situation.

For a given policy $\pi$, its performance is quantified by either the action-value function ($Q^\pi(s, a)$) or the state-value function ($V^\pi(s)$):

$$Q^\pi(s, a) := \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a \right], \quad V^\pi(s) := \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) \,\middle|\, s_0 = s \right]. \tag{1}$$

Note that our results can be extended to the regularized case, i.e., the cost $c(s_t, a_t)$ is replaced by a regularized cost $c(s_t, a_t) + h^{a_t}(s_t)$ for a convex regularizer $h^{\cdot}(s) : \mathcal{A} \to \mathbb{R}$ in (1). We only study the un-regularized case since it simplifies the analysis, and our numerical results do not use such regularizers. These definitions imply the identities $V^\pi(s) = Q^\pi(s, \pi(s))$ and

$$Q^\pi(s, a) = c(s, a) + \gamma \int_{\mathbb{S}} V^\pi(s') \, \mathbb{P}(ds' \mid s, a). \tag{2}$$

The objective of the RL problem is to find a policy that minimizes the expected discounted cost under an initial state distribution $\rho$:

$$\min_\pi \; f_\rho(\pi), \quad f_\rho(\pi) := \int_{\mathbb{S}} V^\pi(s) \, \rho(ds). \tag{3}$$

In this paper, we let $\rho$ be an arbitrary distribution over states $\mathbb{S}$ (see Appendix A.2 for details). Here, an optimal policy $\pi^*$ satisfies $V^{\pi^*}(s) \leq V^\pi(s)$ for all $s \in \mathbb{S}$ and all feasible policies $\pi$. To analyze policy improvement, we define the discounted state visitation measure following a policy $\pi$. For an initial state $s$ and any measurable set $B \subseteq \mathbb{S}$, this measure is given by

$$\kappa_s^\pi(B) := (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \Pr^\pi(s_t \in B \mid s_0 = s), \tag{4}$$

where $\Pr^\pi(s_t \in \cdot \mid s_0 = s)$ denotes the state distribution of $s_t$ following the policy $\pi$ with initial state $s_0 = s$.

The performance difference lemma relates the performance of two policies [Kakade and Langford, 2002, Ju and Lan, 2026b]. For any pair of feasible policies $\pi$ and $\pi'$, the difference in their value functions can be written as follows, where $\psi^\pi$ is the advantage function.

$$V^{\pi'}(s) - V^\pi(s) = \frac{1}{1 - \gamma} \int_{\mathbb{S}} \psi^\pi(q, \pi'(q)) \, \kappa_s^{\pi'}(dq), \quad \text{where } \psi^\pi(q, a) := Q^\pi(q, a) - V^\pi(q). \tag{5}$$

### 3.1 Policy Dual Averaging

**Algorithm 1** Policy dual averaging

1: **Input:** Weights $\beta_k \geq 0$, step sizes $\lambda_k \geq 0$, initial policy $\pi_0$.
2: **for** $k = 0, 1, \ldots$ **do**
3: $\quad \pi_{k+1}(s) = \arg\min_{a \in \mathcal{A}} \sum_{t=0}^{k} \beta_t \psi^{\pi_t}(s, a) + \lambda_k D(\pi_0(s), a)$
4: **end for**

We first briefly introduce the original PDA algorithm (Algorithm 1). Each PDA iteration defines an updated policy $\pi_{k+1}(s)$ as the minimizer of an optimization subproblem involving a weighted sum

of past advantage functions plus a regularization term $\lambda_k D(\pi_0(s), a)$. The theory for dual averaging will set the scalar weights $\beta_t$ to be an increasing sequence so that more recent advantage functions are more influential. Here, $D(\cdot, \cdot)$ is a Bregman divergence (or prox-function) generated by a (without loss of generality by scaling) 1-strongly convex distance generating function $\omega : \mathcal{A} \to \mathbb{R}$ with respect to a norm $\|\cdot\|$:

$$D(a_2, a_1) := \omega(a_1) - [\omega(a_2) + \langle \omega'(a_2), a_1 - a_2 \rangle] \geq \|a_1 - a_2\|^2/2, \quad \forall a_1, a_2 \in \mathcal{A} \tag{6}$$

The choice of $\pi_0(s)$ (the prox-center or anchor point) instead of recent $\pi_k(s)$ will be beneficial later when incorporating function approximation like neural networks.

Similar to PMD, PDA has theoretical guarantees. When the advantage function can be exactly computed, then PDA converges to the globally optimal policy at a linear rate. In the stochastic setting and under standard statistical assumptions, PDA can still converge to global optimality with sample complexity of $O(\epsilon^{-2})$. We also mention that when using strongly convex regularizations $h(s)$, such as entropy regularization, PDA has an improved $O(\epsilon^{-1})$ sample complexity. See Ju and Lan [2026b] for more details.

## 4 Actor-Accelerated Policy Dual Averaging

PDA and AA-PDA use the same cumulative regularized PDA objective and weighted advantage history. In the original PDA formulation, the policy is represented implicitly rather than by an explicit parametric actor, so there is no need to solve the optimization problem over the entire state space in advance. Instead, the action is obtained by solving the corresponding optimization subproblem only when the policy is queried at a particular state. AA-PDA changes only this action-generation mechanism: it trains an actor to approximate the same statewise minimizer, replacing query-time optimization with a single actor forward pass during rollout. This also enables AA-PDA to use standard policy evaluation procedures, the same ones as in PPO.

Algorithm 2 distinguishes the PDA action from its actor approximation. For each queried state $s$, the exact action $\pi_{k+1}(s) = \arg\min_{a \in \mathcal{A}} \tilde{\Psi}_k(s, a)$ minimizes the cumulative regularized action objective. Throughout the paper, we refer to this statewise problem as the *optimization subproblem*. Rather than solving the subproblem whenever the policy is queried, AA-PDA trains an actor $\hat{\pi}_{k+1}(s; \theta_k^\pi)$ to approximate its minimizer, where $\theta_k^\pi$ denotes the actor parameters obtained by fitting the iteration-$k$ objective $\tilde{\Psi}_k$. For the analysis, let $\tilde{\psi}$ denote the advantage function parameterized by $\theta_k$, and define the state-averaged Bregman divergence originating at state $s$ as $\mathcal{D}_s(\pi, \pi') := \mathbb{E}_{q \sim \kappa_s^{\pi^*}}[D(\pi(q), \pi'(q))]$. The step sizes $\beta_k$ and $\lambda_k$ in Algorithm 2 are specified in Theorems 1 and 2.

**Algorithm 2** Actor-accelerated PDA with function approximation

**Input:** Weights $\beta_k \geq 0$, step sizes $\lambda_k \geq 0$, initial policy $\pi_0$, with $\hat{\pi}_0 := \pi_0$.
**for** $k = 0, 1, \ldots$ **do**
  Define an approximate advantage function:
    $\tilde{\psi}(s, a; \theta_k) \approx \psi^{\hat{\pi}_k}(s, a) := Q^{\hat{\pi}_k}(s, a) - V^{\hat{\pi}_k}(s)$
  Define the cumulative objective as:
    $\tilde{\Psi}_k(s, a) := \sum_{t=0}^{k} \beta_t \tilde{\psi}(s, a; \theta_t) + \lambda_k D(\pi_0(s), a)$
  Action-computation *optimization subproblem:*
  $\hat{\pi}_{k+1}(s; \theta_k^\pi) \approx \pi_{k+1}(s) = \arg\min_{a \in \mathcal{A}} \tilde{\Psi}_k(s, a)$
**end for**

## 5 Convergence and Error Analysis

We analyze the convergence of actor-accelerated PDA in the presence of both function approximation error in the advantage function $\tilde{\psi}$ and optimality gap in actor $\hat{\pi}$ for subproblem solution. We start by placing assumptions on $\tilde{\psi}$. Let $\xi_k$ denote the random samples generated at iteration $k$ to fit/learn

$\tilde{\psi}(s, a; \theta_k)$. Define the approximation error

$$\delta_k(s, \cdot) := \tilde{\psi}(s, \cdot; \theta_k) - \psi^{\hat{\pi}_k}(s, \cdot) = \underbrace{[\tilde{\psi}(s, \cdot; \theta_k) - \mathbb{E}_{\xi_k}\tilde{\psi}(s, \cdot; \theta_k)]}_{\delta_k^{sto}(s,\cdot)} + \underbrace{[\mathbb{E}_{\xi_k}\tilde{\psi}(s, \cdot; \theta_k) - \psi^{\hat{\pi}_k}(s, \cdot)]}_{\delta_k^{det}(s,\cdot)},$$

The decomposition is additive. The stochastic error $\delta_k^{sto}$ captures statistical estimation error from the samples used at iteration $k$, while the deterministic error $\delta_k^{det}$ contains bias and function-approximation error. Here $\mathbb{E}_{\xi_k}$ is the expectation w.r.t. $\xi_k$ conditioned on the filtration $F_{k-1} := \{\xi_0, \ldots, \xi_{k-1}\}$.

**Assumption 1** (Regularity)**.** We make the following assumptions regarding the advantage function and the objective:

1. **Lipschitz continuity**: The approximate advantage function $\tilde{\psi}(s, \cdot; \theta)$ is $M_{\tilde{Q}}$-Lipschitz, meaning $\|\tilde{\psi}(s, a; \theta) - \tilde{\psi}(s, a'; \theta)\| \leq M_{\tilde{Q}}\|a - a'\|$ for all $a, a'$. The true advantage function $\psi^{\pi}(s, \cdot)$ is similarly $M_Q$-Lipschitz. Consequently, the approximation error $\delta_k(s, \cdot)$ is $(M_{\tilde{Q}} + M_Q)$-Lipschitz.
2. **Weak convexity**: The approximate advantage function $\tilde{\psi}(s, \cdot; \theta)$ is $\mu_{\tilde{Q}}$-weakly convex, where $\mu_{\tilde{Q}} \in \mathbb{R}$, meaning that the function $\tilde{\psi}(s, a; \theta) + \frac{\mu_{\tilde{Q}}}{2}\|a\|^2$ is convex with respect to $a$. We define $\tilde{\mu}_d := -\mu_{\tilde{Q}}$.
3. **Approximate evaluation error bound**: For all iterations $k$, the expected approximation errors are bounded by a constant $\varsigma > 0$ such that $\max_{s \in \mathbb{S}} \mathbb{E}\Big[\big|\delta_{k-1}^{det}(s, \hat{\pi}_{k-1}(s))\big| + \big|\delta_{k-1}^{det}(s, \pi^*(s))\big|\Big] \leq \varsigma$.

An immediate consequence of the "weak convexity" assumption is that the cumulative objective $\tilde{\Psi}_k(s, a)$ from Algorithm 2 is strongly convex with respect to $a$, with modulus $\tilde{\mu}_k := \tilde{\mu}_d \sum_{t=0}^{k} \beta_t + \lambda_k$ as long as $\lambda_k$ ensures $\tilde{\mu}_k \geq 0$.

We next state an assumption on the approximate updated policy $\hat{\pi}_{k+1}$ from Algorithm 2. This assumption can be viewed as the policy update analogue of the "Approximate evaluation error bound" assumption above. This assumption is needed to guarantee the stronger distribution-free convergence [Ju and Lan, 2026a], where the optimality gap is made small across all states simultaneously. In practice, there may be only a finite set of states of interest, such as the start state in MuJoCo environments. It may then be possible to relax this assumption to hold on a subset of states related to them. We leave the analysis of such relaxed conditions to future work.

**Assumption 2** (Optimality Gap)**.** The actor $\hat{\pi}_{k+1}$ approximately solves the optimization subproblem with a function value error bounded by $\epsilon_{opt,k}(s)$, which satisfies:

$$\tilde{\Psi}_k(s, \hat{\pi}_{k+1}(s)) \leq \tilde{\Psi}_k(s, \pi_{k+1}(s)) + \epsilon_{opt,k}(s), \quad \forall s \in \mathbb{S},\ k. \tag{7}$$

Moreover, there exists a constant $\epsilon$ such that $0 \leq \epsilon_{opt,k}(s) \leq \epsilon$ for all iterations $k$ and states $s \in \mathbb{S}$.

Using the generalized strong convexity with Bregman divergence [Lan, 2020] of $\tilde{\Psi}_k$, this implies a bound on the distance between the ideal and approximate policies:

$$D(\pi_{k+1}(s), \hat{\pi}_{k+1}(s)) \leq \frac{\epsilon_{opt,k}(s)}{\tilde{\mu}_k}, \quad \forall s \in \mathbb{S},\ k. \tag{8}$$

**Discussion on Assumptions.** Lipschitz continuity holds for smooth neural network architectures, such as fully connected neural networks with smooth activation functions like Tanh [Allen-Zhu et al., 2019]. Weak convexity also holds for such smooth neural networks because their Hessians are bounded [Beck, 2017, Davis and Drusvyatskiy, 2019]. The "approximate evaluation error bound" presumes that the function approximator, such as a neural network, can learn the true advantage function up to an error threshold $\varsigma$. The threshold $\varsigma$ is finite when both the costs $c(s, a)$ and neural network outputs are bounded. Furthermore, since neural networks are universal function approximators, $\varsigma$ can possibly be made arbitrarily small when $\psi^{\pi}(\cdot, \cdot)$ is jointly continuous in both inputs (see Ju and Lan, 2026b for sufficient conditions) and the width or depth of the neural network is sufficiently large. In addition, Appendix A.4 has further explanations on $\varsigma$ and $\epsilon$.

### 5.1 Convergence when $\tilde{\mu}_d \geq 0$

We start with the case of $\tilde{\mu}_d \geq 0$, which means the cumulative advantage term $\sum_{t=0}^{k} \beta_t \tilde{\psi}(s, a; \theta_t)$ is convex in $a$. In this case, Theorem 1 controls the global performance gap and global convergence requires the residual approximation terms to vanish.

**Theorem 1.** *Suppose $\lambda_{k+1} \geq \lambda_k \geq 0$ for all $k \geq 0$ and $\lambda_0 > 0$, with the convention $\lambda_{-1} := \lambda_0$ and $\epsilon_{opt,-1} := 0$. Then under Assumptions 1 and 2, if $\tilde{\mu}_d \geq 0$, the performance gap of the sequence of policies generated by actor-accelerated PDA satisfies:*

$$\begin{aligned}(1-\gamma)\frac{1}{\bar{\beta}_k}\sum_{t=0}^{k-1}\beta_t \mathbb{E}[V^{\hat{\pi}_t}(s) - V^{\pi^*}(s)] + \frac{\tilde{\mu}_{k-1}}{\bar{\beta}_k}\mathbb{E}[\mathcal{D}_s(\pi_k, \pi^*)] \\ \leq \frac{\lambda_{k-1}}{\bar{\beta}_k}\mathcal{D}_s(\pi_0, \pi^*) + \frac{M_{\tilde{Q}}^2}{2\bar{\beta}_k}\sum_{t=0}^{k-1}\frac{\beta_t^2}{\tilde{\mu}_{t-1}} + \varsigma + \frac{\bar{\epsilon}_{opt}(k)}{\bar{\beta}_k},\end{aligned} \tag{9}$$

*for any $s \in \mathbb{S}$, where $\bar{\beta}_k = \sum_{t=0}^{k-1} \beta_t$, and the cumulative optimization error term is defined as:*

$$\bar{\epsilon}_{opt}(k) = \mathbb{E}_{s\sim\kappa_q^{\pi^*}}\Bigg[\underbrace{\mathbb{E}[\epsilon_{opt,k-1}(s)] + \sum_{t=0}^{k-1}\mathbb{E}[\epsilon_{opt,t}(s)]}_{\textit{Optimality Gap}} + \underbrace{M_{\tilde{Q}}\sum_{t=0}^{k-1}\mathbb{E}\left[\sqrt{\frac{2\beta_t^2\epsilon_{opt,t-1}(s)}{\tilde{\mu}_{t-1}}}\right]}_{\textit{Distance to Exact Policy}}\Bigg] \tag{10}$$

*When $\tilde{\mu}_d > 0$, if we choose $\beta_k = k+1$, $\lambda_k = \tilde{\mu}_d$, and bound $\epsilon_{opt}$ with $\epsilon$, then*

$$\begin{aligned}\frac{2(1-\gamma)}{k(k+1)}\sum_{t=0}^{k-1}\{(t+1)\mathbb{E}[V^{\hat{\pi}_t}(q) - V^{\pi^*}(q)]\} + \tilde{\mu}_d\mathbb{E}[\mathcal{D}_q(\pi_k, \pi^*)] \\ \leq \frac{2\tilde{\mu}_d\mathcal{D}_q(\pi_0, \pi^*)}{k(k+1)} + \frac{4M_{\tilde{Q}}^2}{\tilde{\mu}_d(k+1)} + \varsigma + \frac{2\epsilon}{k} + \frac{4M_{\tilde{Q}}\sqrt{2\epsilon}}{\sqrt{\tilde{\mu}_d}(k+1)}.\end{aligned} \tag{11}$$

*When $\tilde{\mu}_d = 0$, if we choose $\beta_k = k+1$, $\lambda_k = \lambda(k+1)^{3/2}$, and bound $\epsilon_{opt}$ with $\epsilon$, then*

$$\begin{aligned}&\frac{2(1-\gamma)}{k(k+1)}\sum_{t=0}^{k-1}\{(t+1)\mathbb{E}[V^{\hat{\pi}_t}(q) - V^{\pi^*}(q)]\} \\ &\leq \frac{2\lambda\mathcal{D}_q(\pi_0, \pi^*)\sqrt{k}}{k+1} + \frac{4M_{\tilde{Q}}^2\sqrt{k+1}}{\lambda k} + \varsigma + \frac{2\epsilon}{k} + \frac{4M_{\tilde{Q}}\sqrt{2\epsilon}}{\sqrt{\lambda}}\frac{(k+1)^{1/4}}{k}.\end{aligned} \tag{12}$$

A complete proof of Theorem 1 is provided in Appendix B.2. Our main theoretical result says the $\beta_k$-weighted average optimality gap converges to $\mathcal{O}(\varsigma)$ as $k \to \infty$. Thus a randomized iterate drawn according to the normalized $\beta_k$ weights reaches the stated residual neighborhood. The theorem does not imply that this iterate is exactly globally optimal when $\varsigma > 0$ or actor error remains. Exact global convergence follows in the error-free case under the theorem's remaining conditions. Last-iterate guarantees would require an additional argument and are not claimed here. We mention that $\tilde{\mu}_d \geq 0$ holds for finite actions with exact or linear function approximation of the advantage function, giving global convergence when $\varsigma = 0$ and the actor errors vanish. In the finite state–action setting with exact parameterization and exact action solves, the bound for $\tilde{\mu}_d > 0$ is nearly tight up to a $(1-\gamma)^{-1}$ factor (Appendix A.3).

