FINITE-TIME GENERALIZED ADVANTAGE ESTIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Advantage estimation is a central component of actor-critic reinforcement learning algorithms, such as A2C or PPO. One of the most widely used methods is generalized advantage estimation (GAE). However, its adaptation to practical finite-time rollout implementations is not covered by the infinite-horizon theory commonly used to derive GAE. This paper suggests to explicitly take into account the nature of finite-time rollouts to redefine the weighting of mixtures of k-step advantages. We analyze how standard truncated GAE changes the effective weighting of available estimators near rollout endpoints and propose *finite-time GAE (FT-GAE)*, which normalizes geometric weights over the available targets and retains a linear-time backward recursion. It is shown how our reweighting changes estimation error near rollout endpoints analytically under controlled bias-variance models and empirically in a controlled gridworld experiment. Extensive deep RL experiments across diverse benchmark families show substantial improvements when truncated GAE is replaced with FT-GAE. Controlled terminal-signal ablations provide further evidence that its advantage can be particularly pronounced when endpoint signals are informative.

1 INTRODUCTION

Reinforcement learning (RL) provides a framework for training agents to make sequential decisions through interaction with an environment (Sutton & Barto, 2018). In modern deep RL, policy-gradient methods are widely used, with applications ranging from game playing (Silver et al., 2016; OpenAI et al., 2019a) to robotics (Levine et al., 2016; OpenAI et al., 2019b) and policy optimization for language models (Ouyang et al., 2022; Christiano et al., 2017). Policy-gradient methods optimize a parametrized policy π_θ by estimating gradients of an expected return objective from data. While the policy-gradient theorem (Sutton et al., 1999; Williams, 1992) provides an unbiased gradient representation, direct Monte-Carlo estimators such as REINFORCE (Williams, 1992) often suffer from high variance (Greensmith et al., 2001). Actor-critic methods reduce this variance by using a learned value function as a baseline (Konda & Tsitsiklis, 1999; Mnih et al., 2016), which introduces a bias-variance trade-off. Advantage estimators need to be designed with this trade-off in mind. This is central in algorithms such as A2C (Mnih et al., 2016) and PPO (Schulman et al., 2017), where the advantage estimate directly determines the scale and direction of the policy update.

A standard way to control this bias-variance tradeoff is the number of look-ahead steps an estimator uses. With increasing horizon, an estimator relies less on bootstrapping which decreases sensitivity to critic error, but variance typically increases. Generalized advantage estimation (GAE) (Schulman et al., 2018), motivated by TD(λ) (Sutton & Barto, 2018), forms a geometrically weighted mixture of k-step advantage estimators. This simple mechanism has become the default choice in many PPO-style implementations, including Stable-Baselines3 (Raffin et al., 2021). While GAE is naturally motivated as an infinite-horizon geometric mixture, in practice the estimators are computed from finite rollout buffers. In episodic tasks, this endpoint may also coincide with an environment termination.

This finite-rollout detail has an important consequence that is easy to overlook. When the usual GAE recursion starts at a rollout endpoint, the geometric mass which would be assigned to later time steps is not simply removed. Instead, when truncated GAE is written as a mixture over the multi-step estimators actually observable in the buffer, all remaining tail mass is assigned to the longest available estimator. We refer to this as *tail-mass collapse*. As a result, advantages near rollout endpoints can become more long-horizon oriented and Monte-Carlo-like than the nominal value of λ suggests. The

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

observation itself is not new, tail-mass collapse is known for λ-returns, see Section 7.2 of Sutton & Barto (2018). The novelty of the present work is to study previously overlooked consequences and propose a simple way to modify the estimator to answer the following question:

How do finite rollout lengths affect the weighting of GAE advantage estimators and the learning of PPO, and when can a simple finite-time reweighting improve the learning?

We answer this question in three steps. First, we show that truncated GAE assigns the weight of unavailable horizons to the longest observable estimator and introduce *finite-time GAE* (FT-GAE), which normalizes geometric weights over the available horizons. FT-GAE retains a linear-time backward recursion, changing only the scalar coefficient on the recursive term. Second, we investigate a candidate regime in which tail-mass collapse can be costly through a controlled GridWorld experiment and a suitable bias–variance analysis. Third, we evaluate FT-GAE with PPO across a broad set of environments, using λ controls and endpoint interventions.

2 PRELIMINARIES

We consider a Markovian environment described by $(\mathcal{S}, \mathcal{A}, R, p)$ with initial distribution μ and time-stationary policy π. This induces a process $(S_t, A_t, R_t)_{0 \le t < T}$, where $T \in \mathbb{N} \cup \{\infty\}$. We allow either $T < \infty$ with $\gamma \in [0, 1]$, or $T = \infty$ with $\gamma \in [0, 1)$. For a policy π, define the value function $V_t^\pi(s) := \mathbb{E}^\pi[\sum_{i=t}^{T-1} \gamma^{i-t} R_i | S_t = s]$, state-action value function $Q_t^\pi(s, a) := \mathbb{E}^\pi[\sum_{i=t}^{T-1} \gamma^{i-t} R_i | S_t = s, A_t = a]$ and advantage function $\mathbb{A}_t^\pi(s, a) := Q_t^\pi(s, a) - V_t^\pi(s)$. For notational simplicity, and to stay close to PPO practice, we pretend the critic V is stationary. For a parameterized policy π_θ, the objective is $J(\theta) := V_0^{\pi_\theta}(\mu) := \int_{\mathcal{S}} V_0^{\pi_\theta}(s)\, d\mu(s)$. The policy-gradient theorem in the advantage form is $\nabla_\theta J(\theta) = \sum_{t=0}^{T-1} \gamma^t \mathbb{E}^{\pi_\theta}[\nabla_\theta \log \pi_\theta(A_t \mid S_t)\, \mathbb{A}_t^{\pi_\theta}(S_t, A_t)]$. Actor-critic methods approximate this expression by replacing the advantage with an estimator $\hat{\mathbb{A}}_t$ constructed from rollout data and a learned value function.

Let $V : \mathcal{S} \to \mathbb{R}$ be a value function approximation. The one-step temporal-difference error is $\delta_t := R_t + \gamma V(S_{t+1}) - V(S_t)$. If $V = V^\pi$, then δ_t is conditionally unbiased for the advantage. With an approximate critic, this estimator is biased through the bootstrap term $V(S_{t+1})$. More generally, for $k \ge 0$, define the k-step return and the corresponding advantage estimator $\hat{G}_t^{(k)} := \sum_{\ell=0}^{k} \gamma^\ell R_{t+\ell} + \gamma^{k+1} V(S_{t+k+1})$ and $\hat{\mathbb{A}}_t^{(k)} := \hat{G}_t^{(k)} - V(S_t)$. When $V = V^\pi$, each $\hat{\mathbb{A}}_t^{(k)}$ is conditionally unbiased for $\mathbb{A}^\pi(S_t, A_t)$. The standard bias-variance trade-off is controlled by k: With increasing k, the reliance on bootstrapping and thus the critic-induced bias is reduced, whereas the higher number of sampled rewards leads to higher variance. As $k \to \infty$, the Monte Carlo advantage estimator without bootstrapping is recovered. For $\lambda \in (0, 1)$, the infinite-horizon λ-return and generalized advantage estimators (Schulman et al., 2018) are the mixtures $\hat{G}_t^{\infty,\lambda} := (1-\lambda) \sum_{k=0}^{\infty} \lambda^k \hat{G}_t^{(k)}$ and $\hat{\mathbb{A}}_t^{\infty,\lambda} := (1-\lambda) \sum_{k=0}^{\infty} \lambda^k \hat{\mathbb{A}}_t^{(k)}$. The weight normalization is chosen such that $(1-\lambda) \sum_{k=0}^{\infty} \lambda^k = 1$. This estimator admits a TD-error representation: Using the identity $\hat{\mathbb{A}}_t^{(k)} = \sum_{\ell=0}^{k} \gamma^\ell \delta_{t+\ell}$ yields

$$\hat{\mathbb{A}}_t^{\infty,\lambda} = (1-\lambda) \sum_{k=0}^{\infty} \lambda^k \sum_{\ell=0}^{k} \gamma^\ell \delta_{t+\ell} = (1-\lambda) \sum_{\ell=0}^{\infty} \gamma^\ell \delta_{t+\ell} \sum_{k=\ell}^{\infty} \lambda^k = \sum_{\ell=0}^{\infty} (\gamma\lambda)^\ell \delta_{t+\ell}.$$

Thus, future TD errors are weighted geometrically by the product $\gamma\lambda$. Smaller λ yields more bootstrapping and typically lower variance but greater sensitivity to critic bias, and vice versa.

The preceding definitions are infinite-horizon objects: they require $\delta_{t+\ell}$ to be defined for all future times. For finite rollouts, PPO instead uses truncated GAE (Schulman et al., 2017). If the rollout ends at τ, then

$$\hat{\mathbb{A}}_t^{\mathrm{trunc},\lambda} := \sum_{\ell=0}^{\tau-t-1} (\gamma\lambda)^\ell\, \delta_{t+\ell}, \qquad t \le \tau - 1.$$

This estimator can be computed backward as

$$\hat{\mathbb{A}}_t^{\mathrm{trunc},\lambda} = \delta_t + \gamma\lambda \hat{\mathbb{A}}_{t+1}^{\mathrm{trunc},\lambda}, \qquad \hat{\mathbb{A}}_\tau := 0.$$

However, the original interpretation of GAE as a geometric mixture of k-step estimators with weights $(1-\lambda)\lambda^k$ no longer holds unchanged. Instead, all geometric mass corresponding to unavailable k-step estimators beyond the rollout end is assigned to the longest available estimator.

Proposition 2.1 (Tail-mass collapse of truncated GAE). *For $t \leq \tau - 1$, truncated GAE satisfies*

$$\hat{\mathbb{A}}_t^{\mathrm{trunc},\lambda} = \sum_{k=0}^{\tau-t-2} \underbrace{(1-\lambda)\lambda^k}_{\text{GAE weights}} \hat{\mathbb{A}}_t^{(k)} + \underbrace{\lambda^{\tau-t-1}}_{\text{collapsed tail-mass}} \hat{\mathbb{A}}_t^{(\tau-t-1)}. \tag{1}$$

We call the effect of Proposition 2.1 *tail-mass collapse*; see the blue bars in Figure 1. In a finite rollout, estimators beyond τ are unavailable, so truncated GAE places this entire tail mass on the longest available estimator $\hat{\mathbb{A}}_t^{(\tau-t-1)}$. Consequently, advantage estimators close to the rollout end become substantially more Monte-Carlo-like than the nominal value of λ suggests. Hence, the amount of collapsed mass depends on the remaining rollout length $\tau - t$, so the effective bias-variance trade-off near the endpoint varies locally through the distance to τ, not only by the chosen value of λ. In actor-critic implementations, the endpoint τ is not only determined by the finite rollout buffer, and may correspond either to a true termination or to an artificial rollout endpoint.

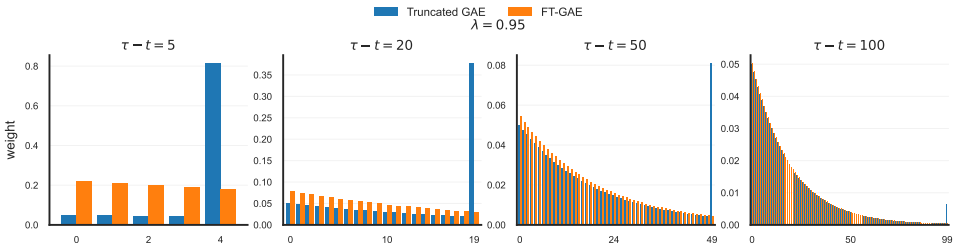


Figure 1: **GAE tail-mass collapse.** Comparison of the weights assigned to the k-step advantage estimators $\hat{\mathbb{A}}_t^{(k)}$ by truncated GAE (blue) and our FT-GAE introduced in Definition 3.1 (orange) for $\lambda = 0.95$. Each panel corresponds to a different remaining horizon $\tau - t$ and shows how, for truncated GAE, the tail mass collapses onto the last available estimator $\hat{\mathbb{A}}_t^{(\tau-t-1)}$, while the finite-time variant redistributes this mass over the observable k-step estimators via finite-horizon renormalization.

3 Mitigating Tail-Mass Collapse: Finite-Time GAE (FT-GAE)

Why can tail-mass collapse be problematic in RL? Solving (discrete) optimal control problems is closely related to dynamic programming and as such should profit from bootstrapping future information. This is inherent in value-based methods such as Q-learning but less so in policy gradient methods (high λ values). As an example it is surprising how little PPO training of Lunar Lander profits from the strong terminal signal at crashing/landing. A reason we identified is that tail-mass collapse leads to excessive Monte Carlo close to termination instead of bootstrapping already learned information. As termination can mean crashing (-100) or landing (+100), this significantly increases variance of the corresponding truncated GAE estimators. More generally, tail-mass collapse can be costly when a terminal reward is salient relative to within-episode rewards and its realized outcome varies across trajectories. If nearby value estimates already capture its expected effect, shorter bootstrapped targets retain the expected signal while decreasing variance.

Require: Policy π_θ, value estimate V, discount γ, GAE parameter λ
1: Generate rollout $\{(s_t, a_t, r_t, v_t)\}_{t=0}^{\tau-1}$ with $v_t \leftarrow V(s_t)$ for $t = 0, \ldots, \tau$
2: $\hat{\mathbb{A}}_\tau^{\mathrm{FT}} \leftarrow 0$
3: **for** $t = \tau - 1, \ldots, 0$ **do**
4: $\quad \delta_t \leftarrow r_t + \gamma v_{t+1} - v_t$
5: $\quad \hat{\mathbb{A}}_t^{\mathrm{FT}} \leftarrow \delta_t + \gamma\lambda \dfrac{1-\lambda^{\tau-t-1}}{1-\lambda^{\tau-t}} \hat{\mathbb{A}}_{t+1}^{\mathrm{FT}}$
6: **end for**
7: **return** $\{\hat{\mathbb{A}}_t^{\mathrm{FT}}\}_{t=0}^{\tau-1}$

Algorithm 1: Finite-time GAE (ours)

We propose a simple modification, *finite-time GAE*, that removes tail-mass collapse by dividing the geometric weights of the available k-step estimators at each rollout position by their finite sum.

Definition 3.1 (Finite-time GAE). For $\lambda \neq 1$, we define the finite-time generalized advantage estimator (FT-GAE) by

$$\hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} := \frac{1-\lambda}{1-\lambda^{\tau-t}} \sum_{k=0}^{\tau-t-1} \lambda^k \hat{\mathbb{A}}_t^{(k)}, \qquad t < \tau. \tag{2}$$

For $\lambda = 1$, this mixture is understood as its continuous extension, yielding the uniform average.

Finite-time GAE preserves the geometric preference for shorter look-aheads, but normalizes this preference over the finite set $k \in \{0, \dots, \tau - t - 1\}$. Thus, both truncated GAE and FT-GAE are position-dependent on finite rollouts, but in different ways: truncated GAE collapses the unavailable tail mass onto the longest available estimator, whereas FT-GAE redistributes this mass over all available estimators according to the normalized finite geometric law. The orange bars in Figure 1 show these weights. Consequently, FT-GAE differs appreciably from truncated GAE only close to rollout endpoints, where the missing tail mass is largest.

Finite-time normalization also changes the limiting behavior as $\lambda \uparrow 1$. For truncated GAE, the limiting mixture puts all mass on the longest available estimator $\hat{\mathbb{A}}_t^{(\tau-t-1)}$. In contrast, FT-GAE remains a normalized finite mixture: at $\lambda = 1$, it is the uniform average over all available k-step estimators, not the Monte Carlo estimator. Although FT-GAE can be extended to $\lambda > 1$, assigning increasing weight to larger k and recovering a Monte Carlo-type limit as $\lambda \to \infty$, we focus on the practically relevant regime $\lambda \leq 1$. Appendix Figure 11 compares the resulting weighting schemes.

FT-GAE also admits a linear-time backward pass, formalized in the following proposition:

Proposition 3.2. *For $\lambda \neq 1$, FT-GAE admits the TD-sum representation*

$$\hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} = \sum_{\ell=0}^{\tau-t-1} (\gamma\lambda)^\ell \, \frac{1-\lambda^{\tau-t-\ell}}{1-\lambda^{\tau-t}} \, \delta_{t+\ell}$$

for $t < \tau$, and satisfies the backward recursion

$$\hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} = \delta_t + \gamma\lambda \, \frac{1-\lambda^{\tau-t-1}}{1-\lambda^{\tau-t}} \, \hat{\mathbb{A}}_{t+1}^{\mathrm{FT},\lambda}, \qquad \hat{\mathbb{A}}_\tau^{\mathrm{FT},\lambda} := 0.$$

Relative to truncated GAE, only the coefficient of the recursive term changes in Algorithm 1: the fixed factor $\gamma\lambda$ becomes the horizon-dependent factor in Proposition 3.2. Thus, existing PPO implementations such as Stable-Baselines3 (Raffin et al., 2021) require only a local modification of the GAE recursion. Implementation details can be found in Appendix E.1.

4 Controlled Insights

Figure 1 and Proposition 3.2 suggest that FT-GAE has pronounced effects near the end of rollouts. To investigate this intuition, we study a simple toy model where exact advantages are available and provide a simple mathematical analysis before moving on to comprehensive experimental evaluation.

4.1 A time-inhomogeneous gridworld

We use a time-inhomogeneous GridWorld (adding time to the state-space makes it a standard MDP) which is hard enough to challenge PPO but simple enough to perform explicit computations of V^π, Q^π, and $\mathbb{A}^\pi$. The agent (blue), target (green), and distractor (red) are placed at new locations at each reset. Collecting the target yields a positive reward that decreases from 2 to 0.2 over the maximal horizon. Collecting the distractor yields -1. Both pickups terminate the episode, and all other rewards are zero.

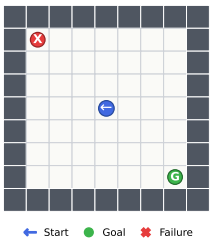


Figure 2: Example GridWorld layout.

We train PPO with truncated GAE and FT-GAE for one million transitions over ten seeds. PPO with FT-GAE (orange) reaches higher evaluation returns throughout training compared with PPO with standard truncated GAE (Figure 3a). The Critic of FT-GAE has lower value RMSE (Figure 3b). To understand where these differences come from we apply both estimators to the same transitions and compare their advantage estimates with the true A^π and their return targets with the true Q^π. Panel (c) plots the corresponding RMSE against the number of steps remaining to the segment endpoint. FT-GAE has substantially lower error near the endpoint. The gap narrows as the distance to rollout end grows. This is consistent with the proposed mechanism of Figure 1. FT-GAE changes the estimator most strongly when few k-step targets remain and progressively approaches truncated GAE farther from the boundary. Return-target RMSE shows a similar picture (Panel (d)). We also compare each estimated PPO-surrogate gradient with the gradient obtained from exact advantages using cosine similarity. The advantage differences are reflected in the policy-gradient diagnostic of Panel (e). More comprehensive results on this toy model are gathered in Appendix B.

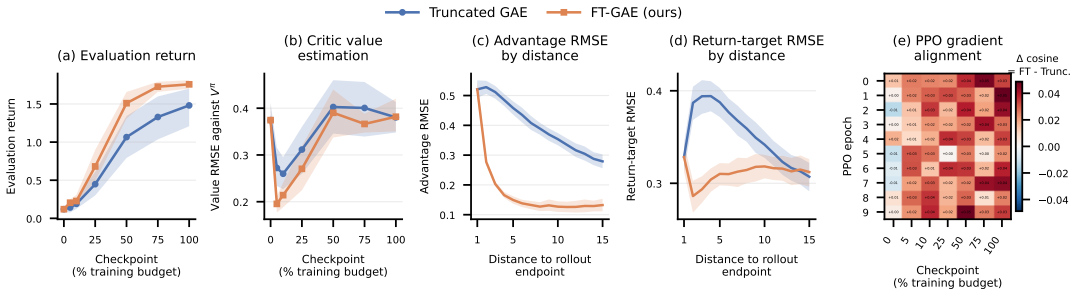


Figure 3: Controlled Fetch-GridWorld results. (a) Evaluation return and (b) critic RMSE relative to exact V^π for independently trained PPO agents. (c,d) Advantage RMSE relative to exact A^π and return-target RMSE relative to exact Q^π, grouped by remaining endpoint distance. (e) Differences in cosine similarity to the PPO-surrogate gradient computed with exact advantages; positive values favor FT-GAE. Shaded bands indicate pointwise 95% bootstrap confidence intervals.

4.2 Mathematical Insights

To investigate a mathematical mechanism for the local boundary behavior, we consider a simple bias-variance model that is supposed to capture the situation of environments with informative endpoint rewards. The model is not intended as a universal description of critic error in arbitrary PPO runs. Rather, it isolates the terminal-reward regime suggested by the controlled GridWorld experiment. Full details are given in Appendix C.

Fix a rollout endpoint $\tau = T$. Since adding $V(S_t)$ to an advantage estimator gives the corresponding λ-return, and since subtracting a baseline does not change the expected policy gradient, we analyze return targets directly. For mixture weights $c = (c_0, \dots, c_{T-t-1})$, define the return target $\hat{G}_t^c = V(S_t) + \sum_{k=0}^{T-t-1} c_k \hat{\mathbb{A}}_t^{(k)}$. Our variance model follows the TD-error decomposition of compound returns (Daley et al., 2024), but replaces the equicorrelation model by a decaying covariance structure.

Assumption 4.1. We assume $e_t := \mathbb{E}[V(S_t) - V^\pi(S_t)] = \varepsilon_{\min} + (\varepsilon_{\max} - \varepsilon_{\min}) \left(\frac{T-t}{T}\right)^2$ and $\mathrm{Cov}[\delta_{t+i}, \delta_{t+j} \mid S_t] = \kappa \rho^{|i-j|}$, for $i, j = 0, \dots, T-t-1$, with $\kappa > 0$ and $\rho \in [0, 1]$.