### 5.2 Convergence when $\tilde{\mu}_d < 0$

We next examine the case of $\tilde{\mu}_d < 0$, which means the cumulative advantage term $\sum_{t=0}^{k} \beta_t \tilde{\psi}(s, a; \theta_t)$ is non-convex in $a$ with bounded lower curvature. In this case, we will establish a different type of convergence based on the negative advantage function.

**Theorem 2.** *Assume that the number of iterations $k$ is fixed a priori. Under Assumptions 1 and 2, if $\tilde{\mu}_d < 0$ and step sizes are chosen such that $\lambda_t = \lambda = k(k+1)|\tilde{\mu}_d|$ and $\beta_t = t+1$, $t = 0, 1, \ldots, k-1$,*

*then for any $s \in \mathbb{S}$, there exists an iteration index $\bar{k}(s)$ such that*

$$-\frac{(M_Q + M_{\tilde{Q}})^2}{|\tilde{\mu}_d|(k+1)} - \frac{4\epsilon}{\bar{k}(s)+1} \leq -\psi^{\hat{\pi}_{\bar{k}(s)}}(s, \hat{\pi}_{\bar{k}(s)+1}(s)) \leq \frac{2\left[V^{\pi_0}(s) - V^{\pi^*}(s)\right]}{k+1} + \frac{3(M_Q + M_{\tilde{Q}})^2}{(1-\gamma)|\tilde{\mu}_d|(k+1)} + \frac{8\epsilon(H_k+1)}{(1-\gamma)(k+1)}, \tag{13}$$

*where $H_k = \sum_{j=1}^{k} \frac{1}{j}$ is the k-th harmonic number, and $\tilde{\mu}_t = \frac{(t+1)(t+2)}{2}\tilde{\mu}_d + k(k+1)|\tilde{\mu}_d|$, for $t = 0, \ldots, k-1$.*

A complete proof of Theorem 2 is provided in Appendix B.3. As $k \to \infty$, the error term involving the harmonic number $H_k$ vanishes because the $k$ in the denominator dominates the growth of the numerator. Under the cost convention, improvement corresponds to a negative advantage, and $-\psi^{\hat{\pi}_{\bar{k}(s)}}(s, \hat{\pi}_{\bar{k}(s)+1}(s))$ measures how much the next policy improves upon the current one. Theorem 2 shows that, under our assumptions, for each state $s$ there exists an iteration $\bar{k}(s)$ at which this advantage improvement decreases at an $O(1/k + \log k/k)$ rate. Thus, the algorithm reaches a stage where an update gives little further advantage-based improvement. In the finite state and action setting, the advantage function is proportional to the gradient [Lan, 2023]. In this well-structured setting, Theorem 2 can be viewed as converging to a local solution via gradient-norm minimization. For more general problems, how to characterize policies with a small negative advantage function remains an open question.

## 6 AA-PDA: Implementation

**Algorithm 3** Actor-Accelerated PDA Training Loop

**Input:** Number of iterations $K$; Networks $V_{\theta_V}$, $\psi^{\Sigma}_{\theta_\psi}$, $\pi_{\theta_\pi}$; Parameters $\lambda, \sigma_0$
**Initialize:** $\Sigma_\beta = 0, \pi_0$ (prior policy or zero)
**for** $k = 0$ to $K-1$ **do**
  Update coefficients: $\beta \leftarrow k+1$, $\Sigma_\beta \leftarrow \Sigma_\beta + \beta$, and $\sigma \leftarrow \sigma_0/(k+1)^{0.3}$.
  Collect $\tau = \{(s_i, a_i, c_i, s_i')\}_{i=1}^{I}$ with noise $\sigma$, and form batches $\boldsymbol{s} = \{s_i\}_{i=1}^{I}, \boldsymbol{a} = \{a_i\}_{i=1}^{I}$.
  Compute discounted cost-to-go $G$ and normalized advantages $\tilde{A}$.
  Update value function: $\theta_V \leftarrow \text{argmin}_{\theta_V'} \mathbb{E}\left[\|V_{\theta_V'}(\boldsymbol{s}) - G\|_2^2\right]$.
  Fix old sum advantage values: $\psi^{\Sigma}_{\text{old}} \leftarrow \psi^{\Sigma}_{\theta_\psi}(\boldsymbol{s}, \boldsymbol{a})$.
  Update sum advantage network: $\theta_\psi \leftarrow \text{argmin}_{\theta_\psi'} \mathbb{E}\left[\|\psi^{\Sigma}_{\theta_\psi'}(\boldsymbol{s}, \boldsymbol{a}) - (\frac{\Sigma_\beta - \beta}{\Sigma_\beta}\psi^{\Sigma}_{\text{old}} + \frac{\beta}{\Sigma_\beta}\tilde{A})\|_2^2\right]$.
  Update actor: $\theta_\pi \leftarrow \text{argmin}_{\theta_\pi'} \mathbb{E}\left[\psi^{\Sigma}_{\theta_\psi}(\boldsymbol{s}, \pi_{\theta_\pi'}(\boldsymbol{s})) + \frac{\beta^{3/2}\lambda}{\Sigma_\beta}\|\pi_{\theta_\pi'}(\boldsymbol{s}) - \pi_0(\boldsymbol{s})\|_2^2\right]$.
**end for**

For on-policy algorithms such as PPO, Gaussian policies are typically used with learnable parameters to control the variance of the sampling noise. The degree of exploration is subsequently controlled by entropy coefficients. AA-PDA instead adds Gaussian exploration noise around the deterministic actor with standard deviation $\sigma(k) = \sigma_0/(k+1)^{0.3}$, avoiding the undefined $k = 0$ denominator. We set $\beta_k = k+1$ and $\lambda_k = \lambda(k+1)^{3/2}$ by the theorem when $\tilde{\mu}_d = 0$. Although the theorem derives this schedule under its stated assumptions, we use it here primarily for its empirical performance rather than under an assumption of global convexity of the learned objective. The base values $\lambda = 0.5$ and $\sigma_0 = 1.3$ were tuned on Hopper-v4 and Walker2d-v4 and then fixed across the benchmark. Sections 7.5 and 8 report the sensitivity evidence and the theory–practice gap.

AA-PDA maintains one more network, sum advantage, in addition to the value network and actor network commonly used for on-policy algorithms. This fixed-size network replaces explicit storage and repeated evaluation of a growing collection of past advantage functions. The actor is then trained against the cumulative sum advantage objective plus the fixed-prox regularizer. PPO uses a GAE estimate from the current rollout and does not explicitly include it in subsequent surrogate objectives. In contrast, the PDA policy update depends on a weighted accumulation of advantage functions across

iterations. The sum advantage network approximates this accumulation recursively, using its previous prediction and the current GAE estimate as the regression target. The pseudo-code of the actor-accelerated PDA implementation is shown in Algorithm 3. Further discussion on implementation is shown in Appendices C and D.

# 7 Experiments

## 7.1 Continuous Control Benchmark

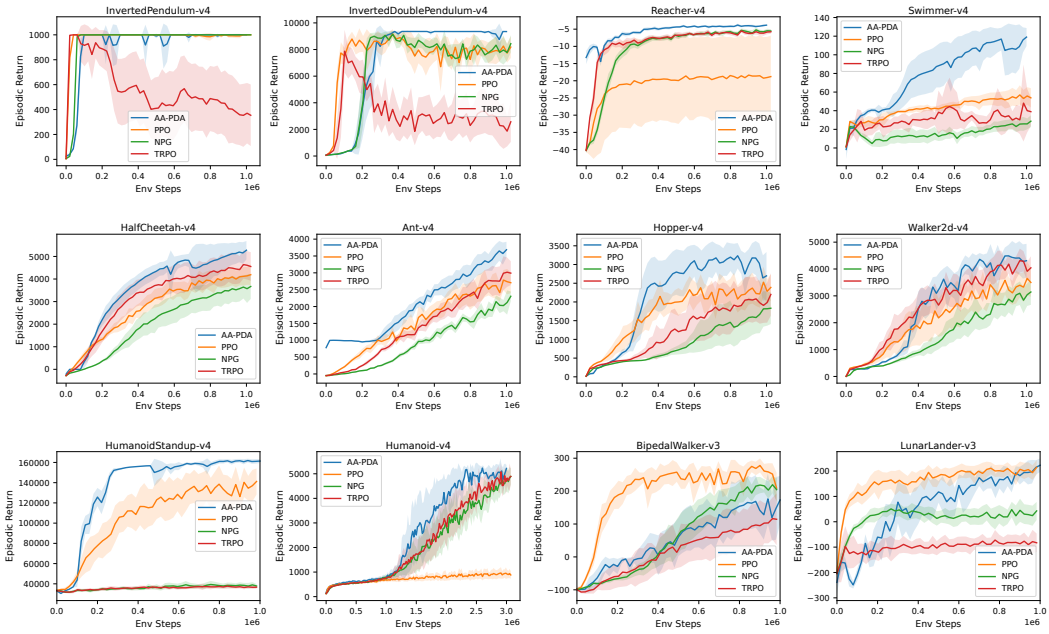



Figure 1: Performance comparison of AA-PDA, PPO, TRPO, and NPG across MuJoCo-v4 and Box2D environments. Curves show the mean across ten seeds, where each seed is evaluated over ten test episodes. Shaded regions indicate $\pm 2\,\mathrm{SEM}$ (standard error of the mean) across seed-level means.

We implemented AA-PDA within the Tianshou [Weng et al., 2022] framework, an open-source RL library built on PyTorch [Paszke et al., 2019] and Gymnasium [Towers et al., 2025]. This enables a robust implementation and fair comparisons with other RL algorithms using its well-tuned hyperparameters. Tianshou's PPO implementation achieves better performance in episodic return on MuJoCo [Todorov et al., 2012] benchmarks compared to the original implementation. For this reason, we adopt Tianshou's PPO as the primary baseline for evaluating our AA-PDA algorithm.

We evaluate AA-PDA on classic continuous control environments, including MuJoCo and Box2D, and compare its performance against widely used on-policy algorithms such as PPO, TRPO, and NPG [Kakade, 2001]. AA-PDA hyperparameters (Table 3) are tuned on Hopper and Walker2d and then *kept fixed for all environments*. Hyperparameters for PPO, TRPO, and NPG are from Tianshou and also *kept constant throughout*. Note that Tianshou's official benchmarks report the maximum testing episodic return over 1 million environment steps, while we use the average testing episodic return of the final five epochs, a more conservative and commonly used metric.

AA-PDA outperforms PPO and the other on-policy baselines in most tasks, as shown in Figure 1 and Table 5. In particular, AA-PDA achieves strong performance in locomotion tasks including HalfCheetah, Ant, Walker2d, Hopper, and the Humanoid variants. For the challenging Humanoid variants, AA-PDA has a higher final-window mean than PPO within the stated 1M or 3M horizon using the default parameters.

### 7.2 Comparison with Direct PDA

To isolate the effect of the actor, we compare AA-PDA with two direct implementations of PDA that solve the action subproblem whenever an action is queried: PDA-RSGF uses the randomized stochastic gradient-free (RSGF) method [Ghadimi and Lan, 2013], and PDA-ACFGM uses the auto-conditioned fast gradient method (ACFGM) [Li and Lan, 2025a]. All three methods share the same policy-evaluation settings. On Ant, HalfCheetah, Hopper, and Walker2d with ten seeds, AA-PDA reaches an aggregate return comparable to the direct solvers while training about 4 and 8 times faster than PDA-RSGF and PDA-ACFGM, respectively (Table 1). Learning curves and runtime breakdowns are in Appendix D.3.

Table 1: AA-PDA versus direct PDA action solvers on four MuJoCo-v4 tasks (ten seeds, 1M environment steps). Return is normalized by the PDA-RSGF final-return reference and averaged over tasks. Runtime is averaged over tasks within seed. Entries are mean $\pm 2\,\mathrm{SEM}$ across seeds.

| Method | Normalized return | Runtime per task (s) | Runtime ratio |
|---|---|---|---|
| AA-PDA | $1.176 \pm 0.117$ | $626.4 \pm 12.2$ | $1.00\times$ |
| PDA-RSGF | $1.026 \pm 0.088$ | $2515.9 \pm 47.3$ | $4.02\times$ |
| PDA-ACFGM | $1.253 \pm 0.092$ | $5004.0 \pm 31.4$ | $7.99\times$ |

### 7.3 Operations Research Benchmark

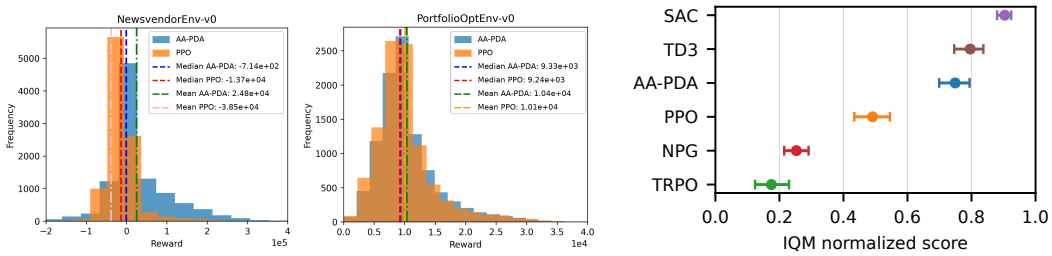



Figure 2: Left: OR-Gym return distributions for AA-PDA and PPO, aggregated over $10^3$ evaluations from each of ten seeds. Vertical lines mark the mean and median. Newsvendor uses a lower display threshold for readability. Training curves are provided in Appendix E.3. Right: aggregate interquartile mean (IQM) of min–max normalized final returns over 12 MuJoCo-v4 and Box2D environments with ten seeds each. Error bars are 95% stratified-bootstrap intervals (Appendix E.7).

Beyond continuous control tasks, RL shows strong potential for addressing decision-making problems in operations research (OR). To assess AA-PDA's performance, we evaluate AA-PDA across four OR environments from the OR-Gym benchmark suite [Hubbs et al., 2020, Balaji et al., 2019, Glasserman and Tayur, 1995, Dantzig and Infanger, 1993]. Our evaluation includes comparisons with on-policy algorithms such as PPO, as well as traditional OR methods. We first consider two classic tasks in stochastic optimization: Newsvendor, a multi-period newsvendor problem with lead times, and PortfolioOpt, a multi-period asset allocation problem (Figure 2, left), where we examine the return distribution of the trained agents. In the Newsvendor environment, AA-PDA has higher pooled mean and median values than PPO, where a mean larger than the median indicates that the displayed return distribution is positively skewed. In the PortfolioOpt environment, AA-PDA achieves slightly better overall performance, with its return distribution shifted positively relative to PPO. For InvManagement environments (multi-echelon supply chain inventory management problems with backlog or lost sales), we compare the RL policies against classical OR methods, including Shrinking Horizon Linear Programming (SHLP), Derivative-Free Optimization (DFO), and Mixed-Integer Programming (MIP). These methods are further compared to an Oracle model that computes a theoretical upper bound using knowledge of the actual demand. As shown in Table 2, the AA-PDA policy achieves comparable performance to PPO. In addition, AA-PDA achieves returns close to SHLP in both Backlog and Lost Sales settings, with much lower standard deviations than the aforementioned classical OR methods.