We use the convention $0^0 = 1$. Thus, in our model, the critic bias is largest far from the rollout endpoint ($e_0 = \varepsilon_{\max}$) and decreases toward the endpoint ($e_T = \varepsilon_{\min}$). The quadratic form is chosen only for simplicity; the relevant property is the monotone decrease of critic bias toward the endpoint. Since the k-step target at time t bootstraps at S_{t+k+1}, its bias is $\gamma^{k+1} e_{t+k+1}$, and therefore $\mathrm{Bias}[\hat{G}_t^c] = \sum_{k=0}^{T-t-1} c_k \gamma^{k+1} e_{t+k+1}$. Analyzing the TD-error representation gives the following compound variance model.

Proposition 4.2. *Let $h_i := \sum_{n=i}^{T-t-1} c_n$ denote the cumulative weights induced by the compound estimator. Then, under Assumption 4.1:* $\mathrm{Var}[\hat{G}_t^c \mid S_t] = \kappa \sum_{i=0}^{T-t-1} \sum_{j=0}^{T-t-1} \gamma^{i+j} h_i h_j \rho^{|i-j|}$.

For truncated λ-returns, $h_i^{\mathrm{trunc}} = \lambda^i$, whereas for FT-$\lambda$-returns, $h_i^{\mathrm{FT}} = \lambda^i \frac{1-\lambda^{T-t-i}}{1-\lambda^{T-t}} \le h_i^{\mathrm{trunc}}$ for $0 \le \lambda < 1$; and the continuous limit at $\lambda = 1$ (Appendix C). Their difference is concentrated near segment boundaries, but vanishes at $T - t = 1$. Under Assumption 4.1, FT-GAE reduces variance at the cost of weakly higher squared bias. For both variants, we compute $\mathrm{MSE}_t(\lambda) = \mathrm{Bias}_t^2(\lambda) + \mathrm{Var}_t(\lambda)$ at each rollout position. In the illustrated regime the variance reduction outweighs the bias cost after averaging uniformly over rollout positions t, yielding lower return-target MSE across a range of λ (Figure 4). This offers a mathematical explanation for the GridWorld's near-boundary improvement.

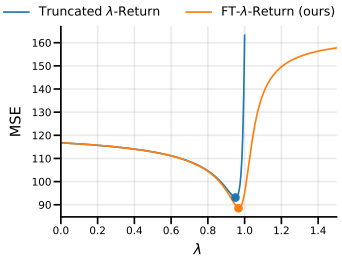


Figure 4: Toy-model return-target MSE, averaged over rollout positions.

5 EXPERIMENTS

We next provide extensive experiments and ablation runs on 24 environments from 5 suites, to highlight the advantages of FT-GAE over the standard truncated GAE estimation.

5.1 PPO RELEVANCE ACROSS ENVIRONMENTS: TRUNCATED GAE VS. FT-GAE

Experimental setup: We evaluate on a diverse set of 24 environments spanning two Box2D tasks, the Atari-5 benchmark (Aitchison et al., 2023), six MiniGrid tasks (Chevalier-Boisvert et al., 2023), six MuJoCo tasks (Todorov et al., 2012), and five MinAtar environments (Young & Tian, 2019). The first four families use the Gymnasium API (Towers et al., 2024) and the Stable-Baselines-Jax (SBX) PPO implementation (Raffin et al., 2021) through the RL Baselines3 Zoo. For MinAtar experiments we use Gymnax (Lange, 2026) and a Jaxnasium-based PPO implementation (Ponse, 2025). Within each environment, the PPO algorithm, networks, optimizer, and hyperparameters are held fixed. The *only change* is the factor $\frac{1-\lambda^{\tau-t-1}}{1-\lambda^{\tau-t}}$ in the backward GAE recursion (Algorithm 1) to isolate the effect of replacing truncated GAE by our proposed FT-GAE. Hyperparameters, probability-of-improvement plots, interquartile means and per-environment results are provided in Appendix E.

Learning curves, Figure 5: Replacing truncated GAE (blue curve) by our FT-GAE (orange curve) improves learning on several tasks without environment-specific retuning. Gains are most pronounced on MinAtar, Box2D and MiniGrid, where FT-GAE improves learning speed and final performance. Box2D and MiniGrid are the benchmark families where terminal information is particularly informative. The curves illustrate the practical effect; the aggregate analysis below assesses its breadth.

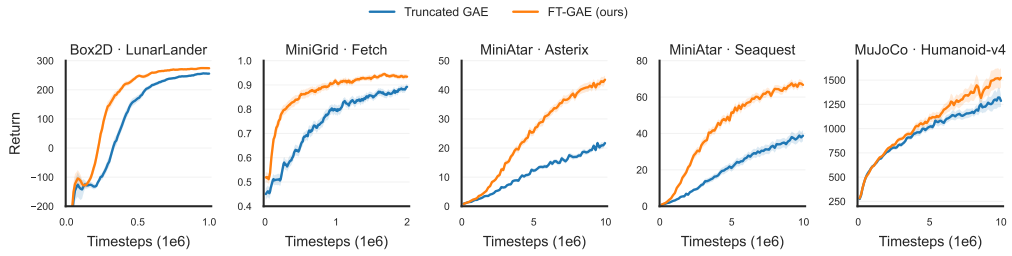


Figure 5: PPO learning curves under default λ-values. Results are averaged over 30 seeds. Shaded regions denote standard errors.

Robust aggregate results, Figure 6: We summarize each run by area under the evaluation-return curve (AUC), which captures sample efficiency, and final return, averaged over the last 10% of evaluation checkpoints. We aggregate results across tasks and seeds using the RLiable methodology (Agarwal et al., 2021). The performance profiles in Figure 6 plot normalized score thresholds on the x-axis against the fraction of runs exceeding each threshold, averaged across tasks, on the y-axis. At a given threshold, a higher curve means that more runs exceed it. Conversely, reading the score where a curve reaches a given y-value reveals a percentile (y = 0.5 marks the median score, while y = 0.25 marks the 75th percentile). For both AUC and final return, FT-GAE's orange curves lie markedly above the blue curves of truncated GAE over a broad range of thresholds. FT-GAE therefore exceeds the same performance targets in a substantially larger share of runs. Accordingly, FT-GAE achieves an overall probability of improvement of 0.69 for AUC and 0.67 for final return.

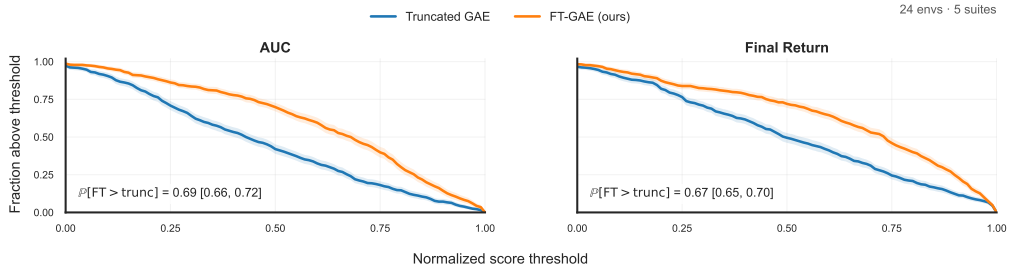


Figure 6: PPO performance across **Atari-5, MuJoCo, MinAtar, Box2D, and MiniGrid**. The robust aggregation shows markedly higher performance for FT-GAE (orange) than for truncated GAE (blue). Probability of improvement in final return is 0.67. Bands show 95% bootstrap confidence intervals.

5.2 λ-SENSITIVITY ABLATIONS

It is reasonable to ask how much of FT-GAE gains could be recovered by different choices of λ. First, note that truncated GAE with a smaller λ does not shift the bias-variance tradeoff in the same way as FT-GAE. While lowering λ in truncated GAE increases bootstrapping throughout a rollout, FT-GAE instead normalizes target weights according to the remaining rollout length, with its largest effect near endpoints (Figure 1). We test whether a global λ can match the gains reported in Section 5.1. We sweep λ separately for each estimator and environment, holding all other PPO hyperparameters fixed. The main grid spans $\{0.75, 0.775, \ldots, 1.0\}$ and extends to 0.65 for selected MuJoCo tasks, extending well beyond the default λ choices of the PPO configurations. Figure 7 shows two illustrative tasks, LunarLander-v3 and MiniGrid-Fetch-5x5-N2-v0. Across the tested values, lowering λ for truncated GAE does not close the observed gap to FT-GAE in either final performance or AUC. The full sensitivity curves are reported in Appendix E.6.

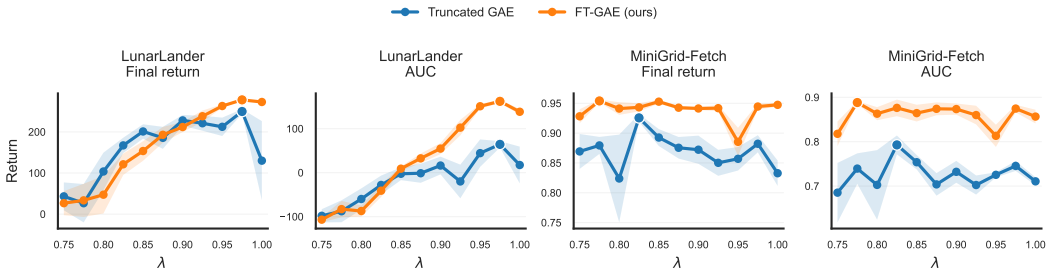


Figure 7: Sensitivity to λ in LunarLander-v3 and MiniGrid-Fetch-5x5-N2-v0. We compare truncated GAE and FT-GAE final performance and AUC over five seeds. Within the tested range, retuning truncated GAE does not close the displayed performance gaps.

For each estimator and environment, in Table 1 we select λ by final return on five sweep seeds and evaluate the selected setting on 30 new seeds. AUC is measured at that same λ. Again, we normalize and aggregate results. Again, the results significantly favor FT-GAE in MinAtar, Box2D and MiniGrid, while MuJoCo shows a positive trend with little separation. Thus, λ retuning does not explain the differences of Section 5.1.

Table 1: Best-over-λ diagnostic for PPO. All hyperparameters except λ are kept fixed. For each environment, we select the λ value with the highest final return and report final return and AUC at that selected value for 30 new seeds. We report IQM with 95% stratified bootstrap confidence intervals and probability of improvement of FT-GAE over truncated GAE. Atari-5 was run with default λ.

Class	Metric	Truncated GAE	FT-GAE	$\mathbb{P}[\mathrm{FT} > \mathrm{trunc}]$
Box2D	Final return	0.79 [0.73, 0.82]	**0.91 [0.90, 0.92]**	**0.78 [0.70, 0.86]**
Box2D	AUC	0.52 [0.47, 0.56]	**0.78 [0.74, 0.81]**	**0.88 [0.82, 0.93]**
MinAtar	Final return	0.61 [0.57, 0.65]	**0.76 [0.72, 0.79]**	**0.69 [0.63, 0.74]**
MinAtar	AUC	0.63 [0.58, 0.67]	**0.68 [0.64, 0.73]**	**0.59 [0.53, 0.64]**
MiniGrid	Final return	0.76 [0.73, 0.79]	**0.82 [0.78, 0.85]**	**0.60 [0.55, 0.65]**
MiniGrid	AUC	0.65 [0.61, 0.69]	**0.75 [0.71, 0.79]**	**0.62 [0.56, 0.67]**
MuJoCo	Final return	0.47 [0.41, 0.53]	0.54 [0.48, 0.59]	0.54 [0.47, 0.60]
MuJoCo	AUC	0.49 [0.44, 0.53]	0.51 [0.47, 0.56]	0.51 [0.46, 0.57]

5.3 POLICY GRADIENT OPTIMIZER ABLATION

To examine whether the observed gains depend on interactions with PPO-specific mechanisms such as clipping, we also replace truncated GAE with FT-GAE in A2C. We deliberately focus on the two Box2D tasks with salient terminal signals where finite-time weighting is expected to matter. Across 20 seeds per setting, FT-GAE achieves higher mean AUC and mean final return in all six tested configurations for LunarLander-v3 and BipedalWalker-v3. The full comparison is reported in Table 2 of the appendix. At the default λ values, LunarLander's mean AUC rises from -63.85 to 16.75 and its mean final return from -33.73 to 54.89. BipedalWalker's mean AUC rises from 182.52 to 208.60 and its mean final return from 242.26 to 278.86.

5.4 Endpoint importance ablation

To illustrate the advantage of FT-GAE under strong endpoint signals, we modified a setting in which truncated GAE was initially stronger. We chose Hopper-v4, where standard truncated GAE outperformed FT-GAE in the default-λ experiments in Section 5.1. In a modified Hopper-v4 environment we add a penalty of $-C$ to the reward at early termination (i.e. for falling down), with $C \in \{0, 5, 100\}$. We evaluate every policy on the original, unmodified reward to keep scores comparable across training conditions. Each comparison uses 40 seeds. Figure 8 shows that FT-GAE performs worse than truncated GAE at $C = 0$, and better with positive and increasing C. Final return follows the same qualitative pattern. Appendix E.2.2 reports further Hopper comparisons (different λ, effect of reward normalization). The penalty-dependent widening persists in both. It also reports an analogous MiniGrid-Fetch intervention, which varies rewards and penalties and finds larger FT-GAE gaps under stronger penalties.

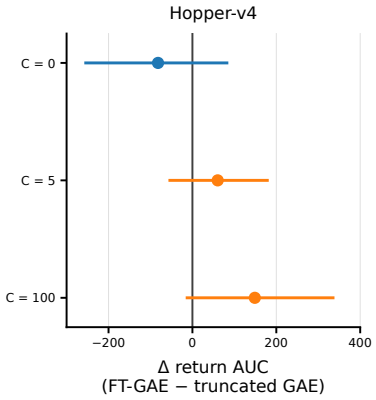


Figure 8: Hopper AUC difference. Points show means; bars show 95% bootstrap intervals.

5.5 Episode terminations vs. rollout cutoffs ablation

True episode terminations and artificial rollout cutoffs differ: a termination ends the episode with $V(s_\tau) = 0$, whereas rollout cutoffs or truncations bootstrap into a continuing episode. Our standard FT-GAE implementation applies finite-time weighting at both, so we ask whether its gain persists when FT-GAE is restricted only to true terminations. To test whether the LunarLander-v3 gain depends on the treatment of rollout cutoffs, we apply FT-GAE only at true terminations and truncated GAE at rollout cutoffs. Across 30 seeds, this hybrid (dashed green line) closely tracks full FT-GAE; both achieve higher mean evaluation returns than truncated GAE (Figure 9). This suggests that FT-GAE's treatment of artificial rollout cutoffs is not the sole source of the gain, but also not a problem.

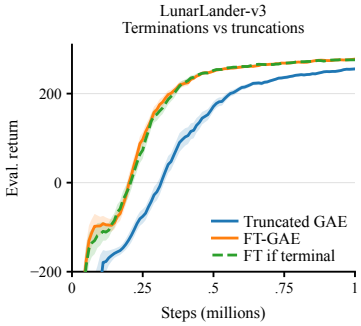


Figure 9: Comparing GAE, FT-GAE, and hybrid FT-GAE (green).

5.6 Actor and/or critic ablation

PPO uses GAE to compute actor advantages and the associated λ-returns as critic targets. To examine the two uses separately, we conduct a 2×2 ablation that combines either truncated GAE or FT-GAE for the actor's advantages with either estimator for the critic's return targets. All four variants use the same default training setup as in Section 5.1. Results average over 30 seeds. In Figure 10, the solid blue curve uses truncated variant for both quantities, while the solid orange curve uses FT variant for both. The green dashed curve uses FT-GAE only for actor advantages and standard λ-returns for critic targets; the purple dash-dotted curve uses truncated GAE for actor advantages and FT-λ-returns for critic targets. Both mixed variants generally lie between the two solid curves. This pattern suggests that using FT-GAE for actor advantages and critic targets contributes to the gain, while neither alone reproduces the full FT-GAE result.

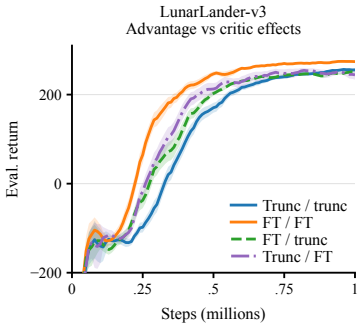


Figure 10: Mixed variants of actor advantages and critic targets.

6 Related work

Advantage estimation is a central variance-reduction mechanism in and actor-critic methods (Mnih et al., 2016; Pignatelli et al., 2024; Schulman et al., 2017; Sutton & Barto, 2018). Prior work has improved policy-gradient estimators through richer baselines and control variates, including off-policy critics (Gu et al., 2016), action-dependent baselines from Stein identities (Liu et al., 2018), and factorized baselines for high-dimensional actions (Wu et al., 2018). Other work has shown that the practical gains of action-dependent baselines can be benchmark-dependent (Tucker et al., 2018). FT-GAE is complementary: it keeps the critic class and changes only the rollout weighting of GAE.

GAE, a standard choice in PPO, builds on the TD(λ) forward view by geometrically mixing multi-step advantage estimators (Schulman et al., 2017; 2018; Sutton & Barto, 2018). Related alternatives modify advantage estimation by learning advantages directly (Pan et al., 2022), using data augmentation for bootstrap advantage targets (Rahman & Xue, 2022), or adapting λ during training (Chen et al., 2021). In contrast, FT-GAE keeps the fixed-λ setting used in standard PPO implementations, e.g. Stable-Baselines3 (Raffin et al., 2021), and revisits finite-rollout normalization of the mixture.

In classical finite-horizon λ-returns, assigning geometric tail mass to the terminal target is a standard convention (Sutton & Barto, 2018). In actor-critic training, the same convention is also an estimator-design choice: near artificial truncations and true episode endings, the tail-mass collapse can make endpoint advantage estimates more Monte-Carlo-like than the value of λ suggests. Truncated GAE has recently appeared in various settings, including language-model RL (Fan et al., 2025) and classical environments (Song et al., 2023). FT-GAE instead renormalizes over observable horizons. Our MSE analysis in Section 4.2 builds on compound-return variance analyses (Daley et al., 2024; Konidaris et al., 2011), extending them to rollout-local critic bias and decaying TD-error correlations.

7 Limitations

FT-GAE is a simple modification of truncated GAE targeting specific settings. Our evaluation compares FT-GAE only with truncated GAE, the standard in many implementations. Advantages over other estimators remain untested. FT-GAE does not offer uniform improvements across all environments, although our evaluation across 24 environments yields a probability of improvement of 0.69 for AUC. Instead, our analysis and experiments identify a specific favorable regime: environments with salient endpoint signals. When endpoint signals are weak or the geometric tail mass is small, the two estimators should behave similarly (and our experiments do not observe significant changes in these settings). This is a comparatively precise characterization for an applied RL modification. Furthermore, truncated GAE (or general MC-like estimates) can still be preferable when critic-induced bootstrap bias outweighs the additional variance of longer-horizon returns. FT-GAE shifts weight from the longest available return toward shorter bootstrapped estimators, thus, similarly to decreasing λ, the variance reduction naturally comes at the cost of greater reliance on the critic.

8 Conclusion and future work

When written as a mixture over observable k-step estimators, truncated GAE assigns unavailable geometric tail mass to the longest available estimator, making estimates near rollout endpoints more Monte-Carlo-like than intended. We propose FT-GAE, which instead uses a normalized finite geometric mixture over available horizons while retaining the linear-time recursion of standard GAE. Empirically, FT-GAE yields substantial PPO gains on tasks with strong terminal signals, such as Box2D and MiniGrid, as well as on MinAtar, while producing no change on MuJoCo. These results suggest FT-GAE as a practical replacement for truncated GAE in PPO implementations. Future work should further clarify when finite-time normalization is beneficial, including the roles of critic accuracy, terminal rewards, rollout length, and episode-length distribution. Another direction is adaptive finite-time weighting, where the k-step mixture is adjusted over training or by estimates of critic bias or TD-error variance. This could interpolate between truncated GAE and FT-GAE, favoring truncated GAE when the critic is unreliable and shifting toward FT-GAE as local value estimates improve. Future work could investigate whether the finite-horizon weighting of FT-GAE also benefits other algorithms that use λ-returns or advantage estimates, such as PQN (Gallici et al., 2025).

AI USE STATEMENT

We used LLMs to support literature discovery, discuss research ideas and aspects of their execution, draft portions of the manuscript, and improve grammar, clarity, and style. The authors critically reviewed all AI-assisted content and made the final research and writing decisions. All core intellectual contributions, including the proposed methods, theoretical framework, experimental design, and analysis, were conceived and carried out by the human authors, who take full responsibility for the paper.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, our implementation builds on the Stable-Baselines3 Zoo (Raffin et al., 2021) and Jaxnasium (Ponse, 2025) codebases. Appendix E describes our implementation, experimental setup, configurations and hyperparameters.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 29304–29320. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/f514cec81cb148559cf475e7426eed5e-Paper.pdf.

Matthew Aitchison, Penny Sweetser, and Marcus Hutter. Atari-5: Distilling the arcade learning environment down to five games. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 421–438. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/aitchison23a.html.

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

Yurou Chen, Fengyi Zhang, and Zhiyong Liu. Adaptive advantage estimation for actor-critic algorithms. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2021. doi: 10.1109/IJCNN52387.2021.9534005.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *CoRR*, abs/2306.13831, 2023.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in neural information processing systems*, pp. 4299–4307, 2017.

Brett Daley, Martha White, and Marlos C. Machado. Averaging n-step returns reduces variance in reinforcement learning. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 9904–9930. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/daley24a.html.

Tiantian Fan, Lingjun Liu, Yu Yue, Jiaze Chen, Chengyi Wang, Qiying Yu, Chi Zhang, Zhiqi Lin, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Bole Ma, Mofan Zhang, Gaohong Liu, Ru Zhang, Haotian Zhou, Cong Xie, Ruidong Zhu, Zhi Zhang, Xin Liu, Mingxuan Wang, Lin Yan, and Yonghui Wu. Truncated proximal policy optimization, 2025. URL https://arxiv.org/abs/2506.15050.

Matteo Gallici, Mattie Fellows, Benjamin Ellis, Bartomeu Pou, Ivan Masmitja, Jakob Foerster, and Mario Martin. Simplifying deep temporal difference learning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 78148–78190, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/c23f3852601f6dd7f0b39223d031806f-Paper-Conference.pdf.

Evan Greensmith, Peter Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. In T. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL https://proceedings.neurips.cc/paper_files/paper/2001/file/584b98aac2dddf59ee2cf19ca4ccb75e-Paper.pdf.

Shixiang Gu, Timothy P. Lillicrap, Zoubin Ghahramani, Richard E. Turner, and Sergey Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic. *CoRR*, abs/1611.02247, 2016. URL http://arxiv.org/abs/1611.02247.

Vijay Konda and John Tsitsiklis. Actor-critic algorithms. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

George Konidaris, Scott Niekum, and Philip Thomas. Td_gamma: Re-evaluating complex backups in temporal difference learning. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper_files/paper/2011/file/fb60d411a5c5b72b2e7d3527cfc84fd0-Paper.pdf.

Robert T. Lange. gymnax-blines: Baselines for gymnax. https://github.com/RobertTLange/gymnax-blines. Accessed: 2026-09-26.

Robert T. Lange. gymnax: A JAX-based reinforcement learning environment library, 2026. URL http://github.com/RobertTLange/gymnax.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

Hao Liu, Yihao Feng, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. Action-depedent control variates for policy optimization via stein's identity, 2018. URL https://arxiv.org/abs/1710.11198.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

OpenAI, :, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d. O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning, 2019a. URL https://arxiv.org/abs/1912.06680.

OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Alex Petron, Alex Paino, Matthias Plappert, Glenn Powell, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019b.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

Hsiao-Ru Pan, Nico Gürtler, Alexander Neitz, and Bernhard Schölkopf. Direct advantage estimation. *Advances in Neural Information Processing Systems*, 35:11869–11880, 2022.

Eduardo Pignatelli, Johan Ferret, Matthieu Geist, Thomas Mesnard, Hado van Hasselt, and Laura Toni. A survey of temporal credit assignment in deep reinforcement learning. 2024.

Koen Ponse. Jaxnasium: A Lightweight Utility Library for JAX-based RL Projects. https://github.com/ponseko/jaxnasium, 2025. Version 0.0.27.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: reliable reinforcement learning implementations. *J. Mach. Learn. Res.*, 22(1), January 2021. ISSN 1532-4435.

Md Masudur Rahman and Yexiang Xue. Bootstrap advantage estimation for policy optimization in reinforcement learning, 2022. URL https://arxiv.org/abs/2210.07312.

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

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018. URL https://arxiv.org/abs/1506.02438.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Xiulei Song, Yizhao Jin, Greg Slabaugh, and Simon Lucas. Partial advantage estimator for proximal policy optimization, 2023. URL https://arxiv.org/abs/2301.10920.

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. *MIT press, second edition*, 2018.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 1999.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A standard interface for reinforcement learning environments, 2024. URL https://arxiv.org/abs/2407.17032.

George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E. Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning, 2018. URL https://arxiv.org/abs/1802.10031.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines, 2018. URL https://arxiv.org/abs/1803.07246.

Kenny Young and Tian Tian. Minatar: An atari-inspired testbed for thorough and reproducible reinforcement learning experiments. *arXiv preprint arXiv:1903.03176*, 2019.

A PROOFS AND DETAILS FOR FINITE-TIME GAE

Remark A.1 (Indexing convention vs. Schulman et al. (2018)). The original GAE paper (Schulman et al., 2018) defines the k-step advantage with bootstrapping at time $t+k$. In contrast, our definition in Section 2 bootstraps at time $t+k+1$. Equivalently, our k-step estimator corresponds to the $(k+1)$-step estimator in the indexing used in Schulman et al. (2018). This is purely a notational shift chosen so that geometric mixtures take the form $\sum_{k\geq 0}\lambda^k \hat{\mathbb{A}}_t^{(k)}$.

Proposition 2.1 (Tail-mass collapse of truncated GAE). *For $t \leq \tau-1$, truncated GAE satisfies*

$$\hat{\mathbb{A}}_t^{\mathrm{trunc},\lambda} = \sum_{k=0}^{\tau-t-2} \underbrace{(1-\lambda)\lambda^k}_{\textit{GAE weights}} \hat{\mathbb{A}}_t^{(k)} + \underbrace{\lambda^{\tau-t-1}}_{\textit{collapsed tail-mass}} \hat{\mathbb{A}}_t^{(\tau-t-1)}. \tag{1}$$

Proof. Fix $t \in \{0,\dots,\tau-1\}$. Let $\tilde{\mathbb{A}}_t := (1-\lambda)\sum_{k=0}^{\tau-t-2}\lambda^k\,\hat{\mathbb{A}}_t^{(k)} \;+\; \lambda^{\tau-t-1}\,\hat{\mathbb{A}}_t^{(\tau-t-1)}$. We prove that the identity for $\tilde{\mathbb{A}}_t$ exactly yields truncated GAE. Since $\hat{\mathbb{A}}_t^{(k)} = \sum_{\ell=0}^{k}\gamma^\ell\delta_{t+\ell}$, we obtain

$$\tilde{\mathbb{A}}_t = (1-\lambda)\sum_{k=0}^{\tau-t-2}\lambda^k\sum_{\ell=0}^{k}\gamma^\ell\delta_{t+\ell} \;+\; \lambda^{\tau-t-1}\sum_{\ell=0}^{\tau-t-1}\gamma^\ell\delta_{t+\ell}.$$

For the first term we swap the order of summation and use standard formulas for

$$\begin{aligned}
(1-\lambda)\sum_{k=0}^{\tau-t-2}\lambda^k\sum_{\ell=0}^{k}\gamma^\ell\delta_{t+\ell} &= (1-\lambda)\sum_{k=0}^{\tau-t-2}\lambda^k\sum_{\ell=0}^{\tau-t-2}\mathbf{1}_{\{\ell\leq k\}}\gamma^\ell\delta_{t+\ell} \\
&= (1-\lambda)\sum_{\ell=0}^{\tau-t-2}\gamma^\ell\delta_{t+\ell}\sum_{k=\ell}^{\tau-t-2}\lambda^k \\
&= (1-\lambda)\sum_{\ell=0}^{\tau-t-2}\gamma^\ell\delta_{t+\ell}\,\lambda^\ell\sum_{j=0}^{\tau-t-\ell-2}\lambda^j \\
&= \sum_{\ell=0}^{\tau-t-2}\gamma^\ell\delta_{t+\ell}\,\lambda^\ell\big(1-\lambda^{\tau-t-\ell-1}\big) \\
&= \sum_{\ell=0}^{\tau-t-2}(\gamma\lambda)^\ell\delta_{t+\ell} \;-\; \lambda^{\tau-t-1}\sum_{\ell=0}^{\tau-t-2}\gamma^\ell\delta_{t+\ell}.
\end{aligned}$$

Plugging this back into the formula for $\tilde{\mathbb{A}}_t$, yields

$$\begin{aligned}
\tilde{\mathbb{A}}_t &= \sum_{\ell=0}^{\tau-t-2}(\gamma\lambda)^\ell\delta_{t+\ell} - \lambda^{\tau-t-1}\sum_{\ell=0}^{\tau-t-2}\gamma^\ell\delta_{t+\ell} + \lambda^{\tau-t-1}\sum_{\ell=0}^{\tau-t-1}\gamma^\ell\delta_{t+\ell} \\
&= \sum_{\ell=0}^{\tau-t-2}(\gamma\lambda)^\ell\delta_{t+\ell} + \lambda^{\tau-t-1}\gamma^{\tau-t-1}\delta_{t+\tau-t-1} = \sum_{\ell=0}^{\tau-t-1}(\gamma\lambda)^\ell\delta_{t+\ell}.
\end{aligned}$$

This is exactly truncated GAE. □

Proposition 3.2. *For $\lambda \neq 1$, FT-GAE admits the TD-sum representation*

$$\hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} = \sum_{\ell=0}^{\tau-t-1}(\gamma\lambda)^\ell\,\frac{1-\lambda^{\tau-t-\ell}}{1-\lambda^{\tau-t}}\,\delta_{t+\ell}$$

for $t<\tau$, and satisfies the backward recursion

$$\hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} = \delta_t + \gamma\lambda\,\frac{1-\lambda^{\tau-t-1}}{1-\lambda^{\tau-t}}\,\hat{\mathbb{A}}_{t+1}^{\mathrm{FT},\lambda}, \qquad \hat{\mathbb{A}}_\tau^{\mathrm{FT},\lambda} := 0.$$

Proof. Fix $t \in \{0, \dots, \tau - 1\}$.

$$\begin{aligned}
\hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} &= \frac{1-\lambda}{1-\lambda^{\tau-t}} \sum_{k=0}^{\tau-t-1} \lambda^k \hat{\mathbb{A}}_t^{(k)} = \frac{1-\lambda}{1-\lambda^{\tau-t}} \sum_{k=0}^{\tau-t-1} \lambda^k \sum_{\ell=0}^{\tau-t-1} \mathbf{1}_{\{\ell \le k\}} \gamma^\ell \delta_{t+\ell} \\
&= \frac{1-\lambda}{1-\lambda^{\tau-t}} \sum_{\ell=0}^{\tau-t-1} \gamma^\ell \delta_{t+\ell} \sum_{k=\ell}^{\tau-t-1} \lambda^k = \frac{1-\lambda}{1-\lambda^{\tau-t}} \sum_{\ell=0}^{\tau-t-1} \gamma^\ell \delta_{t+\ell} \, \lambda^\ell \sum_{k=0}^{\tau-t-\ell-1} \lambda^k \\
&= \frac{1-\lambda}{1-\lambda^{\tau-t}} \sum_{\ell=0}^{\tau-t-1} \gamma^\ell \delta_{t+\ell} \, \lambda^\ell \frac{1-\lambda^{\tau-t-\ell}}{1-\lambda} = \frac{1-\lambda}{1-\lambda^{\tau-t}} \sum_{\ell=0}^{\tau-t-1} \gamma^\ell \delta_{t+\ell} \, \frac{\lambda^\ell - \lambda^{\tau-t}}{1-\lambda} \\
&= \sum_{\ell=0}^{\tau-t-1} \frac{\lambda^\ell - \lambda^{\tau-t}}{1-\lambda^{\tau-t}} \gamma^\ell \delta_{t+\ell} = \sum_{\ell=0}^{\tau-t-1} \frac{1-\lambda^{\tau-t-\ell}}{1-\lambda^{\tau-t}} \delta_{t+\ell} (\gamma\lambda)^\ell.
\end{aligned}$$

Thus, we can use above equation to obtain the recursive formula via induction in t. The base case follows easily with $\hat{\mathbb{A}}_\tau^{\mathrm{FT},\lambda} = 0$ as for $t = \tau - 1$, we have $\hat{\mathbb{A}}_{\tau-1}^{\mathrm{FT},\lambda} = \delta_{\tau-1}$. For general $t \in \{0, \dots, \tau - 2\}$, above formula yields

$$\begin{aligned}
\hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} &= \delta_t + \sum_{\ell=1}^{\tau-t-1} \frac{\lambda^\ell - \lambda^{\tau-t}}{1-\lambda^{\tau-t}} \gamma^\ell \delta_{t+\ell} \\
&= \delta_t + \sum_{\ell=0}^{\tau-t-2} \frac{\lambda^{\ell+1} - \lambda^{\tau-t}}{1-\lambda^{\tau-t}} \gamma^{\ell+1} \delta_{t+1+\ell} \\
&= \delta_t + \frac{1-\lambda^{\tau-t-1}}{1-\lambda^{\tau-t}} \lambda\gamma \, \hat{\mathbb{A}}_{t+1}^{\mathrm{FT},\lambda}.
\end{aligned}$$

□

The case $\lambda = 1$ and weight-profiles. The finite-time normalization changes the limiting estimator at $\lambda = 1$. For truncated GAE, Proposition 2.1 gives

$$\hat{\mathbb{A}}_t^{\mathrm{trunc},1} = \hat{\mathbb{A}}_t^{(\tau-t-1)},$$

since all weights except the collapsed endpoint weight vanish. Thus truncated GAE becomes the longest available estimator.

In contrast, FT-GAE is defined at $\lambda = 1$ by the continuous extension of its normalized finite geometric weights:

$$\hat{\mathbb{A}}_t^{\mathrm{FT},1} = \frac{1}{\tau - t} \sum_{k=0}^{\tau-t-1} \hat{\mathbb{A}}_t^{(k)}.$$

Hence FT-GAE at $\lambda = 1$ is not the longest available estimator, but the uniform average over all observable k-step estimators. In that case we can derive the analogue to Proposition 3.2.

Proposition A.2. *For $t < \tau$, the uniform FT-GAE estimator $\hat{\mathbb{A}}_t^{\mathrm{FT},1}$ admits the TD-sum representation*

$$\hat{\mathbb{A}}_t^{\mathrm{FT},1} = \sum_{\ell=0}^{\tau-t-1} \gamma^\ell \frac{\tau - t - \ell}{\tau - t} \delta_{t+\ell}.$$

Moreover, with $\hat{\mathbb{A}}_\tau^{\mathrm{FT},1} := 0$, it satisfies the backward recursion

$$\hat{\mathbb{A}}_t^{\mathrm{FT},1} = \delta_t + \gamma \frac{\tau - t - 1}{\tau - t} \hat{\mathbb{A}}_{t+1}^{\mathrm{FT},1}.$$

Proof. Using

$$\hat{\mathbb{A}}_t^{(k)} = \sum_{\ell=0}^{k} \gamma^\ell \delta_{t+\ell},$$

we obtain

$$\hat{\mathbb{A}}_t^{\mathrm{FT},1} = \frac{1}{\tau - t} \sum_{k=0}^{\tau-t-1} \hat{\mathbb{A}}_t^{(k)} = \frac{1}{\tau - t} \sum_{k=0}^{\tau-t-1} \sum_{\ell=0}^{k} \gamma^\ell \delta_{t+\ell}.$$

Exchanging the order of summation gives

$$\begin{aligned}\hat{\mathbb{A}}_t^{\mathrm{FT},1} &= \frac{1}{\tau - t} \sum_{\ell=0}^{\tau-t-1} \gamma^\ell \delta_{t+\ell} \sum_{k=\ell}^{\tau-t-1} 1 \\ &= \sum_{\ell=0}^{\tau-t-1} \gamma^\ell \frac{\tau - t - \ell}{\tau - t} \delta_{t+\ell}.\end{aligned}$$

This proves the TD-sum representation.

For the recursion, separate the first TD error:

$$\hat{\mathbb{A}}_t^{\mathrm{FT},1} = \delta_t + \sum_{\ell=1}^{\tau-t-1} \gamma^\ell \frac{\tau - t - \ell}{\tau - t} \delta_{t+\ell}.$$

With the change of index $r = \ell - 1$, this becomes

$$\begin{aligned}\hat{\mathbb{A}}_t^{\mathrm{FT},1} &= \delta_t + \gamma \sum_{r=0}^{\tau-t-2} \gamma^r \frac{\tau - t - 1 - r}{\tau - t} \delta_{t+1+r} \\ &= \delta_t + \gamma \frac{\tau - t - 1}{\tau - t} \sum_{r=0}^{\tau-t-2} \gamma^r \frac{\tau - t - 1 - r}{\tau - t - 1} \delta_{t+1+r}. \\ &= \delta_t + \gamma \frac{\tau - t - 1}{\tau - t} \hat{\mathbb{A}}_{t+1}^{\mathrm{FT},1}.\end{aligned}$$

□

We now use Propositions 2.1 to directly relate the estimators.

Corollary A.3 (Truncated GAE as a mixture of FT-GAE and the longest target)**.** *Fix $t \le \tau - 1$. Then*

$$\hat{\mathbb{A}}_t^{\mathrm{trunc},\lambda} = (1 - \lambda^{\tau-t}) \hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} + \lambda^{\tau-t} \hat{\mathbb{A}}_t^{(\tau-t-1)}. \tag{3}$$

For $\lambda \in [0, 1]$, this is a convex combination of FT-GAE and the longest observable k-step estimator.

Proof. By definition of FT-GAE,

$$(1 - \lambda^{\tau-t}) \hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} = (1 - \lambda) \sum_{k=0}^{\tau-t-1} \lambda^k \hat{\mathbb{A}}_t^{(k)}.$$

Therefore,

$$\begin{aligned}&(1 - \lambda^{\tau-t}) \hat{\mathbb{A}}_t^{\mathrm{FT},\lambda} + \lambda^{\tau-t} \hat{\mathbb{A}}_t^{(\tau-t-1)} \\ &= (1 - \lambda) \sum_{k=0}^{\tau-t-2} \lambda^k \hat{\mathbb{A}}_t^{(k)} + (1 - \lambda) \lambda^{\tau-t-1} \hat{\mathbb{A}}_t^{(\tau-t-1)} + \lambda^{\tau-t} \hat{\mathbb{A}}_t^{(\tau-t-1)} \\ &= (1 - \lambda) \sum_{k=0}^{\tau-t-2} \lambda^k \hat{\mathbb{A}}_t^{(k)} + \lambda^{m-1} \hat{\mathbb{A}}_t^{(\tau-t-1)} \\ &= \hat{\mathbb{A}}_t^{\mathrm{trunc},\lambda},\end{aligned}$$

where the last equality follows from Proposition 2.1. □

Equation (3) gives another view of tail-mass collapse. Truncated GAE equals the normalized finite geometric mixture used by FT-GAE, plus additional mass $\lambda^{\tau-t}$ placed on the longest observable k-step estimator. Thus, for large λ or short rollout suffixes $\tau - t$, truncated GAE becomes substantially more Monte-Carlo-like near the rollout endpoints. FT-GAE removes exactly this additional mass while preserving the relative geometric weighting over the observable horizons. Figure 11 visualizes the comparison of the weighting schemes for truncated GAE and our FT-GAE for different values of λ.

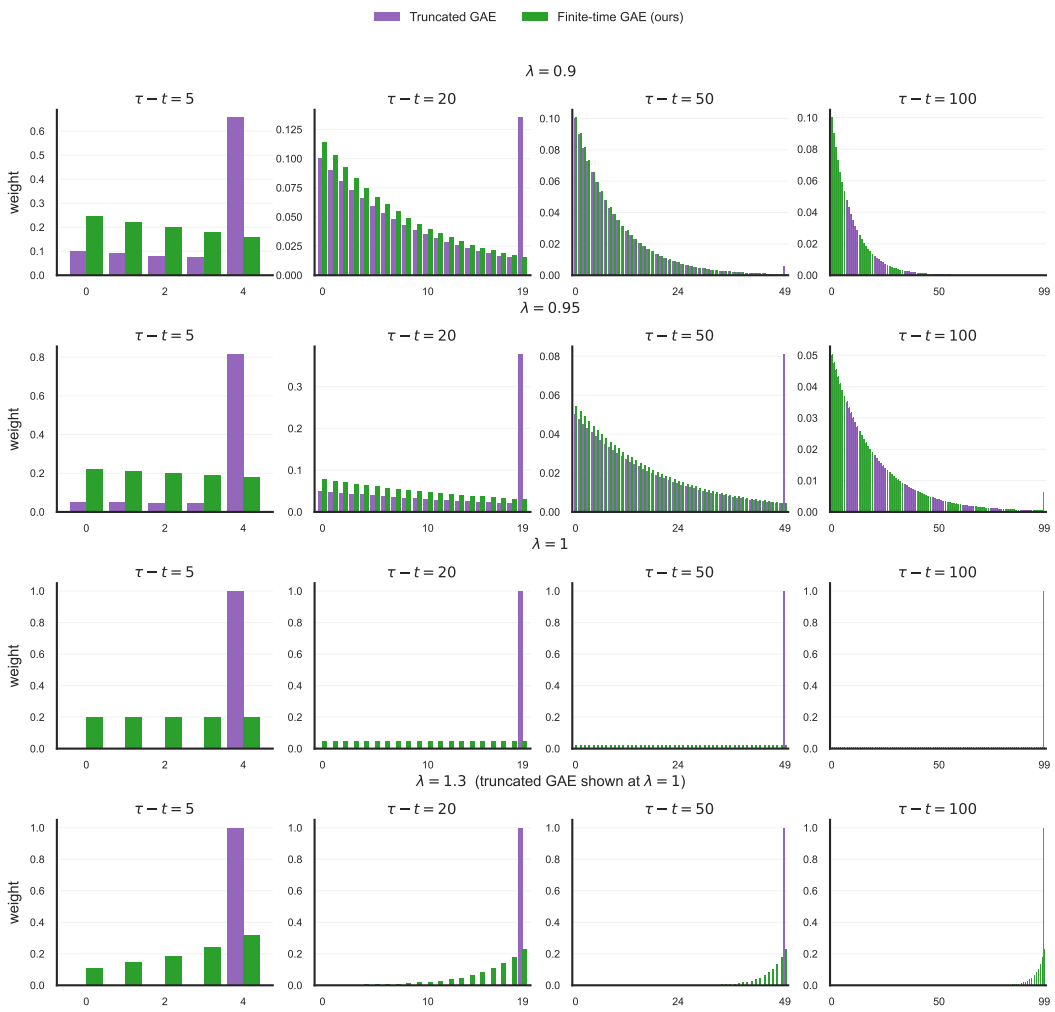


Figure 11: Weight profiles for truncated GAE and FT-GAE across different values of λ (rows) and remaining horizons $\tau - t$ (columns). For $\lambda \leq 1$, truncated GAE exhibits the tail-mass collapse described in Proposition 2.1: lowering λ reduces the collapsed endpoint mass, but changes the estimator globally and does not remove the underlying effect. FT-GAE instead forms a normalized finite mixture over the available k-step estimators. At $\lambda = 1$, this mixture is uniform. For $\lambda > 1$, it remains well-defined, with nonnegative weights that sum to one and increase with k, thereby emphasizing longer lookahead estimators. Truncated GAE has no analogous normalized weight profile for $\lambda > 1$. In those rows, we therefore show its $\lambda = 1$ profile as a reference.

B GRIDWORLD-FETCH DETAILS

This section provides the environment, training, and diagnostic details for the Gridworld-Fetch experiment in Section 4.1. The main results appear in Figure 3. Figures 12,13 and 14 show additional cuts of the diagnostics.