Table 2: Performance comparison of AA-PDA, PPO, and classical OR methods on the InvManagement environments with Backlog and Lost Sales settings. We report the mean episodic return, standard deviation (Std), and performance ratio relative to the Oracle benchmark (lower is better). Results for the OR methods and the Oracle are taken from the original study [Hubbs et al., 2020].

| | **Backlog** | | | | | | **Lost Sales** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | AA-PDA | PPO | SHLP | DFO | MIP | Oracle | AA-PDA | PPO | SHLP | DFO | MIP | Oracle |
| Mean | 491.6 | 496.4 | 508.0 | 360.9 | 388.0 | 546.8 | 472.3 | 447.4 | 485.4 | 364.3 | 378.5 | 542.7 |
| Std | 11.8 | 6.8 | 28.1 | 39.9 | 30.8 | 30.3 | 10.3 | 16.8 | 29.1 | 33.8 | 26.1 | 29.9 |
| Ratio | 1.1 | 1.1 | 1.1 | 1.5 | 1.4 | 1.0 | 1.1 | 1.2 | 1.1 | 1.5 | 1.4 | 1.0 |

### 7.4 Comparison with Off-Policy Methods

While a direct comparison between on-policy methods and off-policy methods such as Soft Actor-Critic (SAC) [Haarnoja et al., 2018] and Twin Delayed Deep Deterministic Policy Gradient (TD3) [Fujimoto et al., 2018] is not strictly fair due to differences in data utilization, we nonetheless benchmark AA-PDA against these methods to assess its performance. Per-task and aggregate results are in Appendices E.4 and E.7. In the aggregate comparison (Figure 2, right), SAC has the strongest IQM, $0.904$ [0.880, 0.924]. AA-PDA has IQM $0.749$ [0.699, 0.794] and TD3 has $0.796$ [0.746, 0.837], with overlapping stratified-bootstrap intervals. TD3 has much higher final-window means on environments such as HalfCheetah-v4, Humanoid-v4, and BipedalWalker-v3, whereas AA-PDA is stronger on some other tasks. On our hardware, AA-PDA trains roughly 13–48 times faster in wall-clock time than SAC and 20–27 times faster than TD3 (Table 4 in Appendix E.1).

### 7.5 Sensitivity Study for $\sigma_0$ and $\lambda$

AA-PDA has two algorithm-specific hyperparameters, the step size $\lambda$ and the initial exploration noise $\sigma_0$, fixed at $\lambda = 0.5$ and $\sigma_0 = 1.3$ throughout the benchmark. A sweep over both on Hopper, Walker2d, Ant, HalfCheetah, and BipedalWalker (Appendix C.4) shows a broad region of competitive settings rather than a single tuned configuration. The preferred region is task dependent: the balancing tasks (Hopper, Walker2d, BipedalWalker) favor higher exploration noise, whereas Ant and HalfCheetah favor lower noise and larger step sizes. Constant noise can work well on individual tasks, but no single constant level suits all of them, which motivates the decreasing noise schedule.

## 8 Limitations

First, AA-PDA trains three neural networks, a value function, a sum advantage function, and an actor, compared with two for PPO. This overhead is not significant in our benchmarks with small networks ($64 \times 64$), but may make a difference with larger networks. Second, the convergence analysis relies on statewise bounds on the actor and evaluation errors, which are needed for the distribution-free guarantees. These errors cannot be computed exactly. The sampled diagnostics in Appendix D.4 provide empirical support for the assumptions rather than a verification. In addition, the schedules for $\beta_k$ and $\lambda_k$ follow the $\tilde{\mu}_d = 0$ case of the analysis, although the implementation does not assume $\tilde{\mu}_d = 0$. Third, AA-PDA is on-policy and does not reuse replay-buffer data. An off-policy extension would change the interpretation of the sum advantage function and the cumulative objective, and would require extending the convergence analysis. Finally, our experiments use standard multilayer perceptrons (MLPs) and established baselines. Combining AA-PDA with recent architectural and off-policy advances is left for future work.

## 9 Conclusion

Actor-accelerated PDA makes PDA practical in continuous action spaces by amortizing the per-query action subproblem with a policy network, and our analysis shows how actor and advantage-evaluation errors enter its convergence guarantees. Empirically, it trains about 4–8 times faster than direct PDA solvers with comparable return and outperforms PPO, TRPO, and NPG on most evaluated tasks.

## Acknowledgements and Disclosures

This work was supported by NSF-Cyber-Physical Systems (CPS)-NIFA (2024-67021-43862) and the NSF AI Institute for Advances in Optimization (2112533). The authors declare no competing interests.

## References

A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22(98): 1–76, 2021.

C. Alfano, R. Yuan, and P. Rebeschini. A Novel Framework for Policy Mirror Descent with General Parameterization and Linear Convergence. *Advances in Neural Information Processing Systems*, 36:30681–30725, Dec. 2023.

Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via over-parameterization. In *International conference on machine learning*, pages 242–252. PMLR, 2019.

M. G. Azar, R. Munos, and H. J. Kappen. Minimax PAC bounds on the sample complexity of reinforcement learning with a generative model. *Machine Learning*, 91(3):325–349, 2013. doi: 10.1007/s10994-013-5368-1. URL `https://doi.org/10.1007/s10994-013-5368-1`.

B. Balaji, J. Bell-Masterson, E. Bilgin, A. Damianou, P. M. Garcia, A. Jain, R. Luo, A. Maggiar, B. Narayanaswamy, and C. Ye. ORL: Reinforcement Learning Benchmarks for Online Stochastic Optimization Problems. In *NeurIPS 2019 Workshop on Deep Reinforcement Learning*, Dec. 2019.

A. Beck. *First-order methods in optimization*. SIAM, 2017.

G. B. Dantzig and G. Infanger. Multi-stage stochastic linear programs for portfolio optimization. *Annals of Operations Research*, 45(1):59–76, Dec. 1993. ISSN 1572-9338. doi: 10.1007/BF02282041.

D. Davis and D. Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019.

Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking Deep Reinforcement Learning for Continuous Control. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1329–1338. PMLR, June 2016.

J. Fan, Z. Wang, Y. Xie, and Z. Yang. A theoretical analysis of deep q-learning. In *Learning for dynamics and control*, pages 486–489. PMLR, 2020.

S. Fujimoto, H. van Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR, 10–15 Jul 2018.

S. Fujimoto, W.-D. Chang, E. Smith, S. S. Gu, D. Precup, and D. Meger. For SALE: State-action representation learning for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/hash/c20ac0df6c213db6d3a930fe9c7296c8-Abstract-Conference.html`.

S. Fujimoto, P. D'Oro, A. Zhang, Y. Tian, and M. Rabbat. Towards general-purpose model-free reinforcement learning. In *International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=R1hIXdST22`.

S. Ghadimi and G. Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM journal on optimization*, 23(4):2341–2368, 2013.

P. Glasserman and S. Tayur. Sensitivity Analysis for Base-Stock Levels in Multiechelon Production-Inventory Systems. *Management Science*, Feb. 1995. doi: 10.1287/mnsc.41.2.263.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870. PMLR, 10–15 Jul 2018.

P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep reinforcement learning that matters. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18, pages 3207–3214, New Orleans, Louisiana, USA, Feb. 2018. AAAI Press. ISBN 978-1-57735-800-8.

C. D. Hubbs, H. D. Perez, O. Sarwar, N. V. Sahinidis, I. E. Grossmann, and J. M. Wassick. OR-Gym: A Reinforcement Learning Library for Operations Research Problems, Oct. 2020. URL `https://arxiv.org/abs/2008.06319`.

C. Ju and G. Lan. Auto-exploration for online reinforcement learning. *arXiv preprint arXiv:2512.06244*, 2025.

C. Ju and G. Lan. Strongly-polynomial time and validation analysis of policy gradient methods. *Mathematical Programming*, pages 1–45, 2026a.

C. Ju and G. Lan. Policy Optimization over General State and Action Spaces. *Accepted for publication in SIAM Journal on Optimization*, 2026b.

S. M. Kakade. A Natural Policy Gradient. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001.

S. M. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In C. Sammut and A. Hoffmann, editors, *Proceedings of the 19th International Conference on Machine Learning*, pages 267–274, Sydney, Australia, 8–12 Jul 2002. Morgan Kaufmann.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

G. Lan. Deterministic Convex Optimization. In G. Lan, editor, *First-Order and Stochastic Optimization Methods for Machine Learning*, pages 53–111. Springer International Publishing, Cham, 2020. ISBN 978-3-030-39568-1. doi: 10.1007/978-3-030-39568-1_3.

G. Lan. Policy mirror descent for reinforcement learning: Linear convergence, new sampling complexity, and generalized problem classes. *Mathematical Programming*, 198(1):1059–1106, Mar. 2023. ISSN 1436-4646. doi: 10.1007/s10107-022-01816-5.

G. Lan, Y. Li, and T. Zhao. Block Policy Mirror Descent. *SIAM Journal on Optimization*, 33(3): 2341–2378, Sept. 2023. ISSN 1052-6234. doi: 10.1137/22M1480409.

H. Lee, D. Hwang, D. Kim, H. Kim, J. J. Tai, K. Subramanian, P. R. Wurman, J. Choo, P. Stone, and T. Seno. Simba: Simplicity bias for scaling up parameters in deep reinforcement learning. In *International Conference on Learning Representations*, 2025a. URL `https://openreview.net/forum?id=jXLiDKsuDo`.

H. Lee, Y. Lee, T. Seno, D. Kim, P. Stone, and J. Choo. Hyperspherical normalization for scalable deep reinforcement learning. In *International Conference on Machine Learning*, 2025b. URL `https://openreview.net/forum?id=kfYxyvCYQ4`.

T. Li and G. Lan. A simple uniformly optimal method without line search for convex optimization. *Mathematical Programming*, 2025a. doi: 10.1007/s10107-025-02250-z. URL `https://doi.org/10.1007/s10107-025-02250-z`.

Y. Li and G. Lan. Policy Mirror Descent Inherently Explores Action Space. *SIAM Journal on Optimization*, 35(1):116–156, Mar. 2025b. ISSN 1052-6234. doi: 10.1137/23M1560215.

Y. Li, G. Lan, and T. Zhao. Homotopic policy mirror descent: Policy convergence, algorithmic regularization, and improved sample complexity. *Mathematical Programming*, 207(1):457–513, Sept. 2024. ISSN 1436-4646. doi: 10.1007/s10107-023-02017-4.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In Y. Bengio and Y. LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

B. Liu, Q. Cai, Z. Yang, and Z. Wang. Neural trust region/proximal policy optimization attains globally optimal policy. *Advances in neural information processing systems*, 32, 2019.

Y. Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1):221–259, Aug. 2009. ISSN 1436-4646. doi: 10.1007/s10107-007-0149-x.

G. Neu, A. Jonsson, and V. Gómez. A unified view of entropy-regularized Markov decision processes, May 2017. URL `https://arxiv.org/abs/1705.07798`.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1889–1897. PMLR, June 2015.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms, Aug. 2017. URL `https://arxiv.org/abs/1707.06347`.

E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, Oct. 2012. doi: 10.1109/IROS.2012.6386109.

M. Tomar, L. Shani, Y. Efroni, and M. Ghavamzadeh. Mirror Descent Policy Optimization. In *International Conference on Learning Representations*, Oct. 2021.

M. Towers, A. Kwiatkowski, J. U. Balis, G. D. Cola, T. Deleu, M. Goulão, K. Andreas, M. Krimmel, A. Kg, R. D. L. Perez-Vicente, J. K. Terry, A. Pierré, S. V. Schulhoff, J. J. Tai, H. Tan, and O. G. Younis. Gymnasium: A Standard Interface for Reinforcement Learning Environments. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, Oct. 2025.

N. Vyas, D. Morwani, R. Zhao, I. Shapira, D. Brandfonbrener, L. Janson, and S. M. Kakade. SOAP: Improving and Stabilizing Shampoo using Adam for Language Modeling. In *The Thirteenth International Conference on Learning Representations*, Oct. 2024.

J. Weng, H. Chen, D. Yan, K. You, A. Duburcq, M. Zhang, Y. Su, H. Su, and J. Zhu. Tianshou: A Highly Modularized Deep Reinforcement Learning Library. *Journal of Machine Learning Research*, 23(267):1–6, 2022. ISSN 1533-7928.

W. Zhan, S. Cen, B. Huang, Y. Chen, J. D. Lee, and Y. Chi. Policy Mirror Descent for Regularized Reinforcement Learning: A Generalized Framework with Linear Convergence. *SIAM Journal on Optimization*, 33(2):1061–1091, June 2023. ISSN 1052-6234.

# A Further Discussion on Assumptions

## A.1 Stochastic Costs

The observed one-step cost may be stochastic. If the environment returns a random cost $\hat{c}(s,a)$, we take $c(s,a) := \mathbb{E}[\hat{c}(s,a) \mid s,a]$ and only require $\hat{c}(s,a)$ to be an unbiased estimate of $c(s,a)$ for every fixed state–action pair, so that no additional bias enters policy evaluation.

## A.2 Explanation on $\rho$

By deriving convergence results for any distribution $\rho$, our results are *distribution-free*, which leads to some important consequences [Ju and Lan, 2026a]. First, it ensures nonlinear programming methods for solving (3) achieve a similar convergence rate as dynamic programming (e.g., value iteration) and linear programming methods. Second, it leads to faster performance because the policy optimization does not depend on the unknown stationary distribution $\nu^*$ of the optimal policy $\pi^*$ as seen in [Lan, 2023, Li et al., 2024].

## A.3 Tightness of the Convergence Guarantees

For general problems with arbitrary bias $\varsigma > 0$ and actor-approximation error $\epsilon > 0$, there do not exist tractable lower sample-complexity bounds, since the subproblems involve solving non-linear least-squares problems, which are NP-hard. In the simpler finite-state and finite-action setting, exact parameterization and exact action solves are possible, so $\varsigma = 0$ and $\epsilon = 0$. In this case, the bound for $\tilde{\mu}_d > 0$ in Theorem 1 is nearly tight up to additional dependence on $(1-\gamma)^{-1}$, using the minimax rate in Azar et al. [2013]. We conjecture that the bound for $\tilde{\mu}_d = 0$ in Theorem 1 is also nearly tight, but no lower bounds have been established for this setting yet.

## A.4 Interpreting and Monitoring the Error Bounds

The constant $\varsigma$ in Assumption 1 bounds the sum of two deterministic error sources, the function-approximation error and the bias of the learned advantage function, evaluated at the action of the executed actor policy $\hat{\pi}_{k-1}$ and at the optimal-policy action. These terms generally cannot be computed exactly, as they depend on population expectations and on the unknown $\pi^*$. Finding a globally optimal neural-network approximation is itself a non-convex and possibly intractable problem. No separate bound is imposed on the stochastic error $\delta_k^{sto}$. Its effect enters the analysis through the Lipschitz constant $M_{\tilde{Q}}$.

Appendix D.4 decomposes the sampled advantage error into its stochastic and deterministic parts and estimates a proxy for $\varsigma$ with $\pi^*$ replaced by a best-found action (Figure 12). Figure 6 compares the actor output with best-found solutions of the action subproblem, which measures the optimality gap in Assumption 2. In these diagnostics, the actor's normalized gap to the best-found solutions rises early in training and then levels off. The 95th-percentile curve of the $\varsigma$ proxy remains within a finite observed range over 4M steps.