Environment: We use a time-inhomogeneous GridWorld (adding time to the state-space makes it a standard MDP) which is hard enough to challenge PPO but simple enough to perform explicit computations of V^π, Q^π, and $\mathbb{A}^\pi$. The environment is a reproduction of Minigrid-Fetch (Chevalier-Boisvert et al., 2023) as Gridworld. Each episode samples distinct interior cells for the agent, target, and distractor, and one of four agent orientations. The intended actions are turn left, turn right, move forward, and pick up the object immediately ahead. The intended action executes with probability 0.8. With probability 0.2, one of the other three actions executes uniformly. Walls and objects block forward movement. A successful pickup on transition t gives

$$R_t = 2\left[1 - 0.9\frac{t+1}{225}\right]. \tag{4}$$

Picking up the distractor gives -1, and all other transitions give zero. Either pickup ends the episode. The task also ends after 225 steps. We include elapsed time in the state, making the finite-horizon task a time-homogeneous Markov decision process on the augmented state space. Its state specifies time, agent pose, and both object positions.

Experiment details: Training. The policy observes 11 components: two normalized agent coordinates, a four-component one-hot orientation indicator, two normalized coordinates for each object, and elapsed time $t/225$. We train independent Truncated-GAE and FT-GAE policies with the SBX PPO implementation (Raffin et al., 2021). Each method uses ten seeds for 1,000,000 steps. We use eight parallel environments, 128 transitions per environment and update, 64-transition minibatches, and ten PPO epochs per update together with $\gamma = 0.99$, $\lambda = 0.95$, learning rate 2.5×10^{-4}, clip range 0.2, value-loss coefficient 0.5, entropy coefficient 0, maximum gradient norm 0.5, and advantage normalization within each minibatch. We save policies at 0%, 5%, 10%,25%, 50%, 75%, and 100% of the training budget for both training configurations (truncated GAE and FT-GAE).

Exact references and diagnostic data. For each frozen checkpoint, we collect rollouts of 4096 transitions under its stochastic policy. For each object layout encountered in these rollouts, we compute exact reference values using the environment's transition probabilities, rewards, and the checkpoint policy. We evaluate both estimators on the *same* transitions and compare the estimators with the exact advantages to compute RMSE. RMSEs are computed across selected visited transitions within each checkpoint batch. At each saved checkpoint, we evaluate both the sampled policy gradient and the gradient of the clipped PPO surrogate (Figure 12). For each quantity, we measure the cosine similarity between the gradient computed with either advantage estimator and the corresponding gradient computed with exact $\mathbb{A}^\pi$.

Further results: Figure 12 compares the direction of gradients computed from estimated advantages with the corresponding direction obtained from exact advantages. FT-GAE generally achieves higher cosine similarity after initialization on checkpoints from both training methods. The improvement also appears in the sampled policy-gradient diagnostic, showing that the favorable alignment is not confined to the clipped PPO objective. Since both estimators use the same diagnostic data and critic, these comparisons support an improvement in gradient direction from changing the finite-segment weighting.

Figure 13 resolves the PPO-surrogate comparison by checkpoint and diagnostic epoch. Each cell reports FT-GAE's cosine similarity minus that of truncated GAE, using the exact-advantage gradient as the common reference. Most cells favor FT-GAE, with positive differences throughout the trained checkpoints. The advantage therefore persists across repeated PPO epochs rather than being restricted to the first pass over the diagnostic data.

Figure 14 compares raw advantage RMSE against exact $\mathbb{A}^\pi$ for different distance regimes to a rollouts endpoint. FT-GAE yields particularly large error reductions near terminations, and the gap narrows farther from the boundary. This pattern appears on both training paths.

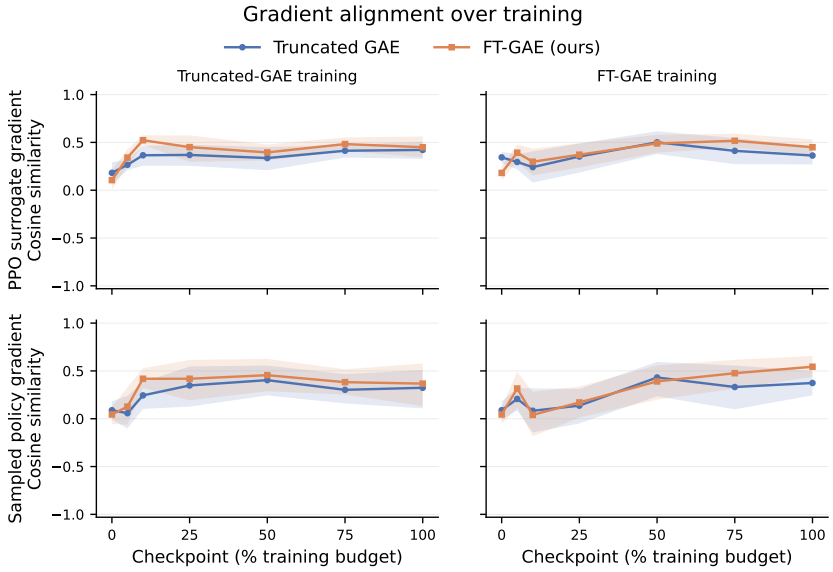


Figure 12: **Gradient alignment across training checkpoints.** Cosine similarity between gradients computed with estimated advantages and the corresponding gradients computed with exact advantages. The top row uses the clipped PPO surrogate. The bottom row uses the sampled policy gradient. Columns identify the method used to train the checkpoint policy. Within each panel, both estimators are evaluated on the same diagnostic data. Curves show means across ten training seeds, with 95% seed-bootstrap intervals. Higher values indicate closer alignment with the exact-advantage gradient.

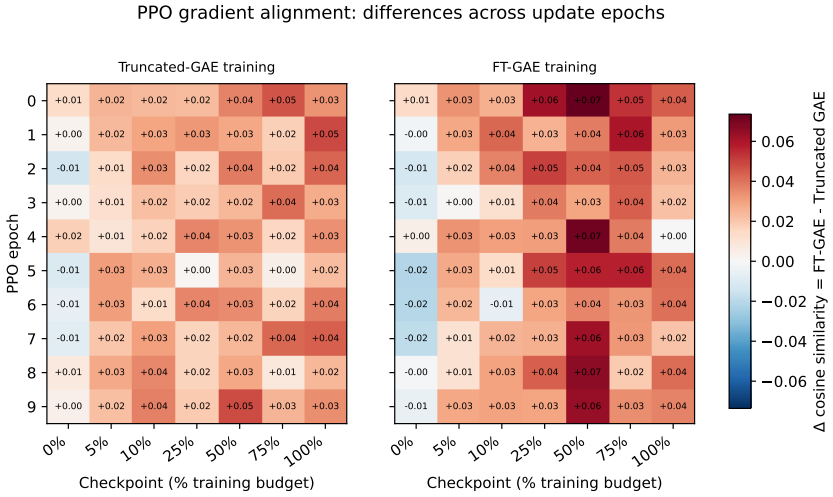


Figure 13: Cells show the paired difference in cosine similarity for clipped PPO-surrogate gradients computed with true advantages. Positive values favor FT-GAE. The horizontal axis identifies the training checkpoint. The vertical axis identifies the epoch. The left and right panels use checkpoints from truncated-GAE and FT-GAE training, respectively. Cell values are averaged across ten training seeds.

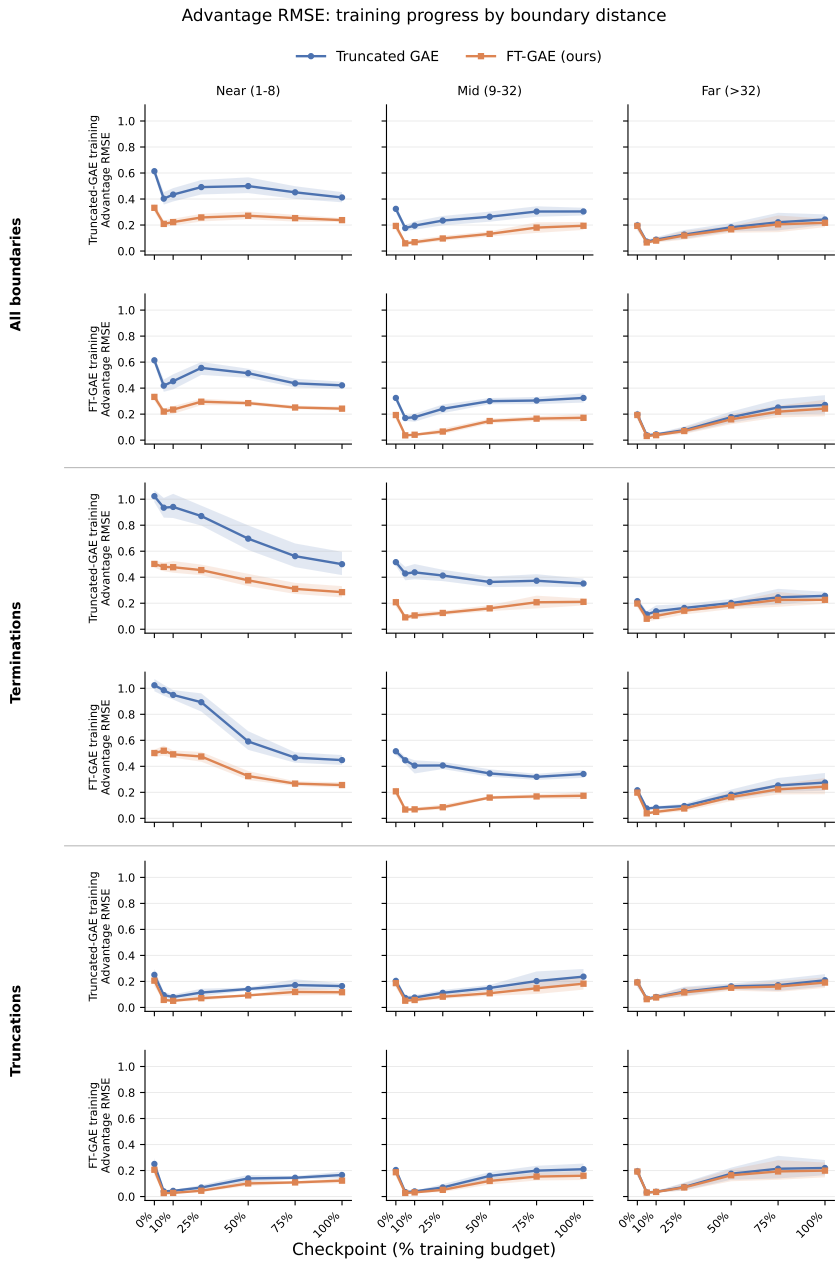


Figure 14: **Advantage accuracy by boundary distance and training progress.** Columns group transitions by remaining segment length. Row pairs show all boundaries, true pickup terminations, and truncations. Within each pair, the upper row uses truncated-GAE-trained checkpoints and the lower row uses FT-GAE-trained checkpoints. Curves compare advantage estimates with exact $\mathbb{A}^{\pi}$ on paired diagnostic transitions. Shading shows 95% seed-bootstrap intervals across ten training seeds. Lower RMSE is better.

C PROOFS OF MATHEMATICAL INSIGHTS

This section provides details for section 4.2 and proves Proposition 4.2. The purpose of the toy model is to isolate the variance mechanism induced by finite TD-error aggregation near rollout endpoints. We consider a rollout segment of fixed length T. Let

$$\delta_t := R_t + \gamma V(S_{t+1}) - V(S_t)$$

be the TD error. We study estimators that can be written as convex combinations of observable n-step bootstrapped returns. Although the main text only compares the standard truncated λ-return with the finite-time λ-return, it is useful to state the variance identity for general compound returns.

For $n \in \{0, \dots, T-t-1\}$, consider the n-step bootstrapped return

$$G_t^{(n)} := \sum_{i=0}^{n} \gamma^i R_{t+i} + \gamma^{n+1} V(S_{t+n+1}).$$

Given weights $c_0, \dots, c_{T-t-1}$ satisfying

$$c_n \geq 0, \qquad \sum_{n=0}^{T-t-1} c_n = 1,$$

the corresponding compound return is

$$G_t^c := \sum_{n=0}^{T-t-1} c_n G_t^{(n)}.$$

Lemma C.1 (TD-error representation of finite compound returns)**.** *For every finite compound return G_t^c,*

$$G_t^c - V(S_t) = \sum_{i=0}^{T-t-1} \gamma^i h_i \delta_{t+i},$$

where

$$h_i := \sum_{n=i}^{T-t-1} c_n, \qquad i = 0, \dots, T-t-1.$$

Proof. For each n-step return, the standard telescoping identity gives

$$G_t^{(n)} - V(S_t) = \sum_{i=0}^{n} \gamma^i \delta_{t+i}.$$

Therefore,

$$\begin{aligned} G_t^c - V(S_t) &= \sum_{n=0}^{T-t-1} c_n \big(G_t^{(n)} - V(S_t)\big) \\ &= \sum_{n=0}^{T-t-1} c_n \sum_{i=0}^{n} \gamma^i \delta_{t+i}. \end{aligned}$$

Since all sums are finite, we can exchange the order of summation:

$$\begin{aligned} G_t^c - V(S_t) &= \sum_{i=0}^{T-t-1} \gamma^i \delta_{t+i} \sum_{n=i}^{T-t-1} c_n \\ &= \sum_{i=0}^{T-t-1} \gamma^i h_i \delta_{t+i}. \end{aligned}$$

This proves the claim. □

This finite-horizon statement is the rollout-local analogue of the compound return representation used in Daley et al. (2024, Lemma 3.4).

We now impose a simple covariance model on the TD errors.

Assumption 4.1. We assume $e_t := \mathbb{E}[V(S_t) - V^\pi(S_t)] = \varepsilon_{\min} + (\varepsilon_{\max} - \varepsilon_{\min})\left(\frac{T-t}{T}\right)^2$ and $\mathrm{Cov}[\delta_{t+i}, \delta_{t+j} \mid S_t] = \kappa\rho^{|i-j|}$, for $i, j = 0, \dots, T-t-1$, with $\kappa > 0$ and $\rho \in [0, 1]$.

Proposition 4.2. *Let $h_i := \sum_{n=i}^{T-t-1} c_n$ denote the cumulative weights induced by the compound estimator. Then, under Assumption 4.1:* $\mathrm{Var}[\hat{G}_t^c \mid S_t] = \kappa \sum_{i=0}^{T-t-1} \sum_{j=0}^{T-t-1} \gamma^{i+j} h_i h_j \rho^{|i-j|}$.

Proof. By Lemma C.1,

$$G_t^c - V(S_t) = \sum_{i=0}^{T-t-1} \gamma^i h_i \delta_{t+i}.$$

Since $V(S_t)$ is deterministic conditional on S_t,

$$\mathrm{Var}[G_t^c \mid S_t] = \mathrm{Var}\Big[\sum_{i=0}^{T-t-1} \gamma^i h_i \delta_{t+i} \mid S_t \Big].$$

Using linearity of conditional covariance,

$$\begin{aligned}\mathrm{Var}[G_t^c \mid S_t] &= \sum_{i=0}^{T-t-1} \sum_{j=0}^{T-t-1} \gamma^i h_i \gamma^j h_j \,\mathrm{Cov}\left[\delta_{t+i}, \delta_{t+j} \mid S_t\right] \\ &= \sum_{i=0}^{T-t-1} \sum_{j=0}^{T-t-1} \gamma^{i+j} h_i h_j \,\mathrm{Cov}\left[\delta_{t+i}, \delta_{t+j} \mid S_t\right].\end{aligned}$$

Substituting the covariance model from Assumption 4.1 yields

$$\mathrm{Var}[G_t^c \mid S_t] = \kappa \sum_{i=0}^{T-t-1} \sum_{j=0}^{T-t-1} \gamma^{i+j} h_i h_j \rho^{|i-j|}.$$

□

We now specialize the identity to the two estimators used in the main text. The standard truncated λ-return satisfies

$$\hat{G}_t^{\mathrm{trunc},\lambda} - V(S_t) = \sum_{i=0}^{T-t-1} (\gamma\lambda)^i \delta_{t+i}.$$

Equivalently, its TD-error coefficients are

$$h_i^{\mathrm{trunc}} = \lambda^i, \qquad i = 0, \dots, T-t-1.$$

Hence,

$$\mathrm{Var}\left[\hat{G}_t^{\mathrm{trunc},\lambda} \mid S_t\right] = \kappa \sum_{i=0}^{T-t-1} \sum_{j=0}^{T-t-1} (\gamma\lambda)^{i+j} \rho^{|i-j|}.$$

For the finite-time λ-return, the normalized finite geometric mixture gives

$$\hat{G}_t^{\mathrm{FT},\lambda} - V(S_t) = \sum_{i=0}^{T-t-1} (\gamma\lambda)^i \frac{1-\lambda^{T-t-i}}{1-\lambda^{T-t}} \delta_{t+i}.$$

Thus,

$$h_i^{\mathrm{FT}} = \lambda^i \frac{1-\lambda^{T-t-i}}{1-\lambda^{T-t}}, \qquad i = 0, \dots, T-t-1.$$

The corresponding conditional variance is

$$\mathrm{Var}\left[\hat{G}_t^{\mathrm{FT},\lambda} \mid S_t\right] = \kappa \sum_{i=0}^{T-t-1} \sum_{j=0}^{T-t-1} (\gamma\lambda)^{i+j} \frac{1-\lambda^{T-t-i}}{1-\lambda^{T-t}} \frac{1-\lambda^{T-t-j}}{1-\lambda^{T-t}} \rho^{|i-j|}.$$

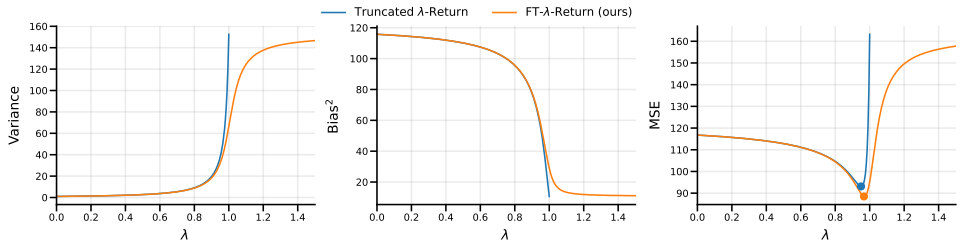


Figure 15: Rollout-averaged MSE analysis in an informative-terminal regime. We use $T = 100$, $\gamma = 0.99$, $\rho = 0.7$, $\kappa = 1$, and critic bias $e_t = 5 + 15((T-t)/T)^2$. The model was designed to mimic episodic tasks with strong terminal reward. Our FT-λ-return reduces MSE (bias^2+variance).

Since $0 < \lambda < 1$, we have

$$0 \le \frac{1-\lambda^{T-t-i}}{1-\lambda^{T-t}} \le 1, \qquad i = 0, \dots, T-t-1.$$

Therefore,

$$0 \le h_i^{\mathrm{FT}} \le h_i^{\mathrm{trunc}} \qquad \text{for all } i = 0, \dots, T-t-1.$$

Under the nonnegative covariance model $\rho \in [0, 1]$, every summand in the variance expression is nonnegative. Consequently,

$$\mathrm{Var}\left[\hat{G}_t^{\mathrm{FT},\lambda} \mid S_t\right] \le \mathrm{Var}\left[\hat{G}_t^{\mathrm{trunc},\lambda} \mid S_t\right].$$

Thus, under the assumed nonnegative TD-error covariances, finite-time normalization reduces the conditional variance at fixed λ. At $T - t = 1$, both estimators equal the one-step return and have identical variance. The magnitude of the reduction depends on the remaining segment length and the parameters γ, λ, and ρ.

D Policy Evaluation Experiment

D.1 Truncated λ-return vs. FT-λ-return for policy evaluation

Recall that GAE and λ-returns are related by $\hat{G}_t^{\infty,\lambda} = \hat{\mathbb{A}}_t^{\infty,\lambda} + V(S_t)$. Thus, adding the value estimate $V(S_t)$ to FT-GAE yields the corresponding FT-λ-return. We evaluate this target in the random-walk prediction task of (Sutton & Barto, 2018, Example 7.1) by using offline TD updates of the form

$$V(S_t) \leftarrow V(S_t) + \alpha\big(\hat{G}_t - V(S_t)\big).$$

The purpose of this experiment is to isolate the estimator effect that motivates FT-GAE. In PPO, changes in performance can be caused by policy improvement, clipping, optimization, exploration, and neural-network approximation. The random-walk task removes these factors and asks a simpler question: when the effective backup length is held fixed, does finite-time normalization change the behavior of the target itself? Following the comparison principle of Daley et al. (2024), we compare three target families: (i) fixed n-step returns, (ii) standard truncated λ-returns, (iii) FT-λ-returns.

To make this comparison meaningful, we match the finite-horizon center of mass of each compound return to the fixed backup lengths $n = 1, 2, 4, 9, 19, 49$. Thus, all three target families are compared at similar effective backup lengths, although the corresponding λ values differ between truncated and finite-time normalization. Full details are given below.

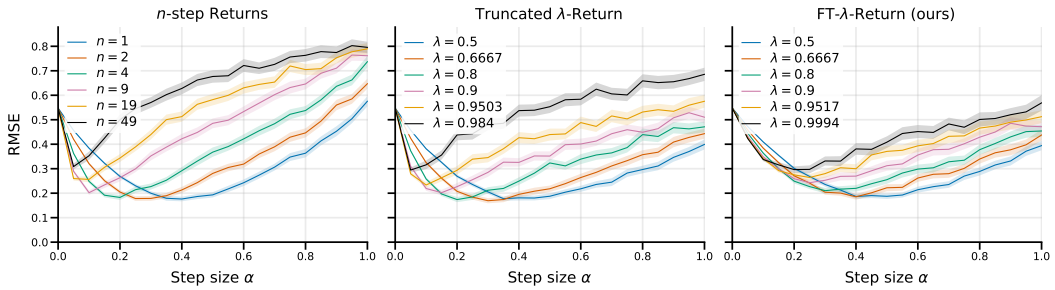


Figure 16: Random-walk policy evaluation with 19 non-terminal states, rollout horizon $T = 100$, and 20 training episodes. Curves show the root mean-squared error to the true value function averaged over all non-terminal states and then over 200 independent trials. Shaded regions indicate 95% confidence intervals. Our FT-λ-return estimator improves truncated λ-return for longer effective backup lengths and, by design, is almost identical for short backup lengths.