# B Proof of Theorems

## B.1 Lemmas

**Lemma 1.** *For any feasible policy $\hat{\pi}_{k+1}(s)$ and $\pi_{k+1}(s)$:*

$$\tilde{\Psi}_k(s, \hat{\pi}_{k+1}(s)) - \epsilon_{opt,k}(s) + \tilde{\mu}_k D(\pi_{k+1}(s), a) \le \tilde{\Psi}_k(s,a)$$

*Proof.* From the strong convexity of $\tilde{\Psi}_k$ and the generalized optimality condition of $\pi_{k+1}$ with Bregman divergence, we have: $\tilde{\Psi}_k(s, \pi_{k+1}(s)) + \tilde{\mu}_k D(\pi_{k+1}(s), a) \le \tilde{\Psi}_k(s,a)$. Substituting $\tilde{\Psi}_k(s, \pi_{k+1}(s))$ with Assumption 2 gives the result. □

### B.2 Proof of Theorem 1

*Proof.* We first write the recursive form of $\tilde{\Psi}_k$ (which follows directly from its definition) for $k \geq 0$:

$$\tilde{\Psi}_k(s,a) = \tilde{\Psi}_{k-1}(s,a) + \beta_k \tilde{\psi}(s,a;\theta_k) + (\lambda_k - \lambda_{k-1}) D(\pi_0(s), a),$$

where, for the index $-1$, we use the convention $\lambda_{-1} := \lambda_0 > 0$, $\tilde{\Psi}_{-1}(s,a) := \lambda_0 D(\pi_0(s), a)$, $\hat{\pi}_0 := \pi_0$, and $\epsilon_{opt,-1}(s) := 0$ for all $s, a$. Setting $a = \hat{\pi}_{k+1}(s)$ above and taking note that $\lambda_k - \lambda_{k-1} \geq 0$ and the Bregman divergence is nonnegative, we arrive at:

$$\beta_k \tilde{\psi}(s, \hat{\pi}_{k+1}(s); \theta_k) \leq \tilde{\Psi}_k(s, \hat{\pi}_{k+1}(s)) - \tilde{\Psi}_{k-1}(s, \hat{\pi}_{k+1}(s)).$$

Using Lemma 1 with index $k-1$ evaluated $a = \hat{\pi}_{k+1}(s)$, we have:

$$-\tilde{\Psi}_{k-1}(s, \hat{\pi}_{k+1}(s)) \leq -\tilde{\Psi}_{k-1}(s, \hat{\pi}_k(s)) + \epsilon_{opt,k-1}(s) - \tilde{\mu}_{k-1} D(\pi_k(s), \hat{\pi}_{k+1}(s)).$$

Combine the two inequalities above, we have:

$$\beta_k \tilde{\psi}(s, \hat{\pi}_{k+1}(s); \theta_k) \leq \tilde{\Psi}_k(s, \hat{\pi}_{k+1}(s)) - \tilde{\Psi}_{k-1}(s, \hat{\pi}_k(s)) + \epsilon_{opt,k-1}(s) - \tilde{\mu}_{k-1} D(\pi_k(s), \hat{\pi}_{k+1}(s))$$

Applying this inequality recursively yields a telescopic sum. Since $\tilde{\Psi}_{-1}(s, \hat{\pi}_0(s)) = \lambda_0 D(\pi_0(s), \pi_0(s)) = 0$, we have:

$$\sum_{t=0}^{k-1} \beta_t \tilde{\psi}(s, \hat{\pi}_{t+1}(s); \theta_t) \leq \tilde{\Psi}_{k-1}(s, \hat{\pi}_k(s)) + \sum_{t=0}^{k-1} \epsilon_{opt,t}(s) - \sum_{t=0}^{k-1} \tilde{\mu}_{t-1} D(\pi_t(s), \hat{\pi}_{t+1}(s)).$$

We now apply Lemma 1 with index $k-1$ and an arbitrary action $a$ to the inequality derived above to upper bound the $\tilde{\Psi}_{k-1}(s, \hat{\pi}_k(s))$ term:

$$\begin{aligned}\sum_{t=0}^{k-1} \beta_t \tilde{\psi}(s, \hat{\pi}_{t+1}(s); \theta_t) \leq & \tilde{\Psi}_{k-1}(s,a) + \epsilon_{opt,k-1}(s) - \tilde{\mu}_{k-1} D(\pi_k(s), a) \\ & + \sum_{t=0}^{k-1} \epsilon_{opt,t}(s) - \sum_{t=0}^{k-1} \tilde{\mu}_{t-1} D(\pi_t(s), \hat{\pi}_{t+1}(s)).\end{aligned}$$

Expanding $\tilde{\Psi}_{k-1}(s,a)$ with definition and substituting, we have:

$$\begin{aligned}\sum_{t=0}^{k-1} \beta_t [\tilde{\psi}(s, \hat{\pi}_{t+1}(s); \theta_t) - \tilde{\psi}(s,a;\theta_t)] \leq & \lambda_{k-1} D(\pi_0(s), a) - \tilde{\mu}_{k-1} D(\pi_k(s), a) \\ & + \sum_{t=0}^{k-1} \epsilon_{opt,t}(s) + \epsilon_{opt,k-1}(s) - \sum_{t=0}^{k-1} \tilde{\mu}_{t-1} D(\pi_t(s), \hat{\pi}_{t+1}(s)).\end{aligned}$$

For the first term on the lefthand side, use Lipschitz continuity and function approximation error from Assumption 1:

$$\tilde{\psi}(s, \hat{\pi}_{t+1}(s); \theta_t) \geq -M_{\tilde{Q}} \|\hat{\pi}_{t+1}(s) - \hat{\pi}_t(s)\| + \delta_t(s, \hat{\pi}_t(s)).$$

For the $\tilde{\psi}(s,a;\theta_t)$ term on the lefthand side, apply function approximation error again and let $a = \pi^*(s)$:

$$-\tilde{\psi}(s, \pi^*(s); \theta_t) = -\psi^{\hat{\pi}_t}(s, \pi^*(s)) - \delta_t(s, \pi^*(s)).$$

Combining these, we have:

$$\begin{aligned}\sum_{t=0}^{k-1} \beta_t [-\psi^{\hat{\pi}_t}(s, \pi^*(s))] \leq & \lambda_{k-1} D(\pi_0(s), \pi^*(s)) - \tilde{\mu}_{k-1} D(\pi_k(s), \pi^*(s)) \\ & + \sum_{t=0}^{k-1} \epsilon_{opt,t}(s) + \epsilon_{opt,k-1}(s) - \sum_{t=0}^{k-1} \tilde{\mu}_{t-1} D(\pi_t(s), \hat{\pi}_{t+1}(s)) \\ & + \sum_{t=0}^{k-1} \beta_t [\delta_t(s, \pi^*(s)) - \delta_t(s, \hat{\pi}_t(s))] + \sum_{t=0}^{k-1} \beta_t M_{\tilde{Q}} \|\hat{\pi}_{t+1}(s) - \hat{\pi}_t(s)\|.\end{aligned}$$

To simplify the last three summations above, we use the triangle inequality $\|\hat{\pi}_{t+1}(s) - \hat{\pi}_t(s)\| \leq \|\hat{\pi}_{t+1}(s) - \pi_t(s)\| + \|\pi_t(s) - \hat{\pi}_t(s)\|$ in the following derivation:

$$\begin{aligned}
&\sum_{t=0}^{k-1} \beta_t M_{\tilde{Q}} \|\hat{\pi}_{t+1}(s) - \hat{\pi}_t(s)\| - \sum_{t=0}^{k-1} \tilde{\mu}_{t-1} D(\pi_t(s), \hat{\pi}_{t+1}(s)) \\
&\quad + \sum_{t=0}^{k-1} \beta_t [\delta_t(s, \pi^*(s)) - \delta_t(s, \hat{\pi}_t(s))] \\
&\leq \sum_{t=0}^{k-1} \beta_t M_{\tilde{Q}} \|\hat{\pi}_{t+1}(s) - \pi_t(s)\| - \sum_{t=0}^{k-1} \tilde{\mu}_{t-1} D(\pi_t(s), \hat{\pi}_{t+1}(s)) \\
&\quad + \sum_{t=0}^{k-1} \beta_t M_{\tilde{Q}} \|\pi_t(s) - \hat{\pi}_t(s)\| + \varsigma \sum_{t=0}^{k-1} \beta_t \\
&\leq \sum_{t=0}^{k-1} \frac{M_{\tilde{Q}}^2 \beta_t^2}{2\tilde{\mu}_{t-1}} + \sum_{t=0}^{k-1} \beta_t M_{\tilde{Q}} \sqrt{\frac{2\epsilon_{opt,t-1}(s)}{\tilde{\mu}_{t-1}}} + \varsigma \sum_{t=0}^{k-1} \beta_t,
\end{aligned}$$

where the first inequality uses Assumption 1. This step holds in expectation, which is taken below. Note that $\hat{\pi}_t$ is $F_{t-1}$-measurable, so $\mathbb{E}[\delta_t^{sto}(s, \hat{\pi}_t(s)) \mid F_{t-1}] = 0$, and likewise at the deterministic action $\pi^*(s)$. Hence $\mathbb{E} \sum_{t=0}^{k-1} \beta_t [\delta_t(s, \pi^*(s)) - \delta_t(s, \hat{\pi}_t(s))] \leq \varsigma \sum_{t=0}^{k-1} \beta_t$ by Assumption 1. Meanwhile, the first summation in the second inequality uses strong convexity of the Bregman divergence $D(x, y) \geq \frac{1}{2}\|x - y\|^2$ and the inequality $bx - \frac{a}{2}x^2 \leq \frac{b^2}{2a}$, and the second summation follows from Assumption 2.

Now fix any $q \in \mathbb{S}$. Combining the last two intermediate results, taking expectations first w.r.t. the noise $\xi$ and then the state $s \sim \kappa_q^{\pi^*}$, applying (5) with $\pi' = \pi^*$, and dividing through by $\bar{\beta}_k := \sum_{t=0}^{k-1} \beta_t$, we have

$$\begin{aligned}
(1-\gamma) \frac{1}{\bar{\beta}_k} \sum_{t=0}^{k-1} \beta_t \mathbb{E}[V^{\hat{\pi}_t}(q) - V^{\pi^*}(q)] + \frac{\tilde{\mu}_{k-1}}{\bar{\beta}_k} \mathbb{E}[\mathcal{D}_q(\pi_k, \pi^*)] \\
\leq \frac{\lambda_{k-1}}{\bar{\beta}_k} \mathcal{D}_q(\pi_0, \pi^*) + \frac{M_{\tilde{Q}}^2}{2\bar{\beta}_k} \sum_{t=0}^{k-1} \frac{\beta_t^2}{\tilde{\mu}_{t-1}} + \varsigma + \frac{\bar{\epsilon}_{opt}(k)}{\bar{\beta}_k},
\end{aligned}$$

where the cumulative optimization error term is defined as:

$$\bar{\epsilon}_{opt}(k) = \mathbb{E}_{s \sim \kappa_q^{\pi^*}} \Bigg[ \underbrace{\mathbb{E}[\epsilon_{opt,k-1}(s)] + \sum_{t=0}^{k-1} \mathbb{E}[\epsilon_{opt,t}(s)]}_{\text{Optimality Gap}} + \underbrace{M_{\tilde{Q}} \sum_{t=0}^{k-1} \mathbb{E}\left[ \sqrt{\frac{2\beta_t^2 \epsilon_{opt,t-1}(s)}{\tilde{\mu}_{t-1}}} \right]}_{\text{Distance to Exact Policy}} \Bigg]$$

When $\tilde{\mu}_d > 0$, if we choose $\beta_k = k + 1$, $\lambda_k = \tilde{\mu}_d$, and use the approximation error bound $0 \leq \epsilon_{opt,k}(s) \leq \epsilon$, then $\tilde{\mu}_{t-1} = \tilde{\mu}_d \left(1 + \frac{t(t+1)}{2}\right)$, and the bound follows after simplifying terms w.r.t. $k$.

When $\tilde{\mu}_d = 0$, if we choose $\beta_k = k + 1$, $\lambda_k = \lambda(k+1)^{3/2}$, and use the approximation error bound $0 \leq \epsilon_{opt,k}(s) \leq \epsilon$, and the integral bound ($\sum_{x=1}^{k} \sqrt{x} \leq \int_0^{k+1} \sqrt{x} dx$), then we get the result after simplifying terms w.r.t. $k$. □

### B.3 Proof of Theorem 2

*Proof.* Similar to the proof for Theorem 1, we apply Lemma 1 for $\tilde{\Psi}_{k-1}(s, \hat{\pi}_{k+1}(s))$:

$$\beta_k \tilde{\psi}(s, \hat{\pi}_{k+1}(s); \theta_k) \leq \tilde{\Psi}_k(s, \hat{\pi}_{k+1}(s)) - \tilde{\Psi}_{k-1}(s, \hat{\pi}_k(s)) + \epsilon_{\text{opt},k-1}(s) - \tilde{\mu}_{k-1} D(\pi_k(s), \hat{\pi}_{k+1}(s)).$$

Apply Lemma 1 again for $\tilde{\Psi}_k(s, \hat{\pi}_{k+1}(s))$ at step $k$ evaluated at $a = \hat{\pi}_k(s)$:

$$\tilde{\Psi}_k(s, \hat{\pi}_{k+1}(s)) + \tilde{\mu}_k D(\pi_{k+1}(s), \hat{\pi}_k(s)) \leq \tilde{\Psi}_k(s, \hat{\pi}_k(s)) + \epsilon_{\text{opt},k}(s).$$

Substituting $\tilde{\Psi}_k(s, \hat{\pi}_{k+1}(s))$ and using the recursive definition $\tilde{\Psi}_k(s, \hat{\pi}_k(s)) = \tilde{\Psi}_{k-1}(s, \hat{\pi}_k(s)) + \beta_k \tilde{\psi}(s, \hat{\pi}_k(s); \theta_k) + (\lambda_k - \lambda_{k-1}) D(\pi_0(s), \hat{\pi}_k(s))$, we obtain:

$$\begin{aligned}\beta_k \tilde{\psi}(s, \hat{\pi}_{k+1}(s); \theta_k) \leq & -\tilde{\mu}_k D(\pi_{k+1}(s), \hat{\pi}_k(s)) - \tilde{\mu}_{k-1} D(\pi_k(s), \hat{\pi}_{k+1}(s)) \\ & + \beta_k \tilde{\psi}(s, \hat{\pi}_k(s); \theta_k) + (\lambda_k - \lambda_{k-1}) D(\pi_0(s), \hat{\pi}_k(s)) \\ & + \epsilon_{\text{opt},k}(s) + \epsilon_{\text{opt},k-1}(s).\end{aligned}$$

Using the function approximation error from Assumption 1 and noting that $\psi^{\hat{\pi}_k}(s, \hat{\pi}_k(s)) = 0$, we have $\tilde{\psi}(s, \hat{\pi}_k(s); \theta_k) = \delta_k(s, \hat{\pi}_k(s))$. Rearranging terms yields:

$$\begin{aligned}\psi^{\hat{\pi}_k}(s, \hat{\pi}_{k+1}(s)) - & \frac{\lambda_k - \lambda_{k-1}}{\beta_k} D(\pi_0(s), \hat{\pi}_k(s)) - \frac{1}{\beta_k}[\epsilon_{\text{opt},k}(s) + \epsilon_{\text{opt},k-1}(s)] \\ & - [\delta_k(s, \hat{\pi}_k(s)) - \delta_k(s, \hat{\pi}_{k+1}(s))] \\ & \leq -\frac{\tilde{\mu}_k}{\beta_k} D(\pi_{k+1}(s), \hat{\pi}_k(s)) - \frac{\tilde{\mu}_{k-1}}{\beta_k} D(\pi_k(s), \hat{\pi}_{k+1}(s)) \leq 0.\end{aligned}$$

Using the Lipschitz property of $\delta_k$, we bound the terms involving $\delta_k$ by $\delta_k(s, \hat{\pi}_k(s)) - \delta_k(s, \hat{\pi}_{k+1}(s)) \leq (M_Q + M_{\tilde{Q}}) \|\hat{\pi}_{k+1}(s) - \hat{\pi}_k(s)\|$. From the 1-strong convexity of Bregman divergence, we have:

$$\begin{aligned}& -\frac{\tilde{\mu}_k}{\beta_k} D(\pi_{k+1}(s), \hat{\pi}_k(s)) - \frac{\tilde{\mu}_{k-1}}{\beta_k} D(\pi_k(s), \hat{\pi}_{k+1}(s)) \\ & \leq -\frac{\tilde{\mu}_k}{2\beta_k} \|\pi_{k+1}(s) - \hat{\pi}_k(s)\|^2 - \frac{\tilde{\mu}_{k-1}}{2\beta_k} \|\pi_k(s) - \hat{\pi}_{k+1}(s)\|^2 \\ & \leq \frac{\tilde{\mu}_k}{2\beta_k} \|\pi_{k+1}(s) - \hat{\pi}_{k+1}(s)\|^2 + \frac{\tilde{\mu}_{k-1}}{2\beta_k} \|\pi_k(s) - \hat{\pi}_k(s)\|^2 \\ & \quad - \frac{\tilde{\mu}_k + \tilde{\mu}_{k-1}}{4\beta_k} \|\hat{\pi}_{k+1}(s) - \hat{\pi}_k(s)\|^2.\end{aligned}$$

The second inequality comes from expanding the policy difference terms using the triangle inequality and Young's inequality to derive $\|a - b\|^2 \geq -\|a\|^2 + \frac{1}{2}\|b\|^2$.