Figure 16 reports the root-mean-square error to the true value function V^π as a function of the step size α, averaged over states and over 200 independent trials after 20 training episodes. The lowest attainable errors are similar across the three target families, but their stability at large step sizes differs. Fixed n-step returns become unstable first, especially for longer backups. Compound returns are more stable because they average over multiple backup lengths. FT-λ-returns are the most stable in this regime, particularly at long effective backup lengths. Since the effective backup length is matched, this improvement cannot be attributed to using shorter backups. Rather, it supports the mechanism identified in our analysis: near rollout endpoints, standard truncated λ-returns place the remaining geometric tail mass on the longest available return, whereas FT-λ-returns redistribute this mass over all available backups. This reduces the influence of high-variance long returns while preserving the same effective backup scale.

D.2 Experiment details

We evaluate the return estimators in an undiscounted Random Walk prediction task. The environment consists of 19 nonterminal states arranged on a line, with two terminal states at the left and right ends. Each episode starts from the center state. The policy is fixed and moves left or right with equal probability. The agent receives reward -1 upon reaching the left terminal state and reward $+1$ upon reaching the right terminal state; all other rewards are zero. The episode terminates when either terminal state is reached. To mimic the finite rollout structure used in actor-critic methods, episodes

are truncated after $T = 100$ environment steps. Thus, each rollout has the form

$$(S_0, R_0, S_1, R_1, \ldots, S_{\tau-1}, R_{\tau-1}, S_\tau),$$

where τ is the minimum of the terminal hitting time and the truncation horizon T. If S_τ is terminal, its value is zero. If the rollout is truncated before termination, targets bootstrap from the current value estimate $V(S_\tau)$.

We train value functions for 20 episodes using offline TD updates of the form

$$V(S_t) \leftarrow V(S_t) + \alpha(\hat{G}_t - V(S_t)).$$

All value estimates are initialized to zero. For each complete rollout, all targets are computed from the value estimate before the offline updates for that rollout are applied. We compare three target families: fixed N-step returns, the classical truncated λ-return with residual mass on the longest available return, and the finite-time normalized λ-return. The estimators are given by

- $\hat{G}_t^{(n)} := \sum_{\ell=0}^{n} \gamma^l R_{t+l} + \gamma^{n+1} V(S_{t+n+1})$, which is the n-step return estimator,
- $\hat{G}_t^{\mathrm{trunc},\lambda} := \sum_{k=0}^{\tau-t-2} (1-\lambda)\lambda^k \, \hat{G}_t^{(k)} + \lambda^{\tau-t-1} \, \hat{G}_t^{(\tau-t-1)}$, which is the truncated λ-return and
- $\hat{G}_t^{\mathrm{FT},\lambda} := \frac{1-\lambda}{1-\lambda^{\tau-t}} \sum_{k=0}^{\tau-t-1} \lambda^k \, \hat{G}_t^{(k)}$, which corresponds to the Finite-time λ-return analogue to our estimator of definition 3.1.

Following Daley et al. (2024), we match compound returns to fixed n-step returns by equating their effective backup length, also known as center of mass (COM). For a compound return $\hat{G}_t^c := \sum_{k=0}^{\infty} c_k \hat{G}_t^k$ the COM is defined by $\sum_{k=0}^{\infty} c_k \, (k+1)$ (c.f. Daley et al. (2024) and remark A.1.) For each target value $n = 1, 2, 4, 9, 19, 49$, we choose separate λ values for the classical truncated λ-return $\hat{G}_0^{\mathrm{trunc},\lambda}$ and for the finite-time normalized λ-return $\hat{G}_0^{\mathrm{FT},\lambda}$ such that their COM for the maximum available rollout-length equals N in order to have a comparable setup (up to some small discrepancy due to the truncation or termination of rollouts). We sweep

$$\alpha \in \{0, 0.05, 0.10, \ldots, 1.0\}$$

and average all reported RMSE values over 200 independent trials. After training, prediction error is measured as

$$\mathrm{RMSE}(V) = \sqrt{\frac{1}{19} \sum_{i=1}^{19} \big(V(i) - V^\pi(i)\big)^2}.$$

E DEEP RL EXPERIMENTS DETAILS

To assess the robustness of the proposed estimator, we evaluate it across a broad suite of environments covering discrete-action visual control, sparse or structured navigation tasks, and continuous-control benchmarks. The suite includes the Arcade Learning Environment and Atari benchmarks (Bellemare et al., 2013; Aitchison et al., 2023), Box2D, MiniGrid (Chevalier-Boisvert et al., 2023), and MuJoCo (Todorov et al., 2012), all accessed through the Gymnasium API (Towers et al., 2024). In addition we train on five MinAtar environments (Young & Tian, 2019), where we use Gymnax (Lange, 2026). The implementations are based on the RL Baselines3 Zoo training framework (Raffin et al., 2021) and a Jaxnasium-based PPO implementation (Ponse, 2025) and require only a few local modifications to the standard advantage-estimation code. We keep the original benchmark hyperparameters fixed throughout and perform no additional environment-specific tuning, except for the explicitly reported λ-sweeps. This section contains all details for Section 5 of the main paper.

E.1 IMPLEMENTATION DETAILS FOR DEEP RL EXPERIMENTS

All deep reinforcement learning experiments use the SBX-PPO implementation of Stable-Baselines3 (SB3) (Raffin et al., 2021) or Jaxnasium-based PPO implementation (Ponse, 2025). The PPO optimization procedures itself are left unchanged. In particular, the policy loss, value loss, entropy regularization, minibatch construction, clipping rule, optimizer, and hyperparameters are the SB3-Zoo choices reported in Section E.3. For our MinAtar experiments we adapted the hyperparameters from the MinAtar PPO configurations in gymnax-blines (Lange). The only modification is the computation of advantages inside the rollout buffer, following Algorithm 1.

Standard PPO computes truncated GAE by a backward recursion over the rollout. By Propositions 3.2 and A.2, FT-GAE admits the same type of backward recursion, with the usual factor $\gamma\lambda$ replaced by a finite-horizon coefficient. Let $h := \tau - t \geq 1$ denote the distance to the rollout endpoint, which is either an episode endpoint or the end of the rollout buffer. The coefficient used in the FT-GAE recursion is

$$c(h, \lambda, \gamma) = \begin{cases} \gamma\lambda \dfrac{1 - \lambda^{h-1}}{1 - \lambda^h} & : \lambda \neq 1 \\ \gamma \dfrac{h-1}{h} & : \lambda = 1. \end{cases}$$

Thus, FT-GAE can be implemented as a local replacement of the usual GAE recursion. The remaining horizon h is not stored in the environment. Instead, it is reconstructed during the backward pass using the existing `episode_starts` indicators and the final `dones` signal. The last transition of each rollout segment is initialized with $h = 1$. Moving backward, the counter is increased by one as long as the next transition belongs to the same episode; otherwise it is reset to 1. For vectorized environments, this counter is maintained componentwise.

For numerical stability, we do not evaluate the above coefficients directly when λ is close to one. For $0 < \lambda < 1$, writing $\ell := \log \lambda$, we compute

$$c(h, \lambda, \gamma) = \gamma\lambda \frac{\mathrm{expm1}((h-1)\ell)}{\mathrm{expm1}(h\ell)},$$

where $\mathrm{expm1}(x) := \exp(x) - 1$. The case $\lambda = 1$ is handled by the continuous limit $c(h, 1, \gamma) = \gamma(h-1)/h$. For completeness, if $\lambda > 1$, we use the inverse-power form

$$c(h, \lambda, \gamma) = \gamma \frac{1 - \lambda^{-(h-1)}}{1 - \lambda^{-h}},$$

which avoids overflow for large h. This case showed no remarkable results in our PPO experiments, but it is useful for testing variants with increasing weights on longer k-step estimators.

The following lemma shows that these stable formulas leave the FT-GAE recursion unchanged.

Lemma E.1. *Let $h \geq 1$ and $\gamma \in [0, 1]$. For $0 < \lambda < 1$,*

$$\gamma\lambda \frac{1 - \lambda^{h-1}}{1 - \lambda^h} = \gamma\lambda \frac{\mathrm{expm1}((h-1)\log\lambda)}{\mathrm{expm1}(h \log \lambda)}.$$

For $\lambda > 1$,

$$\gamma\lambda\frac{1-\lambda^{h-1}}{1-\lambda^{h}} = \gamma\frac{1-\lambda^{-(h-1)}}{1-\lambda^{-h}}.$$

Hence the numerically stable implementation is algebraically equivalent to the FT-GAE recursion coefficient.

Proof. For $0 < \lambda < 1$, let $\ell = \log\lambda$. Since $\lambda^m = \exp(m\ell)$ and $1 - \exp(x) = -\operatorname{expm1}(x)$,

$$\frac{1-\lambda^{h-1}}{1-\lambda^{h}} = \frac{-\operatorname{expm1}((h-1)\ell)}{-\operatorname{expm1}(h\ell)} = \frac{\operatorname{expm1}((h-1)\ell)}{\operatorname{expm1}(h\ell)}.$$

This proves the first identity.

For $\lambda > 1$, multiplying numerator and denominator by λ^{-h} gives

$$\gamma\lambda\frac{1-\lambda^{h-1}}{1-\lambda^{h}} = \gamma\lambda\frac{\lambda^{-h}-\lambda^{-1}}{\lambda^{-h}-1} = \gamma\frac{1-\lambda^{-(h-1)}}{1-\lambda^{-h}}.$$ □

E.2 Ablation details

This section provides additional experimental details for selected ablations.

E.2.1 A2C policy-gradient ablation

In Section 5.3, we examine whether the Box2D gains also occur with A2C, which does not use PPO's clipped policy objective. We compare truncated GAE and FT-GAE on LunarLander-v3 and BipedalWalker-v3. Within each environment and λ setting, the two variants share the A2C training configuration and differ in the estimator used for the actor advantages and corresponding critic targets. Each setting uses 20 seeds. We evaluate policies every 10,000 training steps using 16 episodes per evaluation and report the mean area under the evaluation-return learning curve (AUC) and mean final return across seeds. Table 3 gives the training and evaluation settings. Table 2 reports the results. FT-GAE has higher reported means for both metrics in all six comparisons. This result shows that PPO clipping is not required for the observed ordering on these two tasks.

Table 2: A2C results on two Box2D environments. Entries are mean AUC and mean final return over 20 seeds per setting; higher is better. A dagger marks the default λ for each environment.

		Mean AUC		Mean final return	
Environment	λ	Truncated GAE	FT-GAE	Truncated GAE	FT-GAE
LunarLander-v3	0.90	-29.03	$\mathbf{16.46}$	-24.04	$\mathbf{51.49}$
LunarLander-v3	0.95	-46.72	$\mathbf{14.80}$	-47.02	$\mathbf{44.31}$
LunarLander-v3	0.99	-58.94	$\mathbf{15.05}$	-19.22	$\mathbf{14.45}$
LunarLander-v3	$1.00^{\dagger}$	-63.85	$\mathbf{16.75}$	-33.73	$\mathbf{54.89}$
BipedalWalker-v3	0.90	140.87	$\mathbf{193.22}$	201.34	$\mathbf{260.86}$
BipedalWalker-v3	$0.95^{\dagger}$	182.52	$\mathbf{208.60}$	242.26	$\mathbf{278.86}$

Table 3: Hyperparameters and time budgets used in the A2C and A2C-FT-GAE comparison runs. Values are based on the saved run configurations and logs; parameters that were not overridden use the Stable-Baselines3 A2C defaults.

Parameter	BipedalWalker-v3	LunarLander-v3
Training and rollout		
n_timesteps	5×10^6	1×10^6
normalize	True	False
n_envs	16	8
n_steps	8	5
Return and advantage estimation		
gamma	0.99	0.995
gae_lambda	{0.90, 0.95}	{0.90, 0.95, 0.99, 1.00}
normalize_advantage	False	False
Optimization		
learning_rate	lin_0.00096	lin_0.00083
ent_coef	0	10^{-5}
vf_coef	0.4	0.5
max_grad_norm	0.5	0.5
use_rms_prop	True	True
rms_prop_eps	10^{-5}	10^{-5}
Exploration and network		
use_sde	True	False
sde_sample_freq	-1	–
policy	MlpPolicy	MlpPolicy
log_std_init	-2	–
ortho_init	False	True
net_arch	pi=[64,64], vf=[64,64]	pi=[64,64], vf=[64,64]
activation_fn	Tanh	Tanh
Evaluation		
Evaluation frequency	10 000	10 000

E.2.2 DETAILS OF TERMINAL REWARD INTERVENTION

We vary rewards at episode endpoints to examine how the relative performance of truncated GAE and FT-GAE changes with the strength of the terminal signal. The intervention changes the training reward. Performance is measured on a common evaluation scale within each environment. Figure 17 visualizes the resulting performance gaps. The tables below give the underlying means and the additional Hopper controls.

Hopper-v4: failure-termination penalty. We train PPO agents in Hopper-v4 after adding a penalty of $-C$ to the reward at early termination (a fall), with $C \in \{0, 5, 100\}$. Training uses the same Hopper-v4 PPO configuration across penalty levels within each comparison; only the estimator, its specified λ, and the stated reward-normalization control vary. Every policy is evaluated with the original, unmodified reward ($C = 0$), so returns remain comparable across training conditions. Each method-penalty combination uses 40 seeds. The tables report seed means and relative FT-GAE changes, $100(\text{FT} - \text{truncated})/\text{truncated}$.

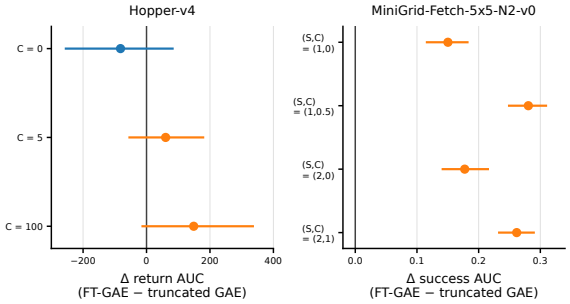


Figure 17: Hopper AUC difference and Minigrid-Fetch success AUC differences. Points show means; bars show 95% bootstrap intervals.

Table 4 uses the best-λ of the sweeps for FT-GAE ($\lambda = 1.0$) and $\lambda = 0.99$ for Truncated-GAE (since the sweep results for truncated GAE were worse than the default value). Reward normalization is enabled during training by default in Stable baselines Zoo. FT-GAE has higher mean final return and AUC at every penalty level; the largest gap occurs at $C = 100$. Since the methods use different λ values, this comparison alone cannot isolate the effect of the estimator.

To hold λ fixed, Table 5 sets $\lambda = 0.99$ for both estimators while retaining reward normalization. This is the comparison shown in Figure 8: at $C = 0$, truncated GAE has higher mean final return and AUC; at $C = 5$ and $C = 100$, FT-GAE has higher means. The FT-GAE AUC difference rises from -8.0% at $C = 0$ to $+16.6\%$ at $C = 100$. Thus, the reversal in the reported means does not depend on assigning a different λ to FT-GAE.

Finally, Table 6 repeats the matched-λ experiment without reward normalization and without retuning the remaining hyperparameters. FT-GAE has higher reported means already at $C = 0$, and its relative advantage is larger at both positive penalties. The direction of the penalty-dependent change therefore persists without reward normalization, although the unpenalized ordering differs between the normalized and unnormalized settings.

MiniGrid-Fetch: success and failure rewards. We also intervene in MiniGrid-Fetch-5x5-N2, a sparse-reward task in which collecting the correct object ends the episode successfully, whereas collecting a wrong object or reaching the time limit does not. We multiply the original success reward by S and assign reward $-C$ to wrong-object terminations. The tested conditions are $(S, C) \in \{(1, 0), (1, 0.5), (2, 0), (2, 1)\}$; $(1, 0)$ is the original task. Each method-condition setting uses 20 seeds. We report success rate rather than the modified training return so that the performance metric is comparable across reward transformations. Table 7 gives final success and the time-averaged success-rate curve (success AUC). FT-GAE has higher reported means in all four conditions. Adding a failure penalty sharply lowers truncated GAE's final success at both values of S, while FT-GAE's final success changes less; the relative gaps are largest in the two penalized conditions.

Table 4: Hopper-v4 terminal-penalty intervention with reward normalization and estimator-specific λ values (truncated GAE: 0.99; FT-GAE: 1.00). Evaluation uses the unmodified reward. Results are means over 40 seeds; Δ is the relative FT-GAE change.

	Final return			Return AUC		
C	Truncated	FT-GAE	Δ	Truncated	FT-GAE	Δ
0	1488.9	1524.8	+2.4%	1024.7	1084.7	+5.9%
5	1284.4	1335.7	+4.0%	907.1	956.8	+5.5%
100	1294.1	1473.4	+13.9%	897.1	1038.8	+15.8%

Table 5: Hopper-v4 terminal-penalty intervention with reward normalization and matched $\lambda = 0.99$. Evaluation uses the unmodified reward. Results are means over 40 seeds; Δ is the relative FT-GAE change.

	Final return			Return AUC		
C	Truncated	FT-GAE	Δ	Truncated	FT-GAE	Δ
0	1488.9	1252.4	−15.9%	1024.7	942.7	−8.0%
5	1284.4	1382.8	+7.7%	907.1	967.4	+6.6%
100	1294.1	1445.0	+11.7%	897.1	1045.9	+16.6%

Table 6: Hopper-v4 terminal-penalty intervention without reward normalization, with matched $\lambda = 0.99$ and no hyperparameter retuning. Evaluation uses the unmodified reward. Results are means over 40 seeds; Δ is the relative FT-GAE change.

	Final return			Return AUC		
C	Truncated	FT-GAE	Δ	Truncated	FT-GAE	Δ
0	1480.6	1694.9	+14.5%	959.4	1143.2	+19.2%
5	1347.5	1732.7	+28.6%	904.0	1124.2	+24.4%
100	1428.6	1821.1	+27.5%	919.5	1148.7	+24.9%

Table 7: MiniGrid-Fetch-5x5-N2 terminal-reward intervention. S multiplies the success reward and $-C$ is assigned to wrong-object termination. Results are mean percentages over 20 seeds; Δ is the relative FT-GAE change.

	Final success			Success AUC		
(S, C)	Truncated	FT-GAE	Δ	Truncated	FT-GAE	Δ
$(1, 0)$	83.8%	94.1%	+12.2%	69.3%	84.3%	+21.7%
$(1, 0.5)$	49.1%	90.5%	+84.4%	49.1%	77.1%	+57.2%
$(2, 0)$	81.9%	94.6%	+15.5%	67.7%	85.4%	+26.2%
$(2, 1)$	49.9%	92.3%	+85.0%	49.6%	75.8%	+52.8%

E.3 HYPERPARAMETERS

This section reports all hyperparameters, training budgets, evaluation frequencies, and moving-average window sizes used to plot the learning curves.

Table 8: Hyperparameters used for the Atari and Box2D PPO experiments. All Atari environments use the same SB3-Zoo Atari configuration with `terminal_on_life_loss=False`, while BipedalWalker-v3 and LunarLander-v3 use their environment-specific SB3-Zoo configurations. Standard PPO and FT-GAE use identical optimization, rollout, network, and evaluation settings; they differ only in the advantage estimator used inside the rollout buffer.

Hyperparameter	Atari	BipedalWalker-v3	LunarLander-v3
Training / rollout			
env_wrapper	`AtariWrapper`	–	–
frame_stack	4	–	–
n_timesteps	2.5×10^{7}	5×10^{6}	10^{6}
normalize	`False`	`True`	`False`
n_envs	8	32	16
n_steps	128	2048	1024
batch_size	256	64	64
GAE / environment			
gamma	0.99	0.999	0.999
gae_lambda	0.95	0.95	0.98
normalize_advantage	`True`	`True`	`True`
Optimization			
learning_rate	`lin_2.5e-4`	3.0×10^{-4}	3.0×10^{-4}
n_epochs	4	10	4
max_grad_norm	0.5	0.5	0.5
PPO objective / regularization			
clip_range	`lin_0.1`	0.18	0.2
clip_range_vf	`None`	`None`	`None`
ent_coef	0.01	0.0	0.01
vf_coef	0.5	0.5	0.5
target_kl	`None`	`None`	`None`
Network			
policy	`CnnPolicy`	`MlpPolicy`	`MlpPolicy`
net_arch	`default`	`default`	`default`
log_std_init	`n/a`	`default`	`n/a`
ortho_init	`default`	`default`	`default`
activation_fn	`default`	`default`	`default`
Evaluation			
evaluation frequency	100000	10000	10000
moving window for plots	5	10	3

Table 9: Hyperparameters used for the Jaxnasium MinAtar PPO and FT-GAE baseline experiments. The parameters are based on gymnax baselines Lange (2026). All five MinAtar environments use the same settings. PPO and FT-GAE differ only in the advantage estimator.

Hyperparameter	All MinAtar environments
Training / rollout	
n_timesteps	$10{,}002{,}432$
n_envs	64
n_steps	128
n_minibatches	8
batch_size	1024
normalize_observations	`False`
normalize_rewards	`False`
GAE	
gamma	0.999
gae_lambda	0.95
normalize_advantage	`True` (per minibatch)
Optimization	
optimizer	Adam
learning_rate	linear: $5\times10^{-4} \rightarrow 0$
n_epochs	4
max_grad_norm	0.5
PPO objective / regularization	
clip_range	0.2
clip_range_vf	10.0
ent_coef	0.01
vf_coef	0.5
Network / evaluation	
policy	separate actor and critic MLPs
net_arch	$(256, 256)$ each
ortho_init	`True`; zero biases
activation_fn	ReLU
Evaluation	
evaluation frequency	131,072
moving window for plots	1

Table 10: Hyperparameters used for the MiniGrid PPO experiments. All MiniGrid environments use the SB3-Zoo MiniGrid configuration with `minigrid.wrappers.FlatObsWrapper`. The reported hyperparameters are used for the `v0` version of each environment throughout all experiments. Standard PPO and FT-GAE use identical optimization, rollout, network, and evaluation settings; they differ only in the advantage estimator used inside the rollout buffer. .