Combining with the Lipschitz bound and using Young's inequality $ax - bx^2 \leq a^2/(4b)$ on the terms involving $\|\hat{\pi}_{k+1} - \hat{\pi}_k\|$:

$$\begin{aligned}& (M_Q + M_{\tilde{Q}}) \|\hat{\pi}_{k+1}(s) - \hat{\pi}_k(s)\| - \frac{\tilde{\mu}_k + \tilde{\mu}_{k-1}}{4\beta_k} \|\hat{\pi}_{k+1}(s) - \hat{\pi}_k(s)\|^2 \\ & \quad \leq \frac{\beta_k (M_Q + M_{\tilde{Q}})^2}{\tilde{\mu}_k + \tilde{\mu}_{k-1}}.\end{aligned}$$

From the 1-strong convexity of Bregman divergence and the Assumption 2, we have:

$$\tilde{\mu}_k \|\pi_{k+1}(s) - \hat{\pi}_{k+1}(s)\|^2 \leq 2\epsilon_{\text{opt},k}(s).$$

Since $\lambda_t = \lambda$ for all $t \geq -1$ under our convention, $\lambda_k - \lambda_{k-1} = 0$ for all $k \geq 0$, and the term $\frac{\lambda_k - \lambda_{k-1}}{\beta_k} D(\pi_0(s), \hat{\pi}_k(s))$ vanishes. Combine the relationships to result in the bound for the advantage:

$$0 \leq -\psi^{\hat{\pi}_k}(s, \hat{\pi}_{k+1}(s)) + \frac{2}{\beta_k}[\epsilon_{\text{opt},k}(s) + \epsilon_{\text{opt},k-1}(s)] + \frac{\beta_k (M_Q + M_{\tilde{Q}})^2}{\tilde{\mu}_k + \tilde{\mu}_{k-1}}. \tag{14}$$

Define constants $C_k \geq 0$ and $E_k \geq 0$ as:

$$C_k := \frac{\beta_k (M_Q + M_{\tilde{Q}})^2}{\tilde{\mu}_k + \tilde{\mu}_{k-1}},$$

$$E_k := \frac{4}{\beta_k}\epsilon \geq \frac{2}{\beta_k}[\epsilon_{\text{opt},k}(s) + \epsilon_{\text{opt},k-1}(s)].$$

Using the performance difference lemma (Equation 5), since $\kappa_s^{\hat{\pi}_{k+1}}$ is a probability measure, we can add and subtract a constant term in and outside the integral:

$$\begin{aligned} V^{\hat{\pi}_{k+1}}(s) - V^{\hat{\pi}_k}(s) &\leq \frac{1}{1-\gamma} \int \left[\psi^{\hat{\pi}_k}(q, \hat{\pi}_{k+1}(q)) - C_k - E_k\right] \kappa_s^{\hat{\pi}_{k+1}}(dq) + \frac{1}{1-\gamma}(C_k + E_k) \\ &\leq \frac{1}{1-\gamma} \left[\psi^{\hat{\pi}_k}(s, \hat{\pi}_{k+1}(s)) - C_k - E_k\right] \kappa_s^{\hat{\pi}_{k+1}}(\{s\}) + \frac{1}{1-\gamma}(C_k + E_k) \\ &\leq \psi^{\hat{\pi}_k}(s, \hat{\pi}_{k+1}(s)) + \frac{\gamma}{1-\gamma}(C_k + E_k), \end{aligned} \tag{15}$$

where the second inequality follows the fact $\kappa_s^{\hat{\pi}_{k+1}}(\{s\}) \geq 1 - \gamma$.

We define $S_k := \sum_{t=0}^{k-1} \beta_t [V^{\hat{\pi}_t}(s) - V^{\hat{\pi}_{t+1}}(s)]$. With $\beta_t = t + 1$,

$$S_k = \sum_{t=0}^{k-1} (t+1)[V^{\hat{\pi}_t}(s) - V^{\hat{\pi}_{t+1}}(s)] = \sum_{t=0}^{k-1} V^{\hat{\pi}_t}(s) - kV^{\hat{\pi}_k}(s) \leq \sum_{t=0}^{k-1} V^{\hat{\pi}_t}(s) - kV^{\pi^*}(s).$$

From Equation 14 and 15, we have:

$$V^{\hat{\pi}_{t+1}}(s) - V^{\hat{\pi}_t}(s) \leq \frac{1}{1-\gamma}(C_t + E_t).$$

Substituting the telescopic sum, we have:

$$V^{\hat{\pi}_t}(s) \leq V^{\hat{\pi}_0}(s) + \sum_{j=0}^{t-1} \frac{1}{1-\gamma}(C_j + E_j)$$

Substituting $V^{\hat{\pi}_t}(s)$ in $S_k$, we have

$$\begin{aligned} S_k &\leq \sum_{t=0}^{k-1} \left( V^{\hat{\pi}_0}(s) + \sum_{j=0}^{t-1} \frac{1}{1-\gamma}(C_j + E_j) \right) - kV^{\pi^*}(s) \\ &\leq kV^{\hat{\pi}_0}(s) + \frac{1}{1-\gamma} \sum_{t=0}^{k-1} \sum_{j=0}^{t-1} (C_j + E_j) - kV^{\pi^*}(s) \end{aligned}$$

Using the relationship $\sum_{t=0}^{k-1} \frac{\beta_t^2}{\tilde{\mu}_t + \tilde{\mu}_{t-1}} \leq \sum_{t=0}^{k-1} \frac{(t+1)^2}{|\tilde{\mu}_d|(k+1)k} \leq \frac{k}{|\tilde{\mu}_d|}$, we have:

$$S_k \leq k \left[ V^{\pi_0}(s) - V^{\pi^*}(s) + \frac{(M_Q + M_{\tilde{Q}})^2}{2(1-\gamma)|\tilde{\mu}_d|} + \frac{4\epsilon H_k}{1-\gamma} \right],$$

where $H_k = \sum_{j=0}^{k-1} \frac{1}{j+1}$ is the $k$-th harmonic number.

Rearranging Equation 15, we have:

$$0 \leq -\beta_k \left[\psi^{\hat{\pi}_k}(s, \hat{\pi}_{k+1}(s)) - (C_k + E_k)\right] \leq \beta_k \left[V^{\hat{\pi}_k}(s) - V^{\hat{\pi}_{k+1}}(s)\right] + \frac{\beta_k}{1-\gamma}(C_k + E_k).$$

Summing the inequality above from $t = 0$ to $k - 1$ and use the inequality for $S_k$:

$$\begin{aligned} 0 \leq \sum_{t=0}^{k-1} -\beta_t \left[\psi^{\hat{\pi}_t}(s, \hat{\pi}_{t+1}(s)) - (C_t + E_t)\right] \leq & k \left[V^{\pi_0}(s) - V^{\pi^*}(s)\right] \\ & + \frac{3k(M_Q + M_{\tilde{Q}})^2}{2(1-\gamma)|\tilde{\mu}_d|} + \frac{4\epsilon}{1-\gamma}(kH_k + k). \end{aligned}$$

Choosing index $\bar{k}(s)$ that minimizes $-\psi^{\hat{\pi}_t}(s, \hat{\pi}_{t+1}(s)) + C_t + E_t$ in the summation, and using $\beta_{\bar{k}(s)} = \bar{k}(s) + 1$, we derive the upper bound on the advantage for Theorem 2. The lower bound follows from Equation 14 at $\bar{k}(s)$.

To derive a bound for policy improvement, we return to Equation 14 without substituting $\|\hat{\pi}_{k+1}(s) - \hat{\pi}_k(s)\|$, we split the term using the triangle inequality and combine with the advantage bound:

$$\frac{\tilde{\mu}_k + \tilde{\mu}_{k-1}}{8\beta_k}\|\hat{\pi}_{k+1}(s) - \hat{\pi}_k(s)\|^2 \leq -\psi^{\hat{\pi}_k}(s, \hat{\pi}_{k+1}(s)) + \frac{4\epsilon}{\beta_k} + \frac{2\beta_k(M_Q + M_{\tilde{Q}})^2}{\tilde{\mu}_k + \tilde{\mu}_{k-1}}.$$

Combining with Equation 13 and the fact that $0 \leq \frac{\beta_{\bar{k}(s)}}{\tilde{\mu}_{\bar{k}(s)} + \tilde{\mu}_{\bar{k}(s)-1}} \leq \frac{1}{|\tilde{\mu}_d|(k+1)}$, we have:

$$\begin{aligned}\|\hat{\pi}_{\bar{k}(s)+1}(s) - \hat{\pi}_{\bar{k}(s)}(s)\|^2 \leq \frac{8}{|\tilde{\mu}_d|(k+1)}\Bigg[&\frac{2\left[V^{\pi_0}(s) - V^{\pi^*}(s)\right]}{k+1} \\ &+ \frac{3(M_Q + M_{\tilde{Q}})^2}{(1-\gamma)|\tilde{\mu}_d|(k+1)} + \frac{2(M_Q + M_{\tilde{Q}})^2}{|\tilde{\mu}_d|(k+1)} \\ &+ \frac{8\epsilon(H_k + 1)}{(1-\gamma)(k+1)} + \frac{4\epsilon}{\bar{k}(s)+1}\Bigg]\end{aligned}$$

□

# C Implementation Details

## C.1 Sum Advantage Estimation

To maintain numerical stability during training and avoid performance degradation when dealing with large values of the cumulative summation term in the PDA objective in Algorithm 2, we implement a scaled version of the objective function. The scaling factors are chosen to ensure that the weighting remains consistent with the original theoretical framework. In addition, we use a recursive scheme to store the weighted sum of advantages. Specifically, the accumulated advantage is updated as

$$\psi^{\Sigma}_{\theta_\psi}(s,a) \approx \left(1 - \frac{\beta_k}{\sum_{i=0}^{k}\beta_i}\right)\psi^{\Sigma}_{\text{old}}(s,a) + \frac{\beta_k}{\sum_{i=0}^{k}\beta_i}\tilde{\psi}(s,a;\theta_k).$$

During training, the previous sum advantage prediction is detached from the computational graph before constructing the new regression target, preventing gradients from propagating through earlier advantage models. The recursion therefore compresses the normalized weighted history into a fixed-size network rather than retaining and re-evaluating all previous models.

With this representation, the action can be obtained by solving the following optimization problem:

$$\pi_{k+1}(s) = \arg\min_{a\in\mathcal{A}} \tilde{\Psi}'(s,a) = \arg\min_{a\in\mathcal{A}}\left[\psi^{\Sigma}_{\theta_\psi}(s,a) + \frac{\lambda_k}{\sum_{i=0}^{k}\beta_i}D(\pi_0(s),a)\right],$$

where $\tilde{\Psi}'$ represents the scaled objective with parameterized accumulated advantage.

For problems with continuous action space, we choose the Euclidean norm as the distance generating function, hence the corresponding Bregman divergence term for PDA is $\frac{1}{2}\|a - \pi_0(s)\|_2^2$. The prox-center $\pi_0(s)$ can be a random initial policy, a pretrained policy, or simply set to zero. The approximate solution of the optimization subproblem can then be obtained with standard optimizers, with gradients obtained through backpropagation of action $a$ on $\tilde{\Psi}'$.

We fix the schedules before training, using $\beta_k = k + 1$ and $\lambda_k = \lambda(k+1)^{3/2}$. This follows the theoretical $\tilde{\mu}_d = 0$ rule as an empirically selected implementation choice and leaves $\lambda$ as a tunable hyperparameter.

## C.2 Summary of Notation

For clarity, the notation below summarizes the key symbols used. The weights $\beta_k$ determine how advantage information is accumulated, $\lambda_k$ controls regularization around the fixed prox-center, $\theta_k$ parameterizes the current approximate advantage, $\theta_\psi$ parameterizes the sum advantage network, and $\theta_\pi$ parameterizes the actor. The exact action $\pi_{k+1}(s)$ minimizes the cumulative objective, whereas $\hat{\pi}_{k+1}(s;\theta^\pi_k)$ is the actor approximation used by AA-PDA.

- The cumulative regularized objective $\tilde{\Psi}_k(s,a) := \sum_{t=0}^{k} \beta_t \tilde{\psi}(s,a;\theta_t) + \lambda_k D(\pi_0(s),a)$.
- The optimization subproblem: $\pi_{k+1}(s) = \arg\min_{a\in\mathcal{A}} \tilde{\Psi}_k(s,a)$.
- $\pi_{k+1}(s)$ denotes the exact minimizer of the cumulative objective $\tilde{\Psi}_k$.
- $\psi^{\pi}(s,a)$: the true advantage function under policy $\pi$.
- $\hat{\pi}_{k+1}(s;\theta_k^{\pi})$: the approximate solution produced by the actor network, i.e., $\pi_{\theta_\pi}$ in Algorithm 3 after iteration $k$.
- $\tilde{\psi}(s,a;\theta_k)$: the approximate advantage function parameterized by $\theta_k$.
- The sum advantage network estimates the scaled weighted sum advantage:

$$\psi^{\Sigma}_{\theta_\psi}(s,a) \approx \left(1 - \frac{\beta_k}{\sum_{i=0}^{k}\beta_i}\right)\psi^{\Sigma}_{\text{old}}(s,a) + \frac{\beta_k}{\sum_{i=0}^{k}\beta_i}\tilde{\psi}(s,a;\theta_k).$$

- The scaled cumulative regularized objective:

$$\tilde{\Psi}'(s,a) = \psi^{\Sigma}_{\theta_\psi}(s,a) + \frac{\lambda_k}{\sum_{i=0}^{k}\beta_i} D(\pi_0(s),a).$$

### C.3 Algorithm Hyperparameters

Table 3: Hyperparameters for the learned-policy benchmarks and the direct-PDA action solvers. The direct solvers share the AA-PDA policy-evaluation settings and differ only in the per-query action optimization settings listed in the lower panel. PDA-RSGF and PDA-ACFGM solve each action subproblem directly with the randomized stochastic gradient-free (RSGF) method and the auto-conditioned fast gradient method (ACFGM), respectively (Appendix D.3).

| **Hyperparameter** | **AA-PDA** | **NPG** | **PPO** | **TRPO** | **SAC** | **TD3** |
|---|---|---|---|---|---|---|
| Hidden sizes | $[64, 64]$ | $[64, 64]$ | $[64, 64]$ | $[64, 64]$ | $[256, 256]$ | $[256, 256]$ |
| Activation | Tanh | Tanh | Tanh | Tanh | ReLU | ReLU |
| Learning rate | $1 \times 10^{-3}$ | $1 \times 10^{-3}$ | $3 \times 10^{-4}$ | $1 \times 10^{-3}$ | $1 \times 10^{-3}$ | $3 \times 10^{-4}$ |
| Batch size | 1000 | 1024 | 64 | 1024 | 256 | 256 |
| Discount factor | 0.99 | 0.99 | 0.99 | 0.99 | 0.99 | 0.99 |
| GAE $\lambda$ | 0.95 | 0.95 | 0.95 | 0.95 | — | — |
| TD n-step | — | — | — | — | 1 | 1 |
| Learning rate decay | None | True | True | True | None | None |
| Max gradient norm | 0.1 | — | 0.5 | — | — | — |
| *Algorithm-Specific Hyperparameters* | | | | | | |
| *AA-PDA* | | | | | | |
| Regularization step size | 0.5 | — | — | — | — | — |
| Action noise | 1.3 | — | — | — | — | — |
| *NPG* | | | | | | |
| Actor step size | — | 0.1 | — | — | — | — |
| Critic optimization iterations | — | 20 | — | — | — | — |
| *PPO* | | | | | | |
| Clipping parameter | — | — | 0.2 | — | — | — |
| Value function coefficient | — | — | 0.25 | — | — | — |
| Entropy coefficient | — | — | 0.0 | — | — | — |
| Value clipping | — | — | None | — | — | — |
| Dual clipping | — | — | None | — | — | — |
| *TRPO* | | | | | | |
| Max KL divergence | — | — | — | 0.01 | — | — |
| Backtrack coefficient | — | — | — | 0.8 | — | — |
| Max backtracks | — | — | — | 10 | — | — |
| Critic optimization iterations | — | — | — | 20 | — | — |
| *SAC* | | | | | | |
| Soft update parameter | — | — | — | — | 0.005 | — |
| Entropy coefficient | — | — | — | — | 0.2 | — |
| Warmup steps | — | — | — | — | 10000 | — |
| *TD3* | | | | | | |
| Soft update parameter | — | — | — | — | — | 0.005 |
| Exploration noise | — | — | — | — | — | 0.1 |
| Policy noise | — | — | — | — | — | 0.2 |
| Noise clip | — | — | — | — | — | 0.5 |
| Actor update frequency | — | — | — | — | — | 2 |
| Warmup steps | — | — | — | — | — | 25000 |

| **Direct-PDA action-solver setting** | **PDA-RSGF** | **PDA-ACFGM** |
|---|---|---|
| Action iterations per query | 75 | 75 |
| Smoothing radius $\mu$ | $10^{-4}$ | — |
| Action learning rate | $10^{-2}$ | — |
| Momentum / Nesterov | 0.9 / True | — |
| ACFGM $\beta$ | — | 0.3 |

## C.4 Sensitivity to $\sigma_0$ and $\lambda$

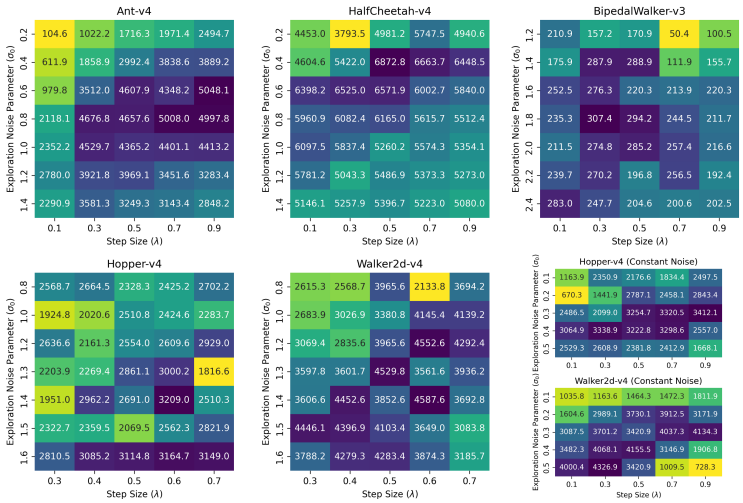



Figure 3: Sensitivity to exploration noise $\sigma_0$ and step size $\lambda$. Each cell is the final-five-epoch mean testing return. Decreasing-noise panels use five seeds and the constant-noise panels use three, with ten tests per seed. The heatmap is descriptive rather than a confidence-band plot.