Hyperparameter	DoorKey	Empty-Rand-5x5	Empty-Rand-6x6	Fetch	FourRooms	GoToDoor
Training / rollout						
env_wrapper	FlatObsWrapper	FlatObsWrapper	FlatObsWrapper	FlatObsWrapper	FlatObsWrapper	FlatObsWrapper
n_timesteps	10^5	2×10^4	2×10^4	2×10^6	2×10^6	2×10^6
normalize	False	False	False	False	False	False
n_envs	8	8	8	8	8	8
n_steps	128	128	128	128	512	128
rollout_size	1024	1024	1024	1024	4096	1024
batch_size	64	64	64	64	64	64
GAE / environment						
gamma	0.99	0.99	0.99	0.99	0.99	0.99
gae_lambda	0.95	0.95	0.95	0.95	0.95	0.95
normalize_advantage	True	True	True	True	True	True
Optimization						
learning_rate	2.5×10^{-4}	2.5×10^{-4}	2.5×10^{-4}	2.5×10^{-4}	2.5×10^{-4}	2.5×10^{-4}
n_epochs	10	10	10	10	10	10
max_grad_norm	0.5	0.5	0.5	0.5	0.5	0.5
PPO objective / regularization						
clip_range	0.2	0.2	0.2	0.2	0.2	0.2
clip_range_vf	None	None	None	None	None	None
ent_coef	0.0	0.0	0.0	0.0	0.0	0.0
vf_coef	0.5	0.5	0.5	0.5	0.5	0.5
target_kl	None	None	None	None	None	None
Network						
policy	MlpPolicy	MlpPolicy	MlpPolicy	MlpPolicy	MlpPolicy	MlpPolicy
net_arch	default	default	default	default	default	default
log_std_init	n/a	n/a	n/a	n/a	n/a	n/a
ortho_init	default	default	default	default	default	default
activation_fn	default	default	default	default	default	default
Evaluation						
evaluation frequency	1000	500	500	10000	10000	10000
moving window for plots	3	2	2	5	5	5

Table 11: Hyperparameters used for the MuJoCo PPO experiments. Standard PPO and FT-GAE use identical optimization, rollout, network, and evaluation settings; they differ only in the advantage estimator used inside the rollout buffer.

Hyperparameter	Swimmer-v4	Hopper-v4	Ant-v4	HalfCheetah-v4	Humanoid-v4	Walker2d-v4
Training / rollout						
n_timesteps	10^6	10^6	10^7	10^6	10^7	10^6
normalize	True	True	True	True	True	True
n_envs	4	4	4	4	4	4
n_steps	1024	512	512	512	512	512
rollout_size	4096	2048	2048	2048	2048	2048
batch_size	256	32	32	64	256	32
GAE / environment						
gamma	0.9999	0.999	0.98	0.98	0.95	0.99
gae_lambda	0.98	0.99	0.8	0.92	0.9	0.95
normalize_advantage	True	True	True	True	True	True
Optimization						
learning_rate	6.0×10^{-4}	9.80828×10^{-5}	1.90609×10^{-5}	2.0633×10^{-5}	3.56987×10^{-5}	5.05041×10^{-5}
n_epochs	10	5	10	20	5	20
max_grad_norm	0.5	0.7	0.6	0.8	2.0	1.0
PPO objective / regularization						
clip_range	0.2	0.2	0.1	0.1	0.3	0.1
clip_range_vf	None	None	None	None	None	None
ent_coef	0.0	0.00229519	4.9646×10^{-7}	0.000401762	0.00238306	0.000585045
vf_coef	0.5	0.835671	0.677239	0.58096	0.431892	0.871923
target_kl	None	None	None	None	None	None
Network						
policy	MlpPolicy	MlpPolicy	MlpPolicy	MlpPolicy	MlpPolicy	MlpPolicy
net_arch	default	[256,256]	default	[256,256]	[256,256]	default
log_std_init	default	-2	default	-2	-2	default
ortho_init	default	False	default	False	False	default
activation_fn	default	default	default	default	default	default
Evaluation						
evaluation frequency	10000	10000	100000	10000	100000	10000
moving window for plots	2	2	2	2	2	2

E.4 NORMALIZATION

For the robust aggregate metrics, scores are normalized separately for each environment and for each metric before applying the rliable aggregators. Let $x^{(m)}_{e,a,s}$ denote the value of metric m for environment e, algorithm a, and seed s. In our implementation, the environment-specific normalization constants are computed from the observed seed-level scores of both compared methods,

$$x^{(m)}_{\min,e} = \min_{a,s} x^{(m)}_{e,a,s}, \qquad x^{(m)}_{\max,e} = \max_{a,s} x^{(m)}_{e,a,s}.$$

For return-based metrics, where larger values are better, the normalized score is

$$\tilde{x}^{(m)}_{e,a,s} = \frac{x^{(m)}_{e,a,s} - x^{(m)}_{\min,e}}{x^{(m)}_{\max,e} - x^{(m)}_{\min,e}}.$$

Thus, within each environment and metric, the worst observed seed-level score among the compared methods is mapped to 0, and the best observed score is mapped to 1. If all observed values for an environment are identical, the denominator is set to one to avoid numerical degeneracy.

The rliable plots and tables are then computed from matrices of normalized scores with shape *seeds* $\times$ *environments*. We report the IQM, median, and mean over these normalized scores, together with bootstrap confidence intervals computed by rliable. Performance profiles are also based on the same normalized scores and evaluate the fraction of scores exceeding thresholds on the normalized scale.

This normalization is applied independently for each benchmark suite and experimental setting. Consequently, the normalized aggregates should be interpreted as relative performance within the corresponding suite and setting, not as absolute performance normalized against external reference scores.

E.5 DETAILED OUT OF THE BOX RESULTS

Table 12: Aggregated PPO results at the default λ settings. Box2D, MiniGrid, MinAtar and MuJoCo use 30 seeds per method and environment; Atari uses six seeds per method and environment. Scores are min-max normalized within each environment and metric before equal-environment aggregation. Entries are IQM or $\mathbb{P}[\mathrm{FT} > \mathrm{trunc}]$ with 95% stratified bootstrap intervals (5,000 resamples). Bold point estimates mark the higher IQM or an improvement probability whose unadjusted 95% interval lies entirely above 0.5.

Class	Metric	Truncated GAE	FT-GAE	$\Pr[\mathrm{FT} > \mathrm{trunc}]$
Box2D	Final return	0.73 [0.68, 0.77]	**0.91** [0.88, 0.93]	**0.80** [0.72, 0.88]
Box2D	AUC	0.45 [0.40, 0.50]	**0.69** [0.62, 0.74]	**0.72** [0.64, 0.81]
MinAtar	Final return	0.30 [0.28, 0.33]	**0.66** [0.62, 0.70]	**0.81** [0.77, 0.86]
MinAtar	AUC	0.25 [0.23, 0.28]	**0.65** [0.62, 0.69]	**0.86** [0.82, 0.90]
MiniGrid	Final return	0.65 [0.61, 0.69]	**0.83** [0.81, 0.86]	**0.69** [0.64, 0.74]
MiniGrid	AUC	0.59 [0.55, 0.63]	**0.78** [0.75, 0.80]	**0.69** [0.64, 0.75]
Atari	Final return	0.41 [0.31, 0.56]	**0.52** [0.38, 0.69]	0.57 [0.42, 0.71]
Atari	AUC	0.44 [0.31, 0.55]	**0.51** [0.42, 0.63]	0.54 [0.40, 0.69]
MuJoCo	Final return	0.50 [0.45, 0.55]	**0.51** [0.46, 0.57]	0.52 [0.46, 0.57]
MuJoCo	AUC	0.48 [0.44, 0.53]	**0.52** [0.47, 0.56]	0.55 [0.49, 0.61]

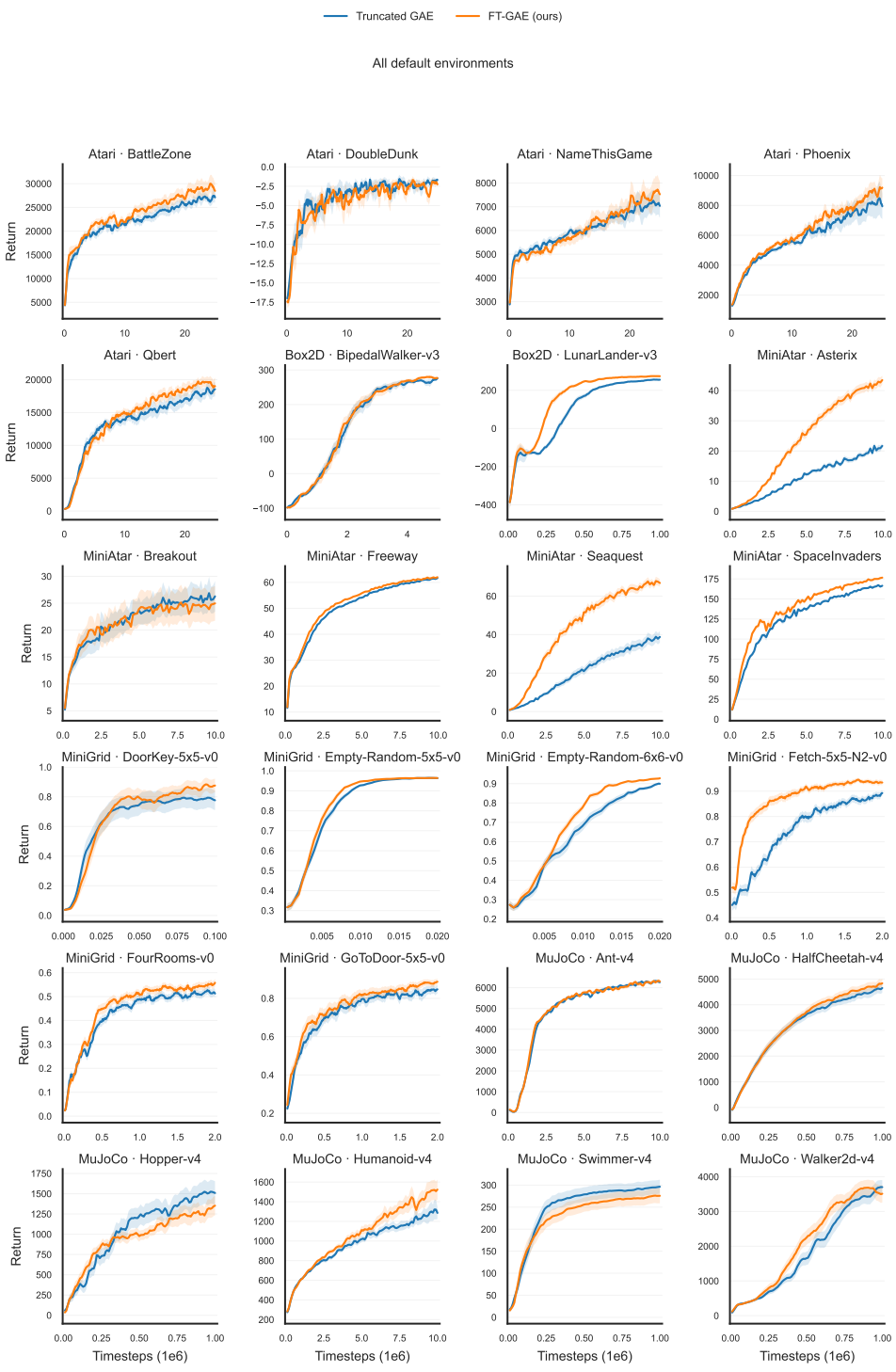
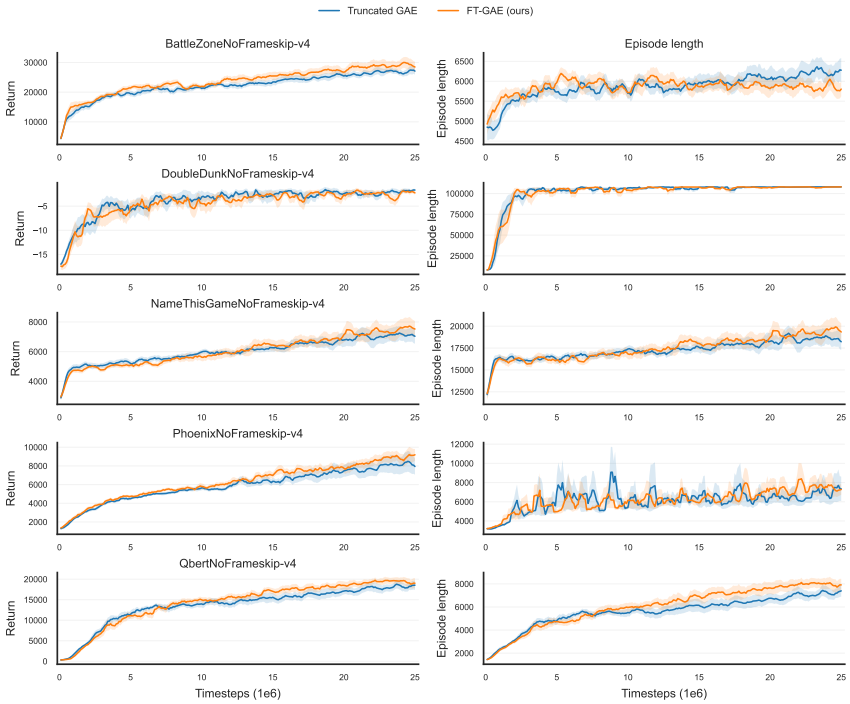
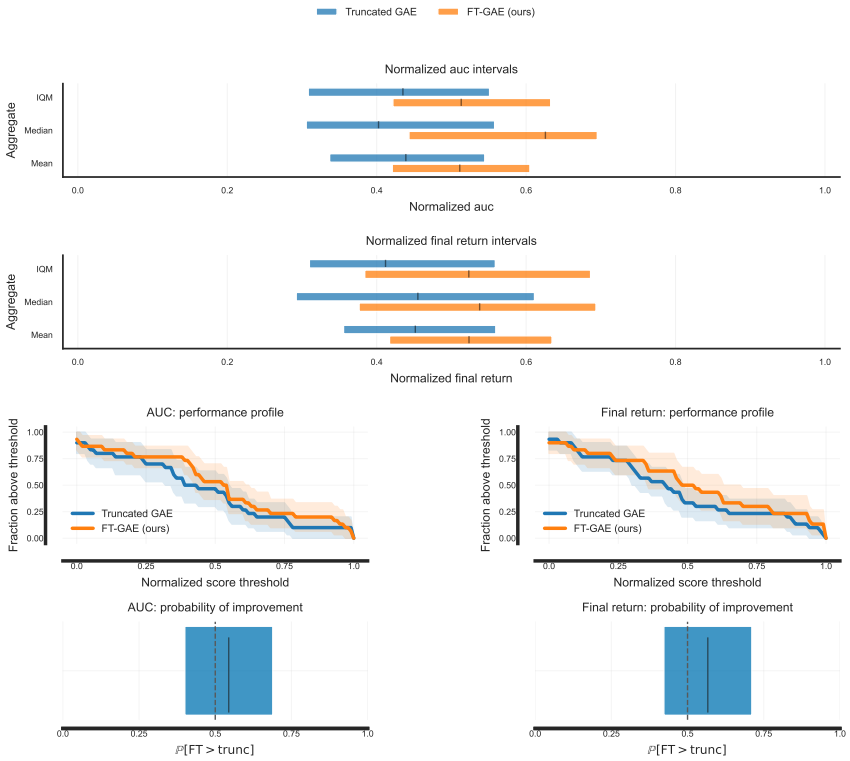


Figure 18: All learning curves for the default λ setting. Results show mean evaluation return over 30 seeds. Shaded regions indicate standard errors.

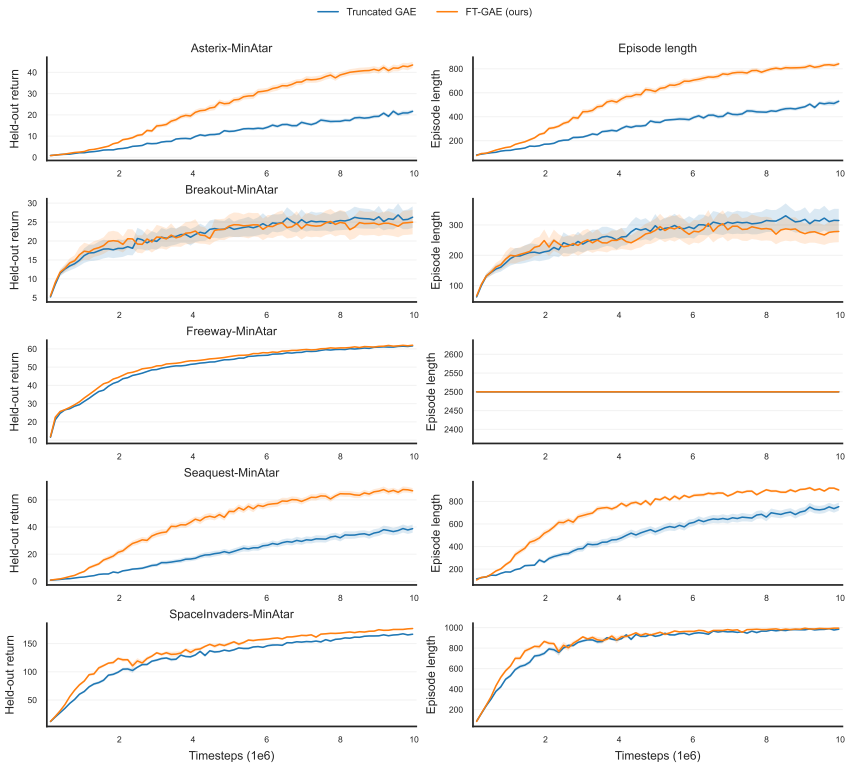


(a) Learning curves for Atari under the default λ setting. Lines show mean evaluation returns, shaded regions indicate standard errors.

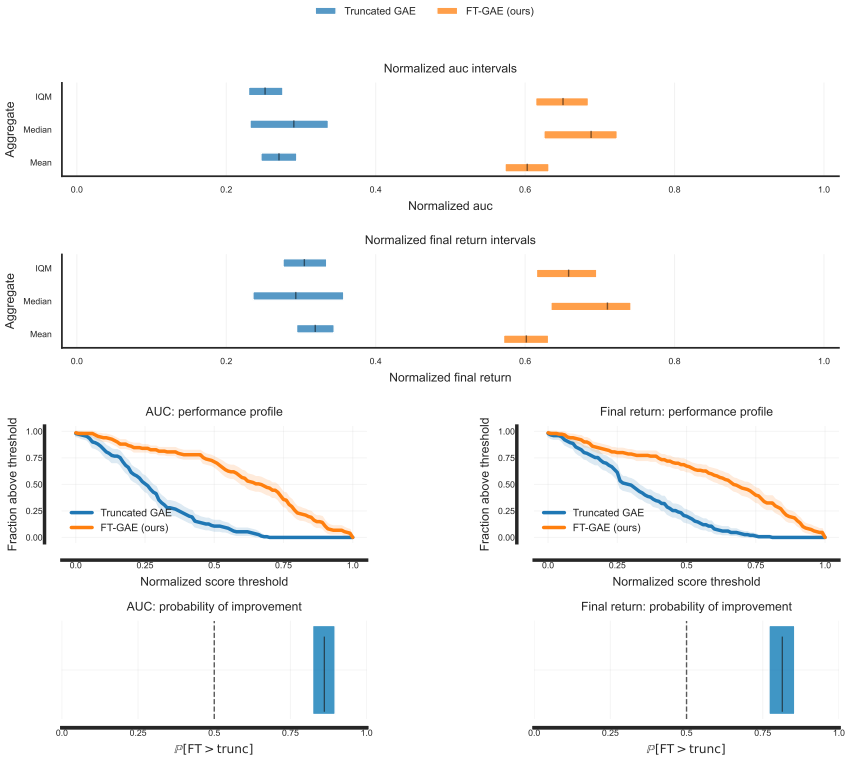


(b) Aggregate performance summary using `rliable` under the default λ setting.

Figure 19: Atari results for the default λ setting. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.

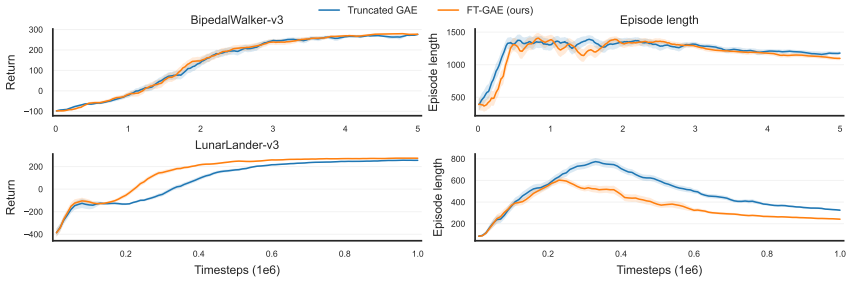


(a) Learning curves for MinAtar under the default λ setting. Lines show mean evaluation returns, shaded regions indicate standard errors.

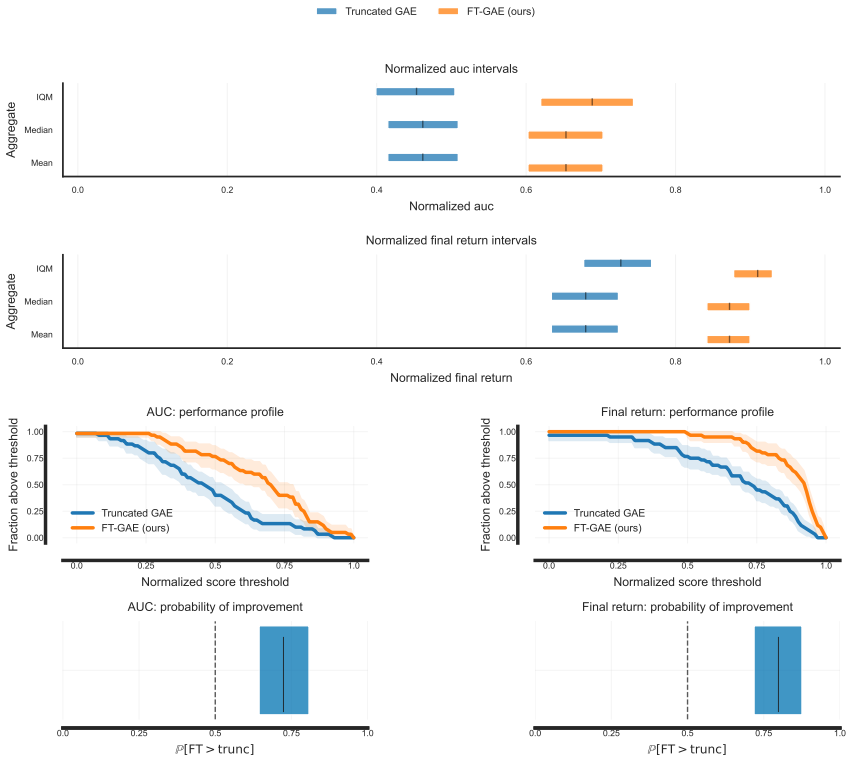


(b) Aggregate performance summary using `rliable` under the default λ setting.

Figure 20: MinAtar results for the default λ setting. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.

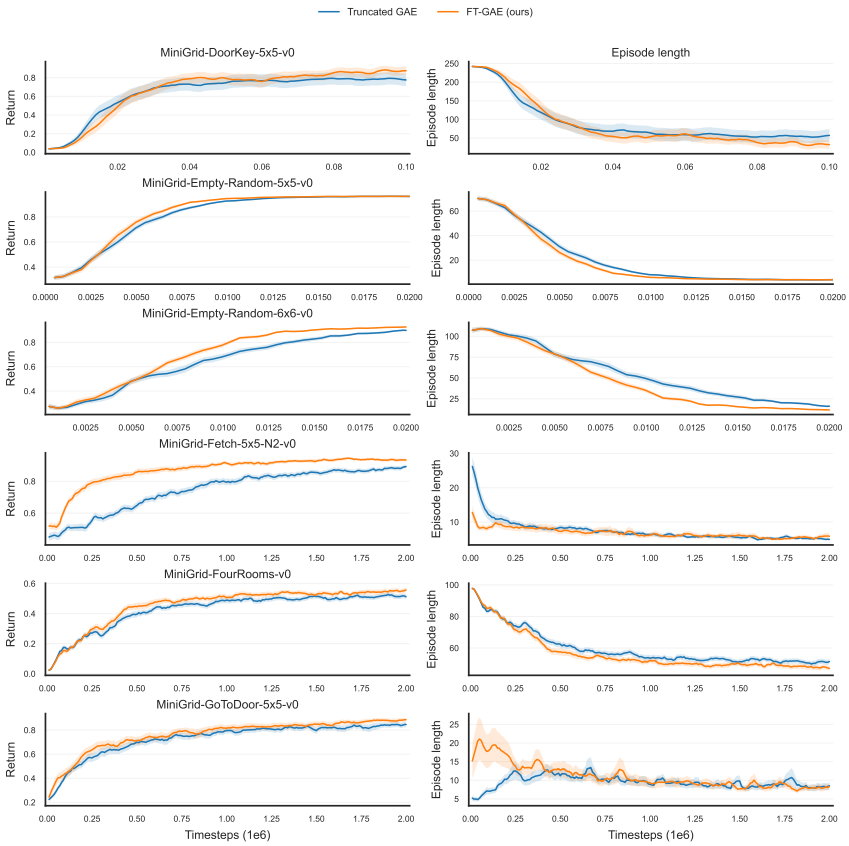


(a) Learning curves for tested Box2D environments under the default λ setting. Lines show mean evaluation returns, shaded regions indicate standard errors.

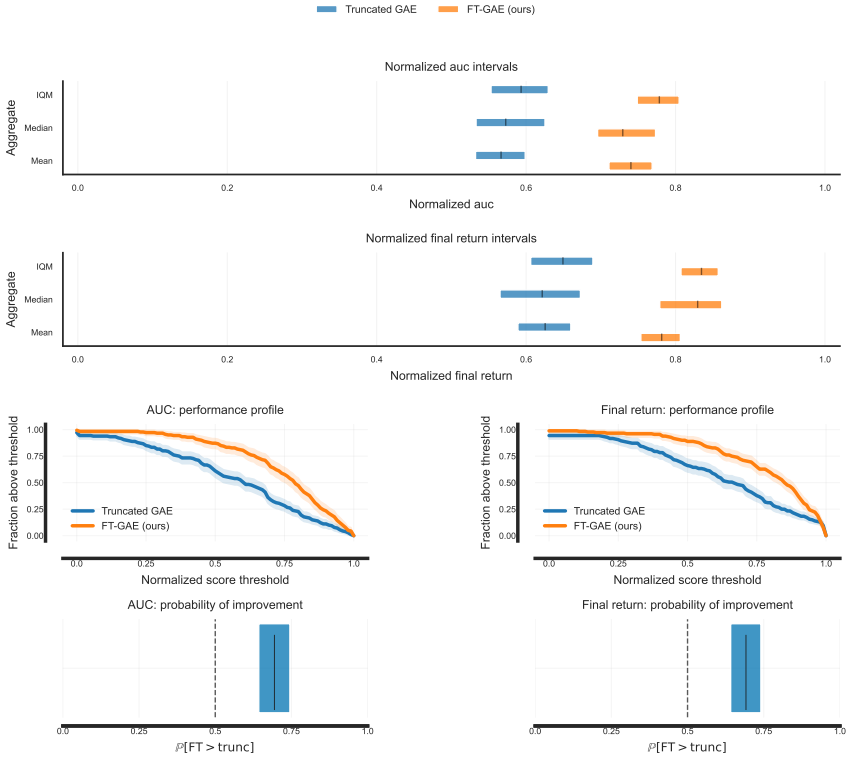


(b) Aggregate performance summary using `rliable` under the default λ setting.

Figure 21: Box2D results for the default λ setting. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.

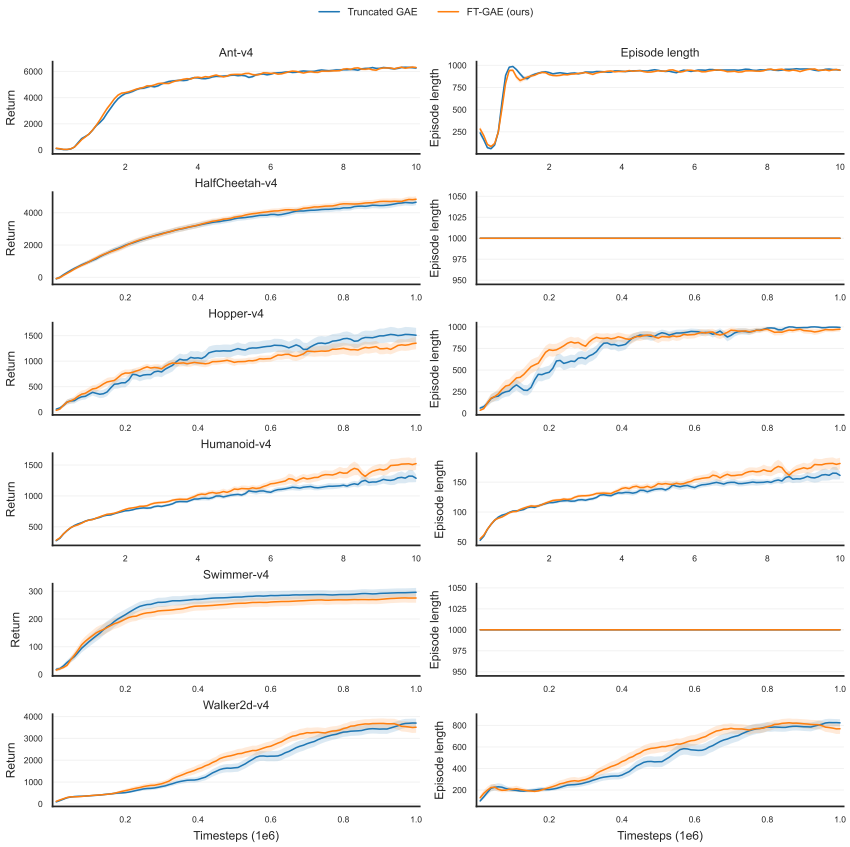


(a) Learning curves for Minigrids under the default λ setting. Lines show mean evaluation returns, shaded regions indicate standard errors.

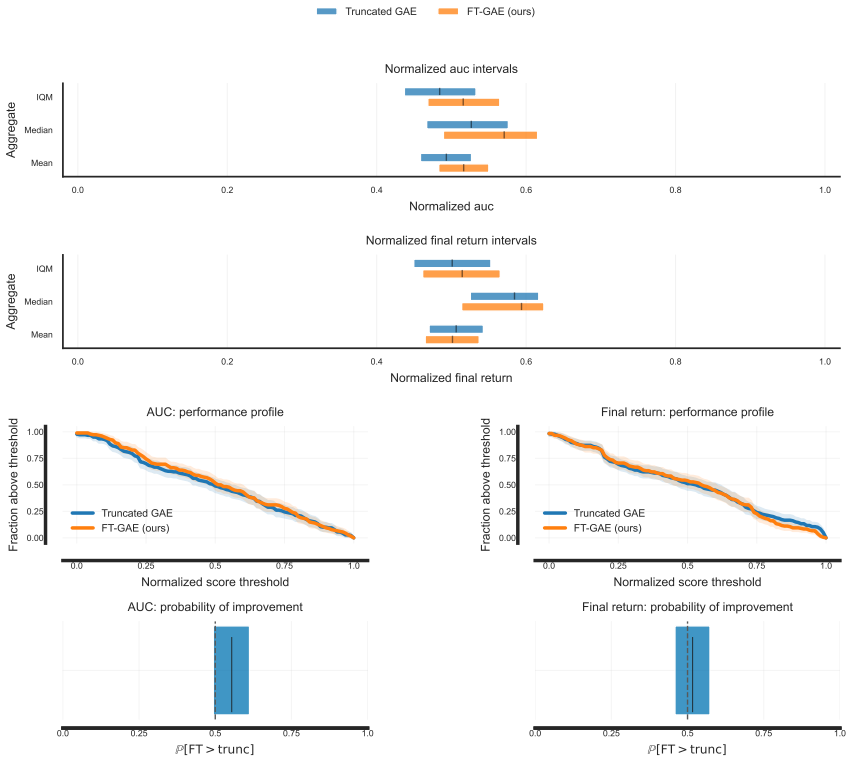


(b) Aggregate performance summary using `rliable` under the default λ setting.

Figure 22: Minigrids results for the default λ setting. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.



(a) Learning curves for MuJoCo under the default λ setting.



(b) Aggregate performance summary using `rliable` under the default λ setting.

Figure 23: MuJoCo results for the default λ setting. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.

E.6 λ-SWEEPS

The λ-sweep experiments test whether the empirical gains of FT-GAE can be explained by retuning the GAE parameter in standard truncated GAE. For each environment, we keep all PPO hyperparameters fixed to the corresponding default λ configuration used in the main default-HP experiments and vary only λ. We evaluate both estimators, truncated GAE and FT-GAE, on the same sweep grid. The tested λ values cover the high-λ regime commonly used with PPO, including $\lambda \in \{0.75, 0.775, 0.80, 0.825, 0.85, 0.875, 0.90, 0.925, 0.95, 0.975, 1.0\}$. For selected MuJoCo environments, including Ant-v4, HalfCheetah-v4, and Humanoid-v4, we additionally include lower values $\lambda \in \{0.65, 0.675, 0.70, 0.725\}$, since preliminary sweeps down to $\lambda = 0.75$ indicated possible performance gains at lower values. For the final sweep figures, we use exactly five seeds per plotted environment-method-λ configuration.

E.6.1 SWEEP RESULTS

For each seed and λ, we compute two metrics from the evaluation curves. Final return is the average return over the last 10% of evaluation checkpoints. AUC is the area under the evaluation-return curve, normalized by the training horizon, and measures sample efficiency over the full training run. For each environment, algorithm, and λ, we report the arithmetic mean of these seed-level metrics across seeds. Shaded regions show standard-error bands around the plotted mean.

Each figure contains one benchmark suite, with one row per environment. In each row, the left panel shows mean final performance as a function of λ, and the right panel shows mean AUC as a function of λ.

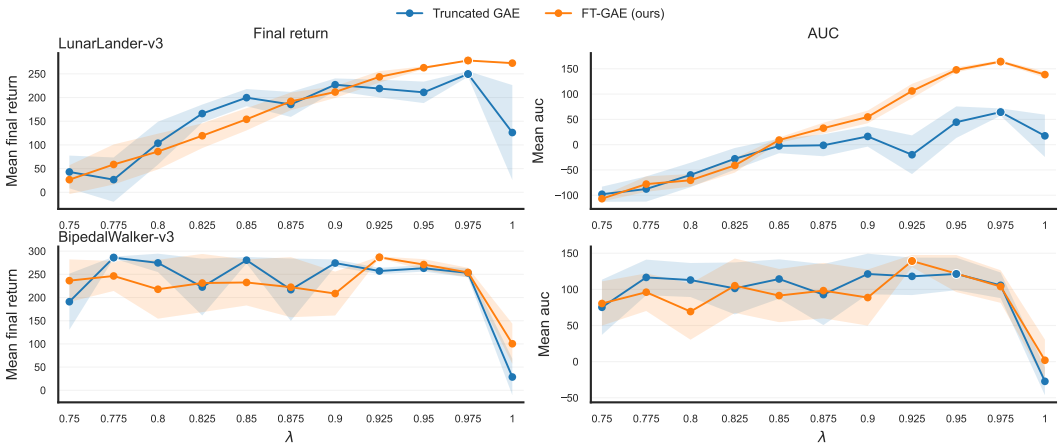


Figure 24: Box2D λ-sweep results. Each row shows one environment and compares truncated GAE with FT-GAE across GAE-λ values. Left panels report final return, computed as the average return over the last 10% of evaluation checkpoints. Right panels report AUC, normalized by the training horizon, as a sample-efficiency metric. Markers show arithmetic means across seeds and shaded regions show standard-errors.

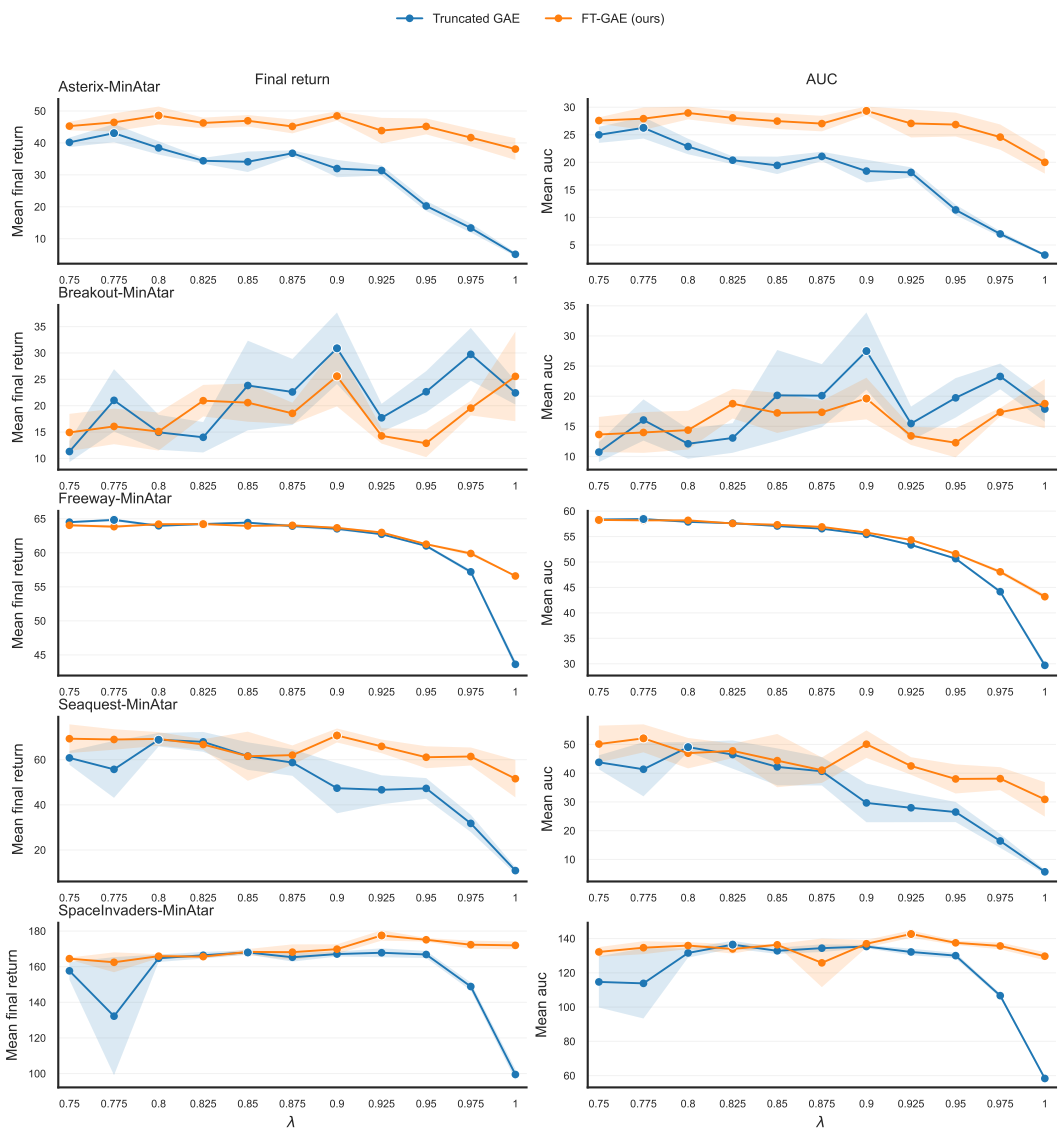


Figure 25: MinAtar λ-sweep results. Each row shows one environment and compares truncated GAE with FT-GAE across GAE-λ values. Left panels report final return, computed as the average return over the last 10% of evaluation checkpoints. Right panels report AUC, normalized by the training horizon, as a sample-efficiency metric. Markers show arithmetic means across seeds and shaded regions show standard-errors.

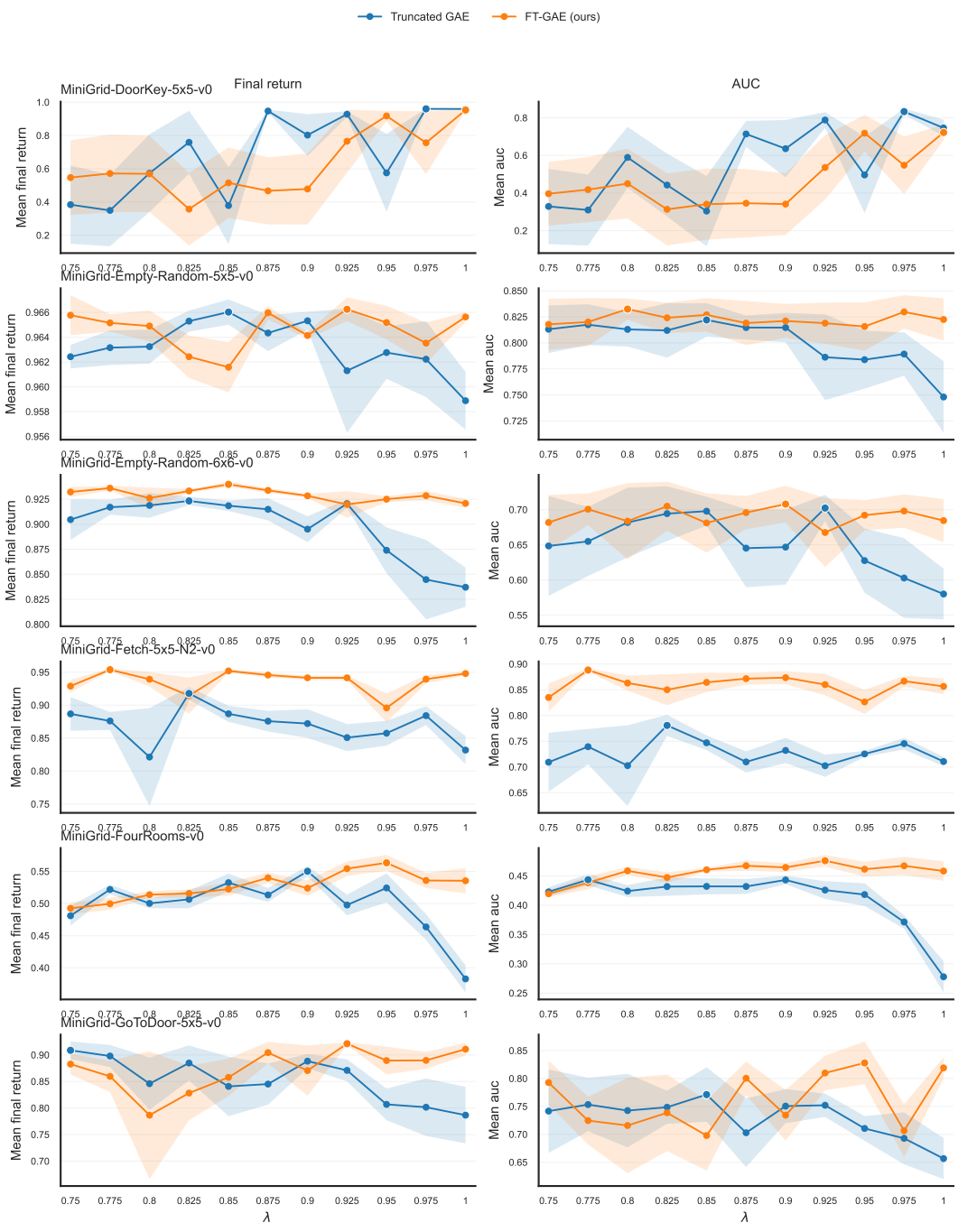


Figure 26: MiniGrid λ-sweep results. Each row shows one environment and compares truncated GAE with FT-GAE across GAE-λ values. Left panels report final return, computed as the average return over the last 10% of evaluation checkpoints. Right panels report AUC, normalized by the training horizon, as a sample-efficiency metric. Markers show arithmetic means across seeds and shaded regions show standard-errors.