RL performance is often highly sensitive to hyperparameter choices [Henderson et al., 2018], hence fully realizing an algorithm's potential requires careful tuning. Throughout the benchmark, we apply a fixed set of hyperparameters across all environments, with AA-PDA-specific hyperparameters, step size set to $\lambda = 0.5$ and the exploration noise standard deviation parameter set to $\sigma_0 = 1.3$. To better understand the robustness of AA-PDA and the effect of tuning, we conduct a sensitivity study over $\lambda$ and $\sigma_0$ across five continuous-control environments: Hopper, Walker2d, Ant, HalfCheetah, and BipedalWalker (Figure 3). Sweeping through a range of parameters shows the sensitivity of AA-PDA with respect to hyperparameter tuning and helps to explain the rationale for parameter selection.

In dynamic balancing tasks (Hopper, Walker2d, BipedalWalker), higher exploration noise is generally favored, which facilitates the discovery of stabilizing behaviors in the state-action space. In contrast, quadrupedal locomotion tasks (Ant and HalfCheetah) are more stable due to multiple contact points to the floor, which require lower exploration noise and larger step sizes to achieve better performance. In this case, increased step sizes provide stronger regularization and improve training efficiency. Interestingly, the initial performance of the Ant-v4 training curve (Figure 1) is due to the healthy reward gained from stable behaviors by AA-PDA regularization. The two algorithm-specific hyperparameters ($\lambda$ and $\sigma_0$) have intuitive interpretations about the algorithm's training dynamics. Hence, physical intuitions about task dynamics and stability can guide the identification of effective parameter regimes. Importantly, there exists a broad region of hyperparameters that yields competitive results. Parameter-free methods for RL are an active research topic even in basic settings [Ju and Lan, 2025]. Integrating them with AA-PDA is left for future research.

In addition, we explore the option of using constant noise $\sigma(k) = \sigma_0$, illustrated as the last two plots in Figure 3. The results indicate that constant noise can also be advantageous, particularly for environments that require stronger and sustained exploration like Hopper and Walker2d. However, it is challenging to identify a single $\sigma_0$ that performs well across all environments at the same time. A decreasing noise schedule traverses a wider range of noise levels, hence making it more adaptive and robust across different tasks.

## C.5 Choice of Optimizer

We utilize the SOAP [Vyas et al., 2024] optimizer for neural network training in AA-PDA. SOAP employs Kronecker-factored preconditioning, which calculates and maintains preconditioner matrices that approximate second-order curvature information. As a result, it benefits from a larger batch size and fewer iterations for training. We choose this optimizer for its strong empirical performance in

neural network training and because it is readily applicable in RL training environments, even though the theoretical underpinnings are generally lacking.

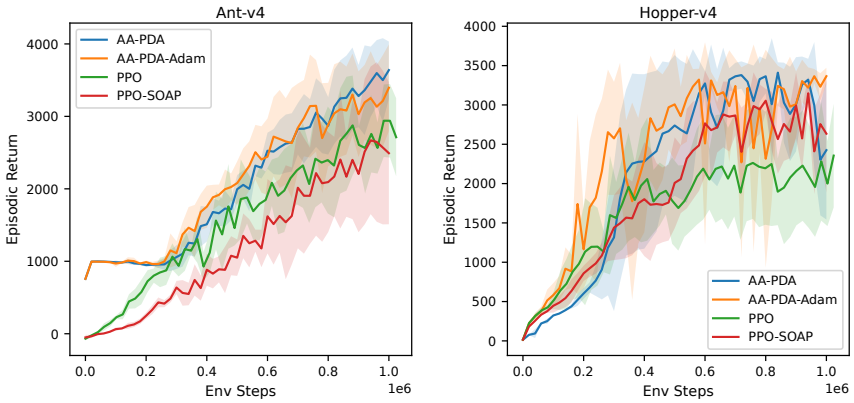



Figure 4: Optimizer ablation on Ant-v4 and Hopper-v4 over five seeds. Curves show the seed-level mean evaluation return with shaded $\pm 2\,\mathrm{SEM}$ regions. AA-PDA uses SOAP by default, AA-PDA-Adam substitutes Adam for SOAP, and PPO-SOAP substitutes SOAP for PPO's default Adam optimizer.

Figure 4 compares AA-PDA (using SOAP by default) with AA-PDA-Adam, as well as PPO (using Adam [Kingma and Ba, 2015] by default) with PPO-SOAP, for the Ant and Hopper environments. In both benchmarks, AA-PDA and AA-PDA-Adam show similar learning curves, and both variants can achieve higher episodic rewards than PPO. Although SOAP accelerates training in our implementation, it does not provide a consistent improvement in sample efficiency or final performance for AA-PDA relative to Adam. Moreover, substituting Adam with SOAP in PPO does not lead to consistent performance gains. These results indicate that while SOAP is an effective tool for accelerating computation, it is not a primary contributor to AA-PDA's performance advantages.

# D Actor Approximation and Diagnostics

## D.1 Visualization of Optimum Tracking

We use the inverted pendulum swing-up (Pendulum-v1) environment from Gymnasium [Towers et al., 2025] as an example to visualize the behavior of the actor. In this problem, the state is defined as $s = [\cos(\theta), \sin(\theta), \dot{\theta}]$, where $\theta$ and $\dot{\theta}$ represent the pendulum angle and angular velocity, respectively. The goal of this problem is to apply a torque $\tau$ to maintain the pendulum at its upright position. By fixing the angular velocity $\dot{\theta}$, we can visualize the landscape of $\tilde{\Psi}'$ over the $(\theta, \tau)$ domain, which allows for observing how the actor learns to track the solution of the optimization subproblem. As shown in Figure 5, the first three plots track the evolution of the PDA objective ($\tilde{\Psi}'$) and the learning trajectory of the actor, confirming that the actor successfully tracks the optimum and solves the problem. The final plot demonstrates that the approximation error of the actor stabilizes during the extended training process.

## D.2 Actor Approximation Quality

Similar to the Pendulum-v1 diagnostic, we empirically evaluate the quality of the actor approximation in higher-dimensional environments. For Ant-v4, HalfCheetah-v4, Hopper-v4, and Walker2d-v4, we take snapshots throughout training and compare the actor output with the best-found solution at a set of fixed states. We evaluate 256 states at 200 checkpoints across 4M environment steps. At each state, we compare the actor output with a best-found reference solution of $\tilde{\Psi}_k(s, a)$ over $a$, obtained using multistart projected Adam.

Figure 6 reports the resulting normalized objective difference. Curves show the mean and median of the gap over the evaluated states, averaged over ten seeds, with $\pm 2\,\mathrm{SEM}$ across seeds. Over this

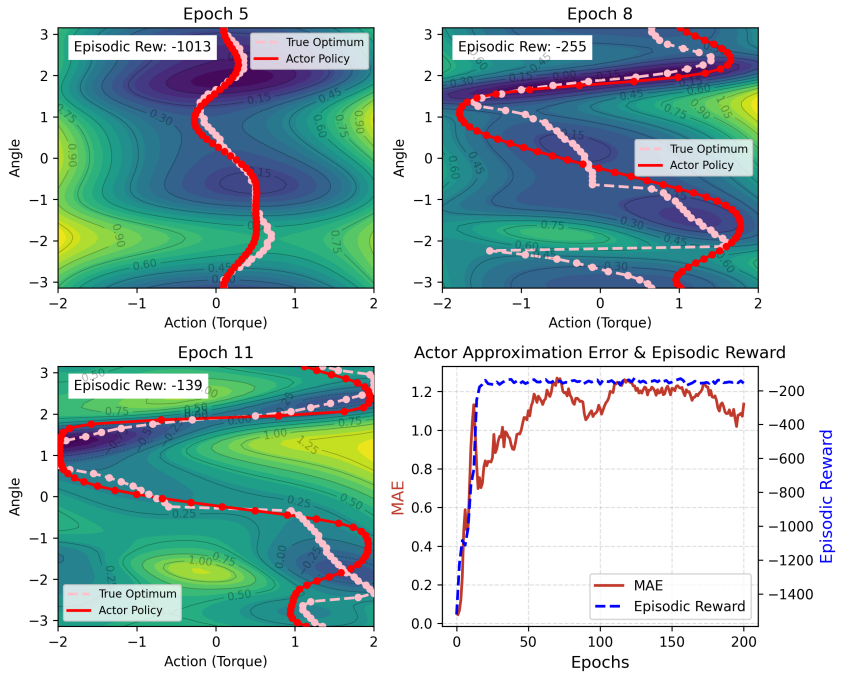



Figure 5: Visualization of optimum tracking with the actor in the Pendulum-v1 environment. The landscape evolution of the scaled optimization subproblem $\tilde{\Psi}'$ over epochs 5, 8, and 11, at a fixed $\dot{\theta} = 0.2$ is shown in the first three plots. The pink dashed line represents the true optimum of $\tilde{\Psi}'$, and the solid red line represents the output of the actor network. The final plot tracks the mean absolute error (MAE) between the true optimum and the actor output averaged across a range of states at each epoch for an extended training process.

extended training, the normalized gap rises early in training and then levels off rather than diverging. These finite sampled results provide empirical support for the actor approximation quality.

### D.3 Direct PDA Comparisons with AA-PDA

We compare actor-accelerated PDA (AA-PDA) with direct PDA baselines. Within the same framework, PDA-RSGF uses the randomized stochastic gradient-free (RSGF) method [Ghadimi and Lan, 2013] and PDA-ACFGM uses the auto-conditioned fast gradient method [Li and Lan, 2025a] to optimize $\tilde{\Psi}_k$ directly whenever an action is required. Figures 7–9 compare the performance and training time of the three methods across four MuJoCo environments.

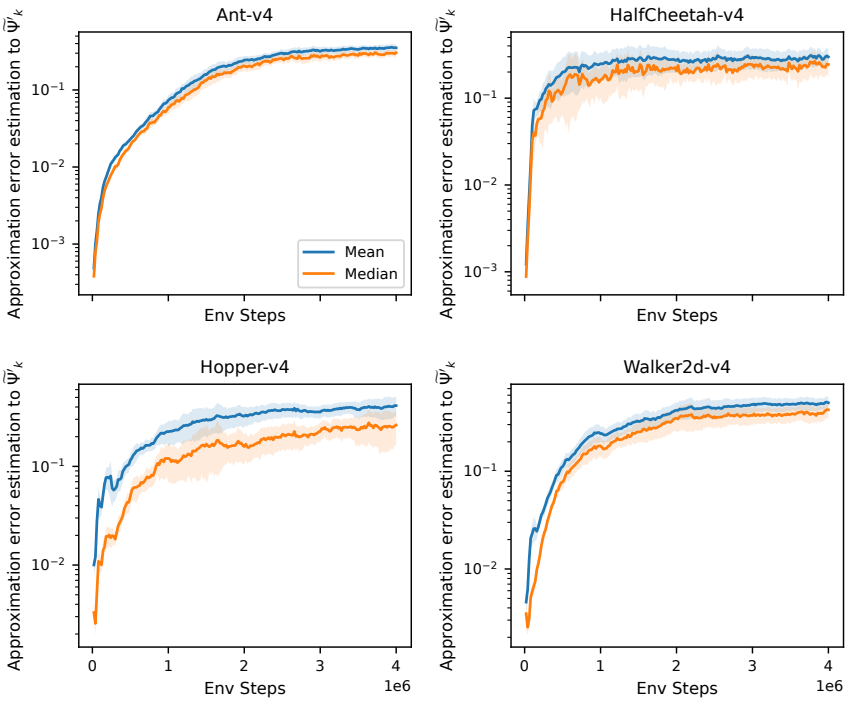



Figure 6: Actor approximation quality on Ant-v4, HalfCheetah-v4, Hopper-v4, and Walker2d-v4 over 4M environment steps. At each of 200 training checkpoints, 256 fixed states are evaluated for each of ten seeds. For each state, the reference action is obtained by selecting the best sampled action and refining it with multistart projected Adam. Curves report the mean and median of the normalized objective gap between the actor action and the reference action over the 256 states, averaged over seeds, with shaded regions showing $\pm 2\,\mathrm{SEM}$ across seeds.

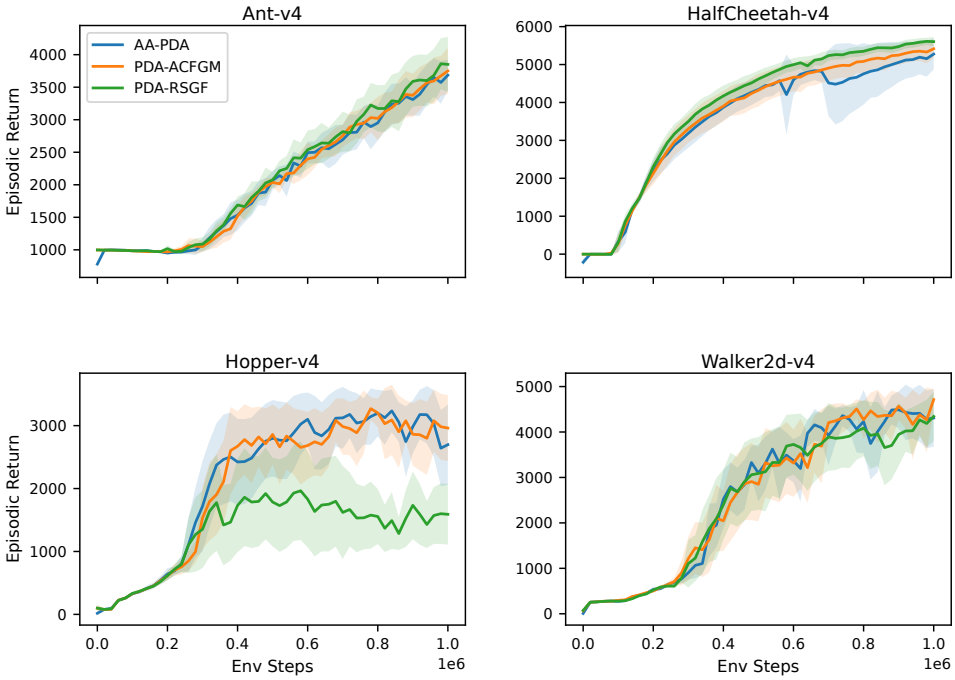



Figure 7: AA-PDA, PDA-RSGF, and PDA-ACFGM returns on Ant-v4, HalfCheetah-v4, Hopper-v4, and Walker2d-v4. Curves show the mean over ten seeds, where each seed is evaluated over ten test episodes. Shaded regions indicate $\pm 2\,\mathrm{SEM}$ across seed-level means.