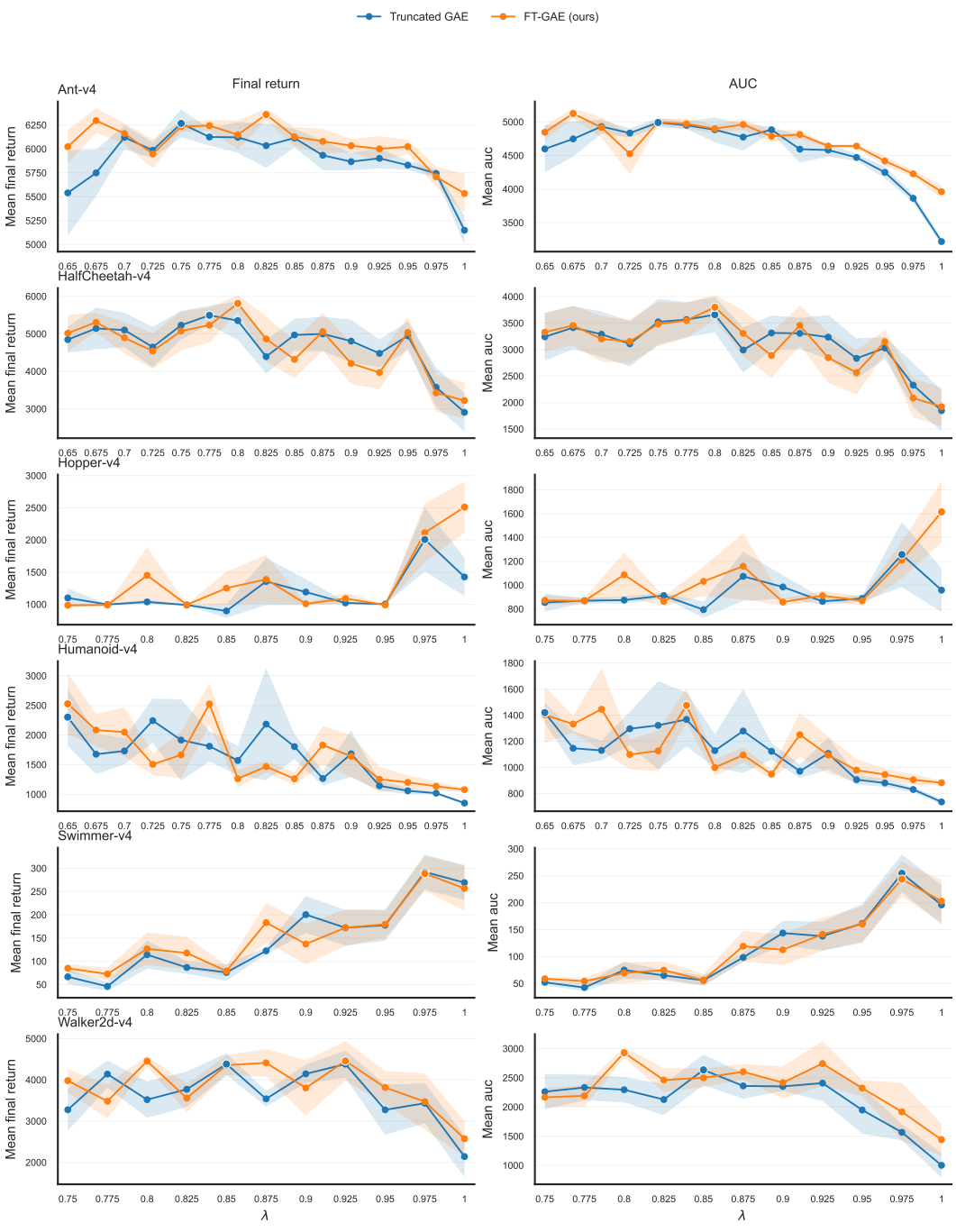
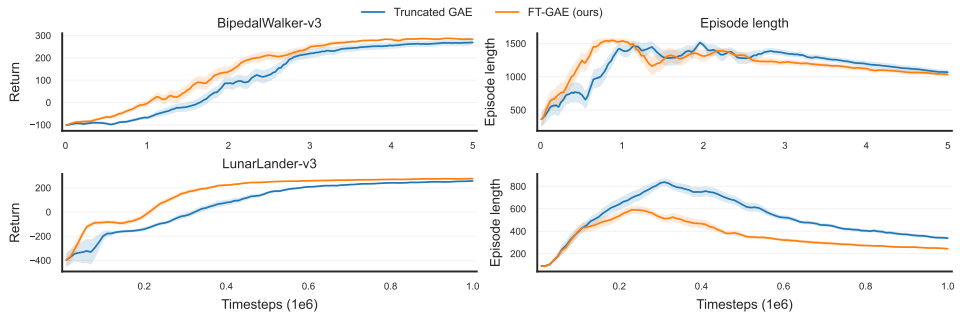


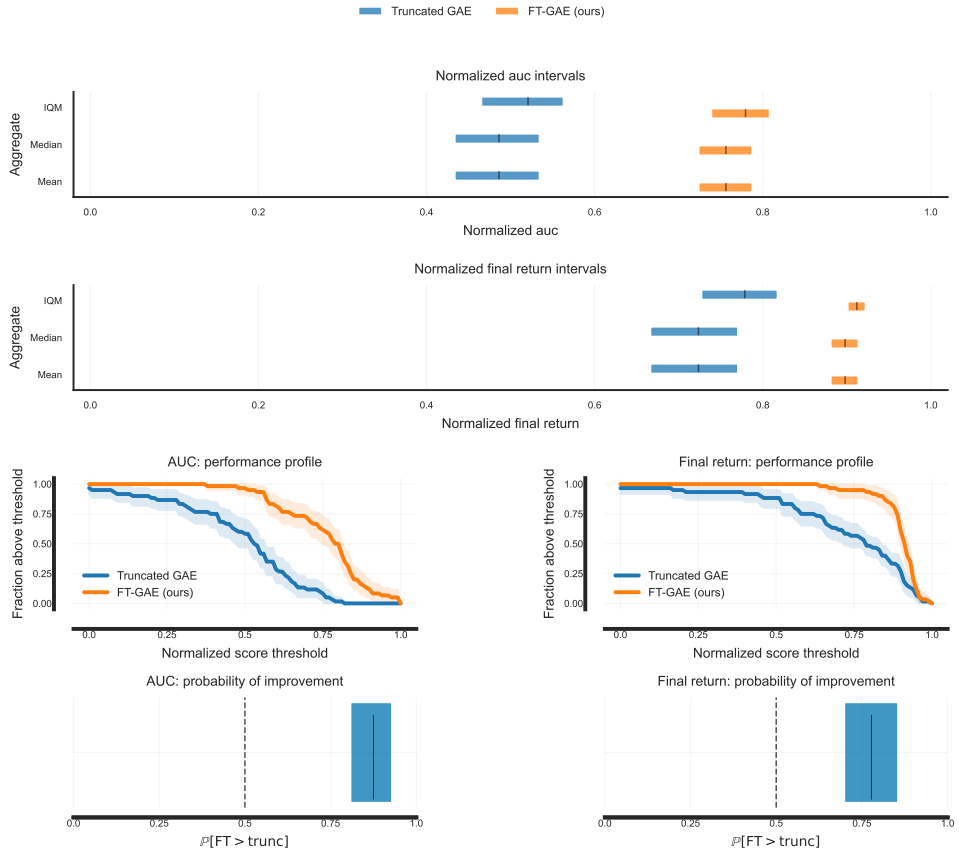
Figure 27: MuJoCo λ-sweep results. Each row shows one environment and compares truncated GAE with FT-GAE across GAE-λ values. Left panels report final return, computed as the average return over the last 10% of evaluation checkpoints. Right panels report AUC, normalized by the training horizon, as a sample-efficiency metric. Markers show arithmetic means across seeds and shaded regions show standard-errors.

E.6.2 RESULTS FOR BEST PERFORMING GAE PARAMETERS

This section summarizes a best-λ comparison based on the sweep experiments above. For each environment and estimator, we select the λ value with the highest mean final return across the sweeps and then compare truncated GAE and FT-GAE with 30 new seeds under these environment-wise best performing settings. This comparison is therefore more favorable to both methods than the default-HP evaluation. For the best performing λ-values we normalize scores per environment as described in Section E.4. Then we compute robust metrics. We report the same normalized final-return and AUC metrics as before, aggregated across environments.

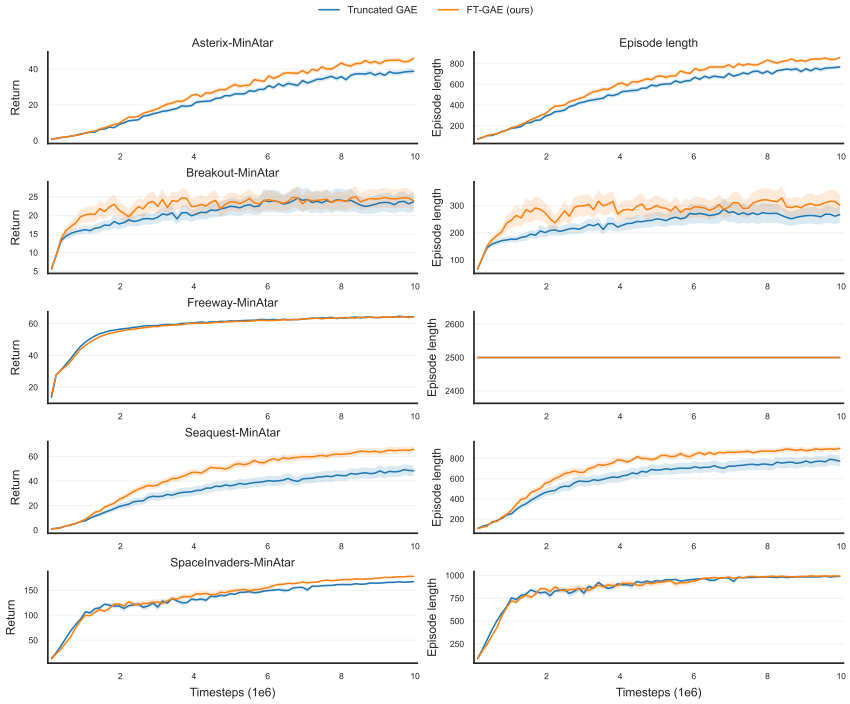


(a) Learning curves showing mean evaluation return over 30 seeds for Box2D under environment-wise best performing-λ values. Shaded regions indicate standard errors.

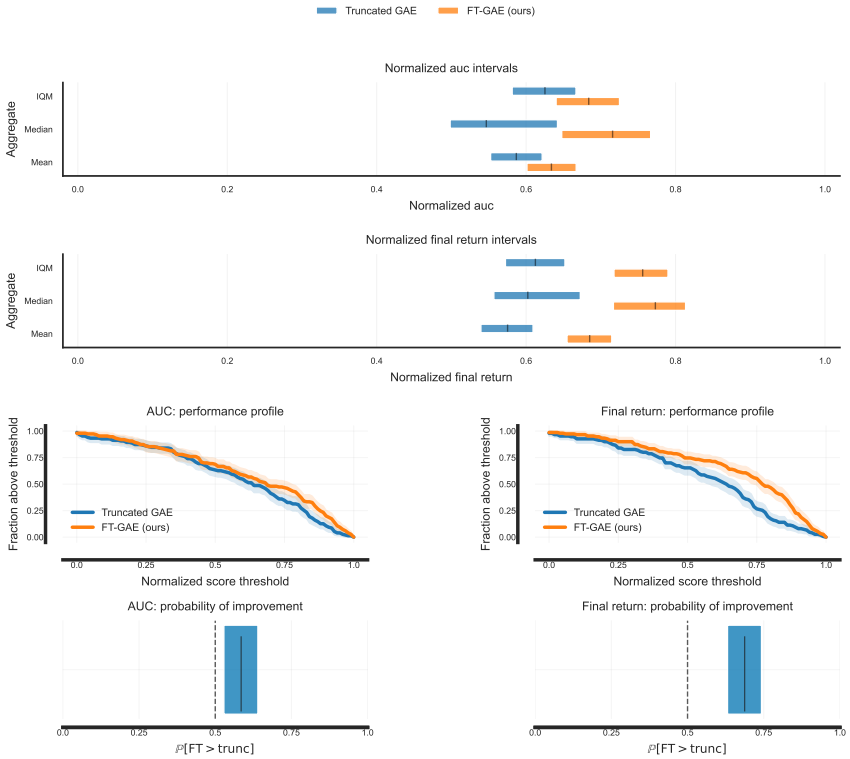


(b) Aggregate performance summary using `rliable` under environment-wise best performing-λ values.

Figure 28: Box2D results environment-wise tuned λ values. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.

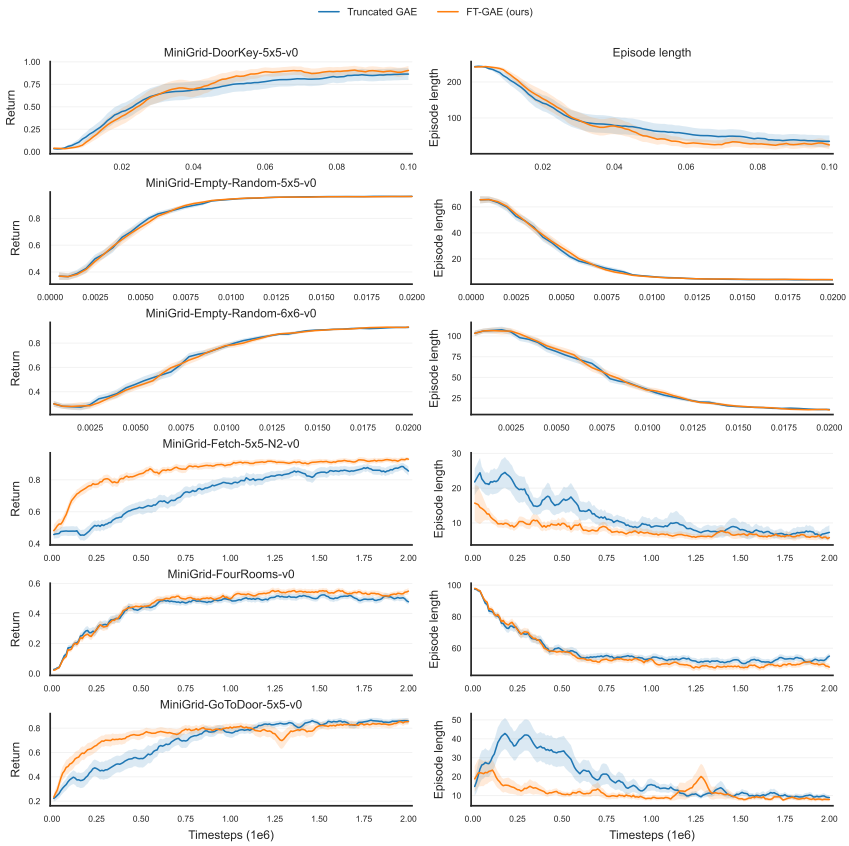


(a) Learning curves showing mean evaluation return over 30 seeds for MinAtar under environment-wise best performing-λ values. Shaded regions indicate standard errors.

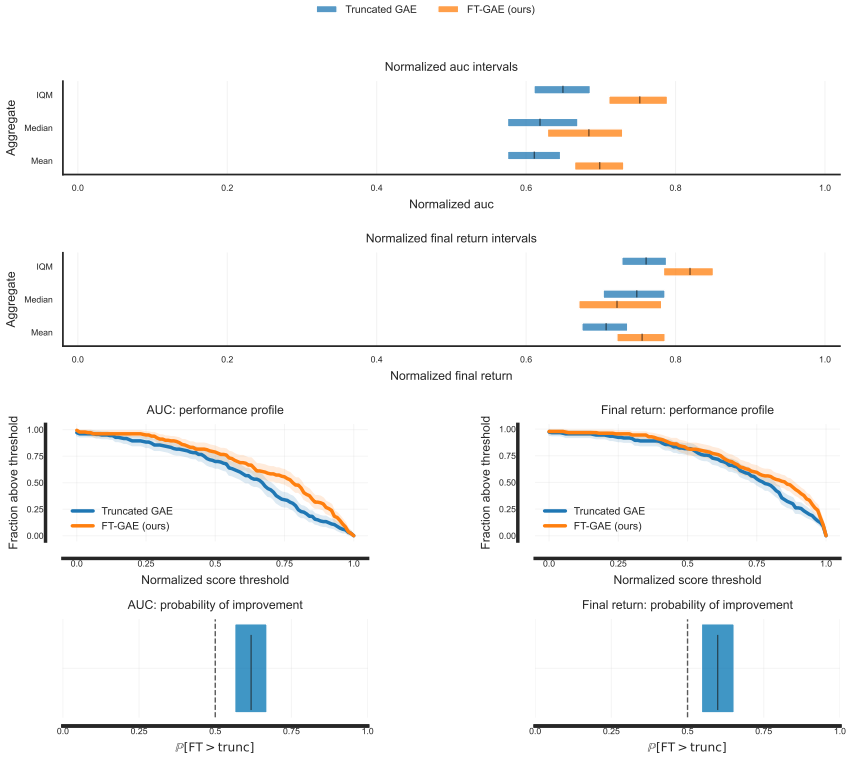


(b) Aggregate performance summary using `rliable` under environment-wise best performing-λ values.

Figure 29: MinAtar results environment-wise tuned λ values. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.

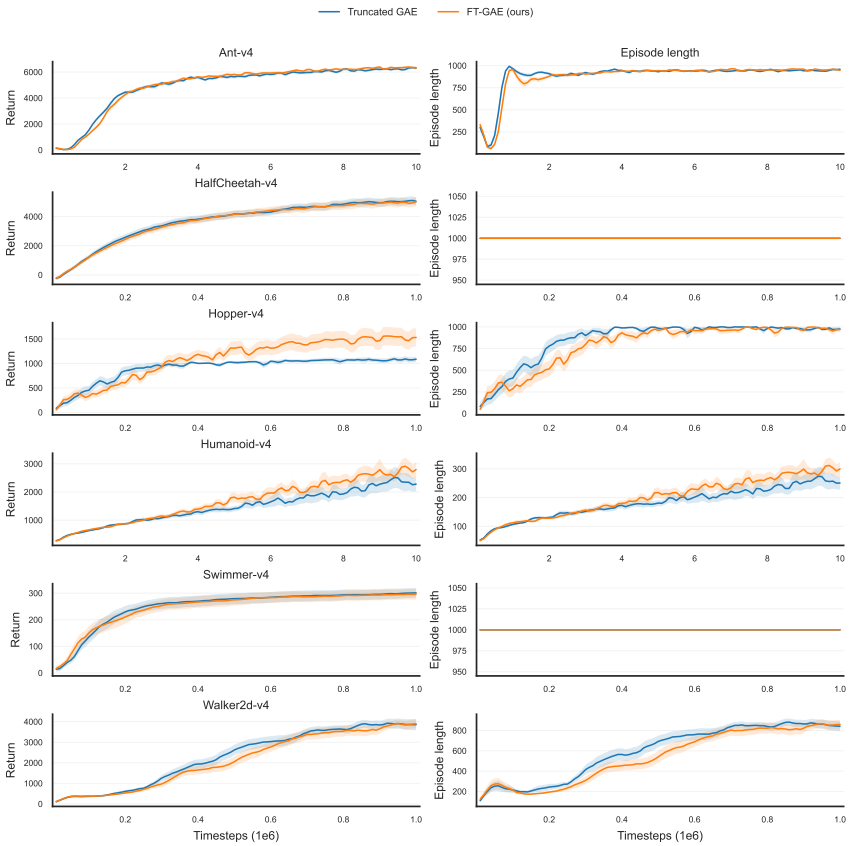


(a) Learning curves showing mean evaluation return over 30 seeds for Minigrid under environment-wise best performing-λ values. Shaded regions indicate standard errors.

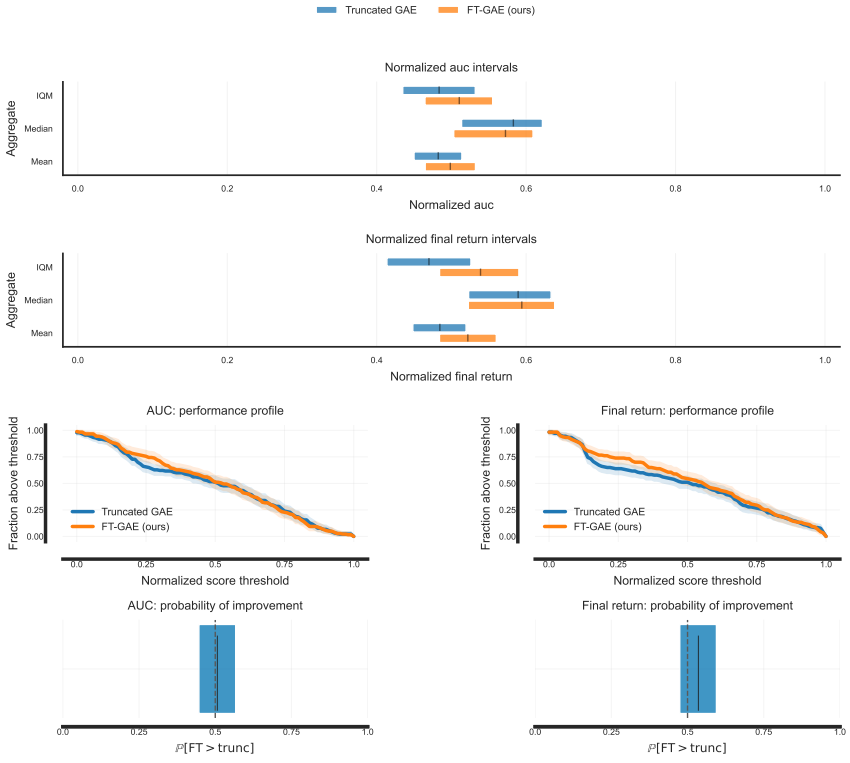


(b) Aggregate performance summary using `rliable` under environment-wise best performing-λ values.

Figure 30: Minigrids results environment-wise tuned λ values. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.



(a) Learning curves showing mean evaluation return over 30 seeds for MuJoCo under environment-wise best performing-λ values. Shaded regions indicate standard errors.



(b) Aggregate performance summary using `rliable` under environment-wise best performing-λ values.

Figure 31: MuJoCo results environment-wise tuned λ values. The top panel shows the learning curves, while the bottom panel reports the corresponding aggregate robust `rliable` summary.