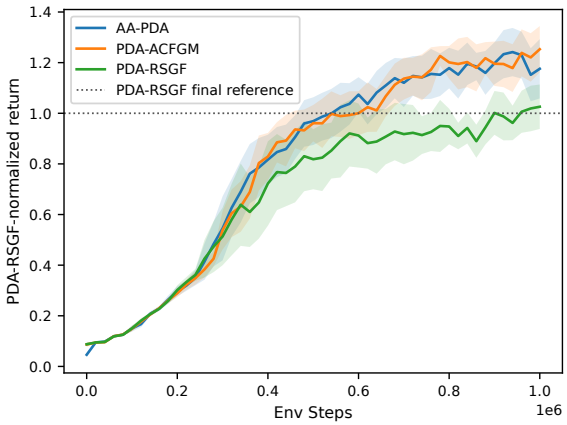



Figure 8: Aggregate return for AA-PDA, PDA-RSGF, and PDA-ACFGM on four MuJoCo-v4 tasks and ten seeds, normalized by the PDA-RSGF final-return reference. Curves show the seed mean with $\pm 2$ SEM.

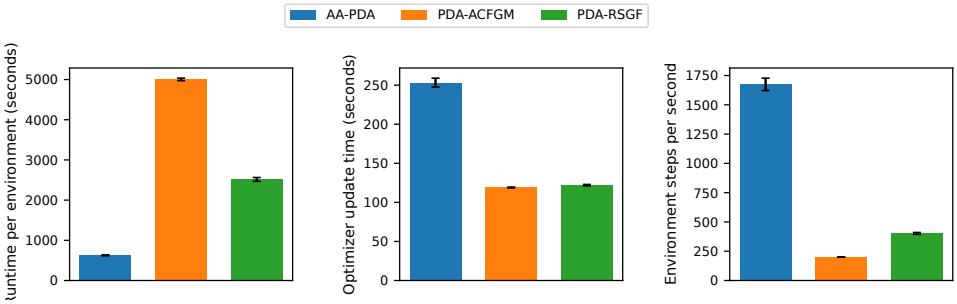



Figure 9: PDA runtime, optimizer-update time, and environment throughput. Values are first averaged over four environments within seed and then summarized across ten matched seeds as mean $\pm 2$ SEM.

At 1M steps, aggregate normalized return is $1.176 \pm 0.117$ for AA-PDA, $1.026 \pm 0.088$ for PDA-RSGF, and $1.253 \pm 0.092$ for PDA-ACFGM. Average runtime per environment is $626.4 \pm 12.2$ seconds for AA-PDA, compared with $2515.9 \pm 47.3$ and $5004.0 \pm 31.4$ seconds for PDA-RSGF and PDA-ACFGM, respectively. Thus, in these experiments, the learned actor provides an approximately 4–8 times speedup with comparable aggregate performance. Moreover, directly optimizing each action with RSGF or ACFGM introduces additional optimizer-specific hyperparameters that must be tuned.

### D.4 Training and Approximation Diagnostics

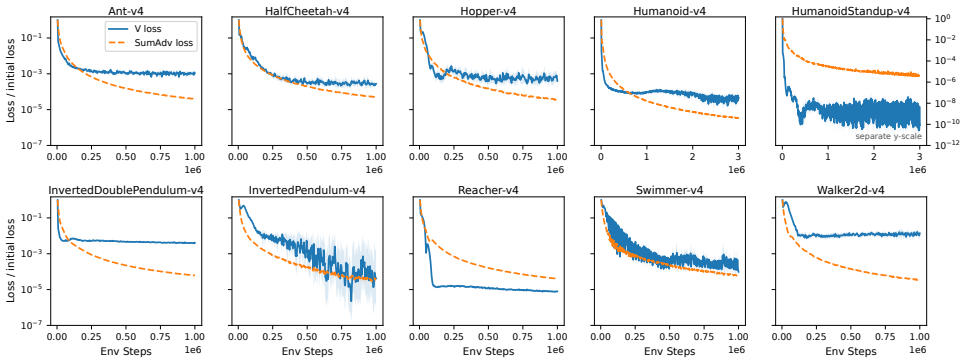



Figure 10: Raw value network and sum advantage network fitting losses on ten MuJoCo-v4 environments and ten seeds. Curves show the seed-level mean, with shaded region indicating $\pm 2\,\mathrm{SEM}$.

From an implementation perspective, AA-PDA uses generalized advantage estimation (GAE), as PPO does, so the accuracy of the estimated advantages depends largely on the accuracy of the value-function estimate. As empirical fitting diagnostics, we evaluate the training losses for both the value function and the sum-advantage estimates across MuJoCo environments using ten random seeds. Figure 10 records the corresponding training curves, and both losses decrease by orders of magnitude across the environments.

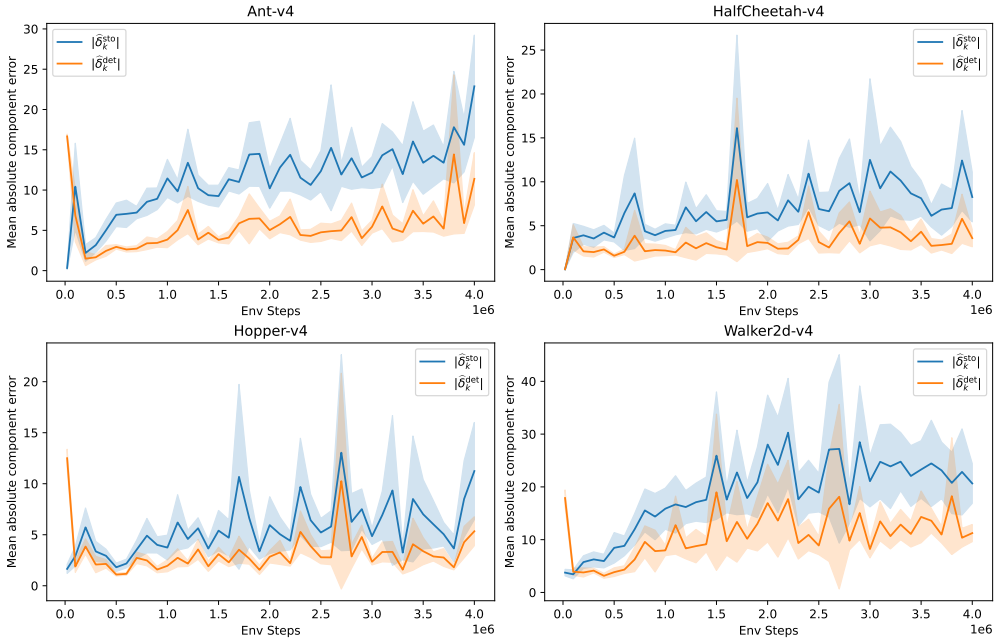



Figure 11: Sampled decomposition of advantage approximation error into stochastic and deterministic components. Curves report the mean over ten seeds with shaded regions showing $\pm 2\,\mathrm{SEM}$.

Because the true advantage error cannot be estimated exactly except in simple cases such as finite-state settings, we use sampled states and trajectories to construct rough empirical diagnostics. Figures 11 and 12 evaluate training checkpoints every 20k environment steps up to 4M for Ant-v4, HalfCheetah-v4, Hopper-v4, and Walker2d-v4. At each checkpoint, 16 states are evaluated for each of ten seeds using a rollout horizon of 1,000 steps.

For each seed-checkpoint pair, we approximate the advantage-estimation error as the difference between the GAE estimate from AA-PDA and a 1,000-environment-step rollout advantage. Figure 11 decomposes this sampled estimate into stochastic and deterministic components following their definitions. Note that no separate bound is imposed on $\delta_k^{sto}$, and $\delta_k^{det}$ is bounded only at the actor (executed-policy) and optimal actions through $\varsigma$ in Assumption 1. Across the four environments, the sampled stochastic component is generally larger and more variable than the deterministic component, although their magnitudes and evolution are task-dependent.

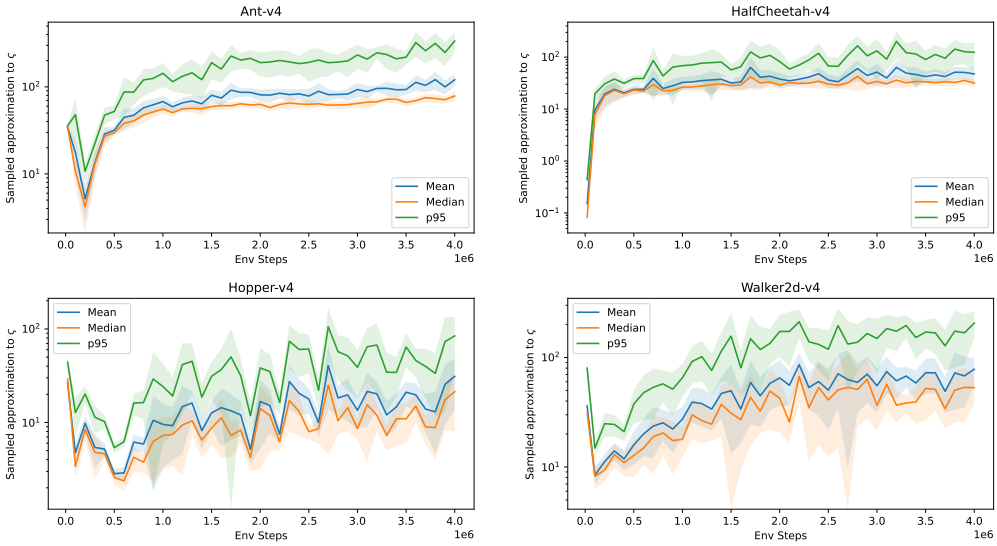



Figure 12: Sampled approximation to $\varsigma$. For each state, the finite candidate set contains the best-found reference action and eight alternatives. Mean, median, and p95 are computed over 16 states within each seed. Curves then report the across-seed mean with shaded regions showing $\pm 2\,\mathrm{SEM}$. The p95 is the point statistic, while the shading is its seed-level uncertainty.

The constant $\varsigma$ is the sum of two deterministic error sources: function-approximation error and bias of the learned advantage function. In general, these terms cannot be evaluated exactly because their values depend on the unknown optimal solution. Moreover, when a neural network is used for policy evaluation, computing the corresponding globally optimal approximation is generally non-convex and may be computationally intractable. The analysis therefore assumes that these terms are absolutely bounded through Assumption 1. Similar to the advantage-error diagnostic, we use sampled states and trajectories to construct an empirical diagnostic for $\varsigma$, following the definition in Assumption 1 with $\pi^*$ approximated by the best-found action. In Figure 12, the 95th-percentile curve over sampled states remains within a finite observed range over 4M steps, substantially beyond the 1M-step benchmarking horizon, providing empirical support for the assumption.

# E Benchmark Results

## E.1 Algorithm Runtime on MuJoCo-v4

Table 4: Wall-clock comparisons for the algorithms. The upper panel reports five parallel fixed-budget runs on an Intel i7-14700K. SAC and TD3 additionally use an NVIDIA RTX 4070. Times are seconds, reported as mean ± standard deviation, and ratios are relative to AA-PDA within each environment group. The lower panel reports the matched direct-PDA comparison over Ant, HalfCheetah, Hopper, and Walker2d: runtime is first averaged over the four environments within seed and then summarized over ten seeds as mean $\pm 2\,\mathrm{SEM}$.

| **Algorithm** | **MuJoCo-v4 (w/o humanoid)** | **MuJoCo-v4 (humanoid variants)** |
|---|---|---|
| AA-PDA | 423.4 ± 104.8 **(1.00x)** | 1480.7 ± 178.2 **(1.00x)** |
| PPO | 932.1 ± 120.1 **(2.20x)** | 1960.8 ± 167.2 **(1.32x)** |
| NPG | 344.1 ± 108.4 **(0.81x)** | 1011.3 ± 137.0 **(0.68x)** |
| TRPO | 349.5 ± 112.6 **(0.83x)** | 1009.6 ± 127.6 **(0.68x)** |
| SAC | 20335.9 ± 1878.3 **(48.03x)** | 19531.4 ± 2181.6 **(13.19x)** |
| TD3 | 11571.7 ± 510.6 **(27.33x)** | 29113.7 ± 596.8 **(19.66x)** |

*Matched direct-PDA comparison (four environments, ten seeds)*

| **Algorithm** | **Runtime/environment (seconds)** | **Ratio to AA-PDA** |
|---|---|---|
| AA-PDA | 626.4 ± 12.2 | **1.00x** |
| PDA-RSGF | 2515.9 ± 47.3 | **4.02x** |
| PDA-ACFGM | 5004.0 ± 31.4 | **7.99x** |

## E.2 On-Policy Benchmark Results

Table 5: Performance comparison with on-policy baselines. Each cell shows the mean $\pm 2\,\mathrm{SEM}$ over ten seed-level means, after averaging the final five evaluations per seed. All environments are trained for 1M environment steps unless marked (3M).

| Environment | AA-PDA | NPG | PPO | TRPO |
|---|---|---|---|---|
| HalfCheetah-v4 | **5174.6 ± 456.2** | 3556.4 ± 553.1 | 4067.5 ± 346.1 | 4496.8 ± 636.7 |
| Ant-v4 | **3568.2 ± 285.6** | 2002.5 ± 131.0 | 2589.5 ± 244.0 | 2807.1 ± 387.4 |
| Hopper-v4 | **2944.3 ± 454.7** | 1650.8 ± 460.9 | 2329.7 ± 354.3 | 2017.0 ± 548.1 |
| Walker2d-v4 | **4367.1 ± 547.5** | 2923.2 ± 435.7 | 3277.0 ± 457.6 | 4128.8 ± 351.3 |
| InvertedPendulum-v4 | **1000.0 ± 0.0** | **1000.0 ± 0.0** | **993.3** ± 13.5 | 380.3 ± 257.7 |
| InvertedDoublePendulum-v4 | **9167.5 ± 366.6** | 7967.2 ± 516.9 | 7926.8 ± 637.9 | 2723.4 ± 1469.0 |
| Reacher-v4 | **-4.0 ± 0.1** | -5.7 ± 0.4 | -18.7 ± 11.7 | -6.1 ± 0.4 |
| Swimmer-v4 | **111.3 ± 19.5** | 25.1 ± 6.1 | 54.2 ± 9.3 | 35.0 ± 18.0 |
| Humanoid-v4 | **760.1 ± 91.1** | **745.1** ± 87.1 | 669.2 ± 67.6 | 719.5 ± 66.2 |
| Humanoid-v4 (3M) | **5020.0 ± 277.9** | 4650.6 ± 386.0 | 933.3 ± 191.4 | 4745.1 ± 339.0 |
| HumanoidStandup-v4 | **161184.5 ± 1839.7** | 38364.7 ± 2454.5 | 135853.7 ± 12646.5 | 36737.7 ± 1413.0 |
| LunarLander-v3 | **204.7 ± 25.8** | 34.1 ± 37.5 | **204.4** ± 24.3 | -83.0 ± 33.2 |
| BipedalWalker-v3 | 149.0 ± 69.5 | 212.3 ± 21.5 | **251.6 ± 20.7** | 106.3 ± 67.8 |
| NewsvendorEnv-v0 | **-1.52e5±9.32e4** | -2.74e7±3.62e7 | -1.92e5±2.40e5 | -3.37e7±3.45e7 |
| NewsvendorEnv-v0 (3M) | **2.19e4±0.73e4** | -1.02e7±1.54e7 | -3.72e4±2.65e4 | -5.26e7±6.75e7 |
| PortfolioOptEnv-v0 | **10550.9 ± 367.9** | 10095.3 ± 543.2 | 10062.3 ± 360.9 | 10195.6 ± 510.9 |
| InvManagementBacklogEnv-v0 | 491.6 ± 5.1 | 430.4 ± 20.4 | **496.4 ± 2.6** | 397.5 ± 16.6 |
| InvManagementLostSalesEnv-v0 | **472.3 ± 4.0** | 432.3 ± 4.0 | 447.4 ± 10.4 | 431.5 ± 2.7 |

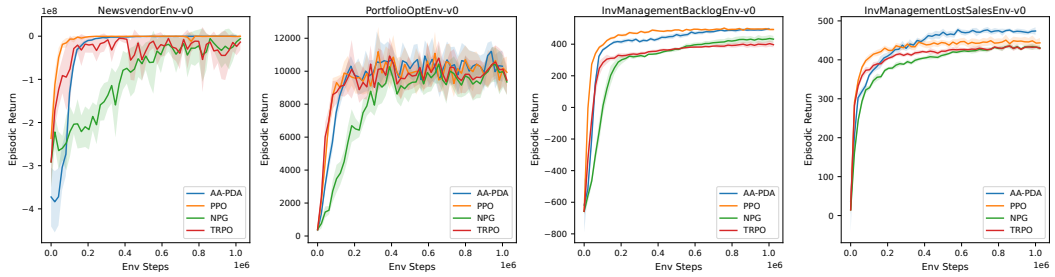



Figure 13: AA-PDA, PPO, NPG, and TRPO learning curves on four OR-Gym environments. Curves show the mean across ten seeds, where each seed is evaluated over ten test episodes. Shaded regions indicate $\pm 2\,\mathrm{SEM}$ across seed-level means.

### E.3 Training Curves for OR-Gym

### E.4 Benchmark Results and Training Curves for Comparison with Off-Policy Methods

Table 6: Performance comparison with off-policy baselines. Each cell shows the mean $\pm 2\,\mathrm{SEM}$ over ten seed-level means, after averaging the final five evaluations per seed. All environments are trained for 1M environment steps unless marked (3M).

| Environment | AA-PDA | SAC | TD3 |
|---|---|---|---|
| HalfCheetah-v4 | 5174.6 ± 456.2 | **11892.1 ± 494.2** | 10228.4 ± 417.5 |
| Ant-v4 | 3568.2 ± 285.6 | **5764.8 ± 167.0** | 3914.1 ± 881.7 |
| Hopper-v4 | 2944.3 ± 454.7 | **3387.0 ± 125.6** | 2896.6 ± 465.8 |
| Walker2d-v4 | **4367.1 ± 547.5** | 3864.3 ± 509.6 | 3435.5 ± 381.2 |
| InvertedPendulum-v4 | **1000.0 ± 0.0** | **1000.0 ± 0.0** | 783.5 ± 261.6 |
| InvertedDoublePendulum-v4 | 9167.5 ± 366.6 | **9183.2 ± 312.9** | 8134.8 ± 1833.3 |
| Reacher-v4 | -4.0 ± 0.1 | **-3.6 ± 0.1** | -3.7 ± 0.1 |
| Swimmer-v4 | **111.3 ± 19.5** | 43.5 ± 0.8 | 93.3 ± 16.3 |
| Humanoid-v4 | 760.1 ± 91.1 | 4785.3 ± 372.3 | **4846.9 ± 124.9** |
| HumanoidStandup-v4 | **161184.5 ± 1839.7** | 146249.0 ± 8054.1 | 107311.1 ± 12563.0 |
| LunarLander-v3 | 204.7 ± 25.8 | **284.6 ± 1.2** | 229.0 ± 64.7 |
| BipedalWalker-v3 | 149.0 ± 69.5 | -9.1 ± 5.6 | **295.2 ± 8.4** |
| NewsvendorEnv-v0 | -152387.9 ± 93157.7 | **-19212.1 ± 1583.3** | -19758.2 ± 844.6 |
| PortfolioOptEnv-v0 | **10550.9 ± 367.9** | 9010.2 ± 476.1 | — |
| InvManagementBacklogEnv-v0 | **491.6 ± 5.1** | 281.8 ± 24.4 | -1575.7 ± 443.5 |
| InvManagementLostSalesEnv-v0 | **472.3 ± 4.0** | 356.5 ± 6.1 | -311.9 ± 312.9 |

Table 6 reports the final returns. Figures 14 and 15 show that no method dominates every task. SAC has the strongest aggregate continuous-control result, while AA-PDA and TD3 have overlapping aggregate IQM intervals and differ substantially across individual environments. On OR-Gym, SAC is strongest on Newsvendor, AA-PDA is strongest on PortfolioOpt and both InvManagement settings.

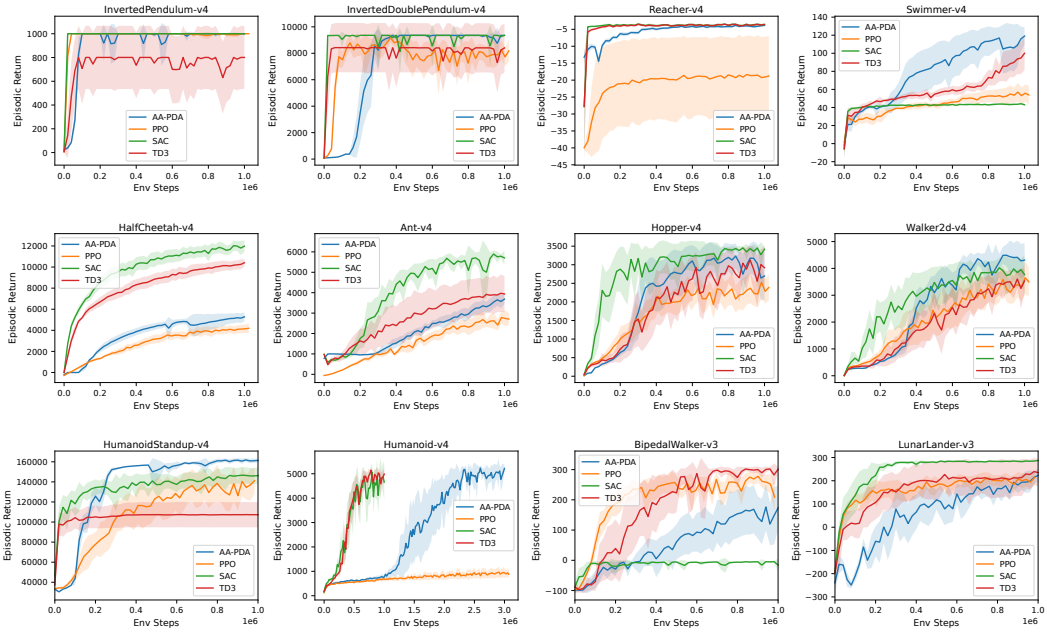



Figure 14: AA-PDA, PPO, SAC, and TD3 learning curves on MuJoCo-v4 and Box2D. Curves show the mean over ten seeds, where each seed is evaluated over ten test episodes. Shaded regions indicate $\pm 2\,\mathrm{SEM}$ across seed-level means.

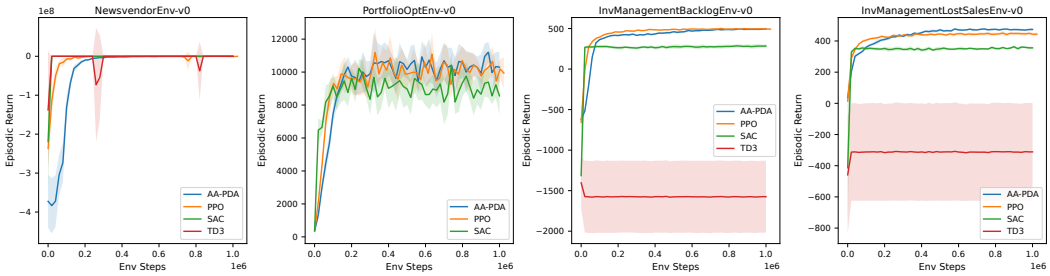



Figure 15: AA-PDA, PPO, SAC, and TD3 learning curves on OR-Gym. Curves show the mean over ten seeds, where each seed is evaluated over ten test episodes. Shaded regions indicate $\pm 2\,\mathrm{SEM}$ across seed-level means.

### E.5 Architectural and Off-Policy Advances

Our experiments use simple MLP architectures and established baselines rather than the latest state-of-the-art agents, keeping the evaluation focused on the contribution of AA-PDA itself. Recent advances such as SimBa and SimBaV2 study scalable architectures and normalization [Lee et al., 2025a,b], TD7 augments TD3 with state-action representations and checkpointing [Fujimoto et al., 2023], and MR.Q develops a general-purpose model-free agent [Fujimoto et al., 2025]. Their architectural and off-policy advances may be complementary to AA-PDA. However, these methods are not evaluated in this study.

An off-policy version of AA-PDA would introduce additional challenges in maintaining the theoretical guarantees of AA-PDA. In PDA, each advantage function is defined with respect to the current policy, while the sum-advantage function accumulates information across all past policies. Introducing replay would mix data generated by different behavior policies, raising the question of how to preserve the interpretation of both the sum advantage function and the resulting cumulative objective.

Although replay could improve data reuse, it would also alter the assumptions underlying the current convergence analysis and require additional algorithmic modification.

### E.6 Additional Benchmark Results for MuJoCo-v5

Table 7: Performance comparison with on-policy baselines for MuJoCo-v5. Each cell shows the mean $\pm 2\,\mathrm{SEM}$ over ten seed-level means, after averaging the final five evaluations per seed. All environments are trained for 1M environment steps unless marked (3M).

| Environment | AA-PDA | PPO |
|---|---|---|
| HalfCheetah-v5 | **5174.6 ± 456.2** | 4067.5 ± 346.1 |
| Ant-v5 | **2625.1 ± 232.8** | 1115.4 ± 155.5 |
| Hopper-v5 | **2693.9 ± 534.3** | 2397.8 ± 116.6 |
| Walker2d-v5 | 3642.3 ± 844.2 | **3796.7 ± 543.7** |
| InvertedPendulum-v5 | 916.0 ± 135.4 | **975.8 ± 48.4** |
| InvertedDoublePendulum-v5 | **9349.2 ± 6.3** | 6782.6 ± 864.2 |
| Reacher-v5 | **-3.9 ± 0.1** | -11.1 ± 8.5 |
| Swimmer-v5 | **111.3 ± 19.5** | 54.2 ± 9.3 |
| Humanoid-v5 | **793.1 ± 87.8** | 671.1 ± 79.1 |
| Humanoid-v5 (3M) | **5180.1 ± 214.9** | 941.5 ± 182.6 |
| HumanoidStandup-v5 | **161199.0 ± 1893.1** | 130309.3 ± 17096.2 |

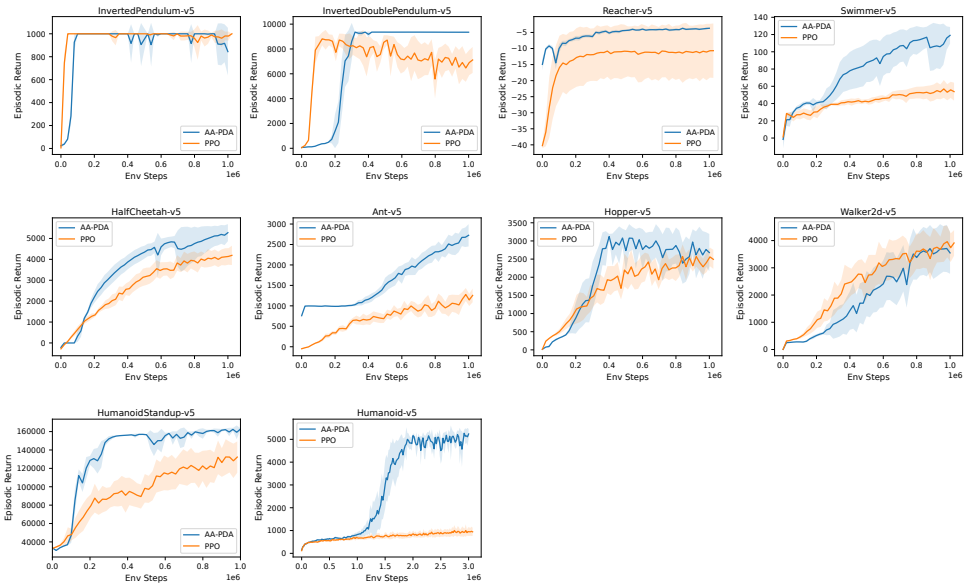



Figure 16: AA-PDA and PPO learning curves on MuJoCo-v5. Curves show the mean over ten seeds, where each seed is evaluated over ten test episodes. Shaded regions indicate $\pm 2\,\mathrm{SEM}$ across seed-level means.

### E.7 Aggregate Benchmark Summaries

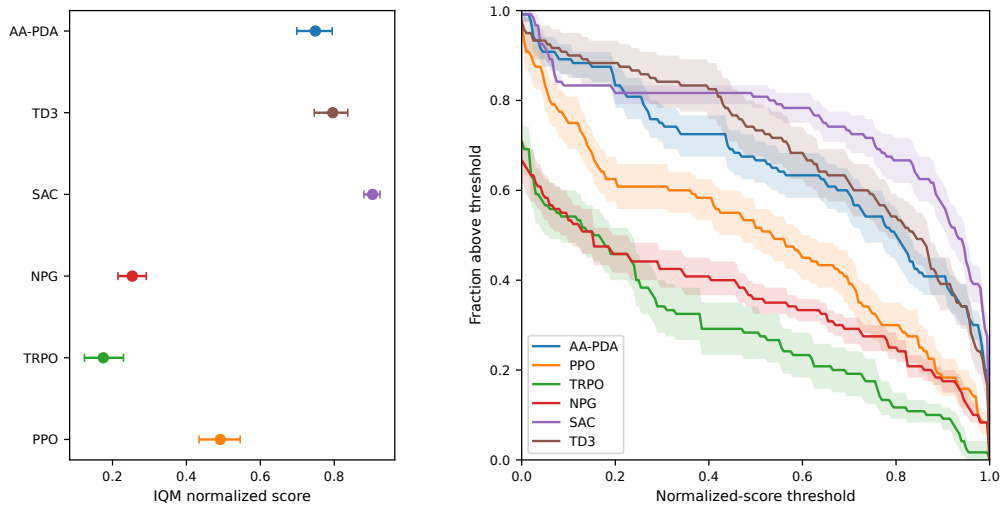



Figure 17: Aggregate final performance over 12 MuJoCo-v4 and Box2D environments with ten seeds per environment. Returns are normalized using frozen per-environment min–max references. Left: interquartile mean (IQM). Right: fraction of runs exceeding each normalized-score threshold. Bands and error bars are 95% stratified-bootstrap intervals from 5,000 replicates, resampling seeds within environments.

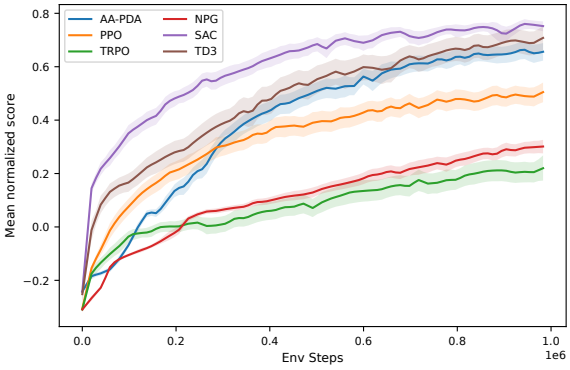



Figure 18: Aggregated normalized learning curve across the 12 continuous-control environments and ten seeds per environment. The center is the mean normalized score and the band is a 95% stratified-bootstrap interval obtained by resampling seeds within environments.

We additionally report an IQM plot (Figure 17) across 12 MuJoCo and Box2D environments using per-environment min–max normalization and stratified-bootstrap 95% confidence intervals. AA-PDA and TD3 have overlapping confidence intervals, whereas SAC performs better overall. Because IQM is less sensitive to extreme values, these results suggest that AA-PDA's overall performance is comparable to TD3 under this statistic. The runtime comparison in Table 4 shows that AA-PDA trains about 20–27 times faster than TD3. However, TD3 performs substantially better than AA-PDA in several individual environments, including Humanoid-v4, HalfCheetah-v4, and BipedalWalker-v3. In addition to the IQM results, Figure 18 reports the aggregate mean normalized score across all 12 environments.

## NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

**The checklist answers are an integral part of your paper submission.** They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our theoretical analysis and experimental results reflect the claims on runtime, approximation error, and performance in the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Section 8 discusses the additional sum advantage network, the uniform state-wise assumptions and diagnostics, and the on-policy design without replay. Section 7.4 compares with off-policy methods.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We stated the assumptions in Section 5. Further discussion about the assumptions is in Section 5 and Appendix A. Proof for lemmas and theorems is in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

Justification: The algorithm is described in Algorithm 3. Implementation and hyperparameters are stated in Appendix C and C.3.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code and a README are released at the GitHub repository linked in the abstract.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Implementation and hyperparameters are stated in Appendix C and C.3. Discussion on hyperparameter choices and optimizer is in Section 7.5 and Appendices C.4 and C.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Benchmark tables report the mean $\pm 2$ SEM over ten seed-level means, learning curves show $\pm 2$ SEM bands across seeds, and aggregate results use 95% stratified-bootstrap confidence intervals. The OR-method comparison reports standard deviations.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Computer resources are stated in Appendix E.1.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: We are committed to conforming with the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: The study focuses on an RL algorithm development. We do not foresee direct societal impacts beyond those of general-purpose RL methods.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our work does not require safeguards as we do not foresee the imminent risks. from using our algorithms.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cited and credited the open-source packages and data used in the paper. Licenses: Tianshou (MIT), Gymnasium (MIT), MuJoCo (Apache 2.0), Box2D (MIT), OR-Gym (MIT), and PyTorch (BSD-3-Clause).

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provided code in supplementary materials and a readme file for running.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer:[N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.