TRUST REGIONS SELL, BUT WHO'S BUYING? GENERALIZED POWER-LAW SURROGATES FOR CLIPPED AND KL-CONSTRAINED REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Standard proximal (e.g., PPO) and trust-region (e.g., TRPO) methods stabilize reinforcement learning but optimize linear importance-ratio surrogates that do not account for the curvature of the surrogate under large likelihood-ratio displacements. We introduce Symphony, a generalized family of RL objectives derived from the n-th root likelihood ratio, $(\pi'/\pi)^{1/n}$. By isolating higher-order geometric contributions via a binomial expansion, Symphony unifies computationally efficient first-order clipped surrogates with KL-constrained natural gradient updates. We analyze its local geometry, showing that this power-law transformation preserves the Fisher Information Metric direction at the origin while damping ratio-driven gradient growth away from it. Evaluations across the MJX Continuous Control benchmark show substantial gains in sample efficiency and asymptotic stability, alongside competitive, environment-dependent Atari performance on visual control tasks in JAXAtari. Notably, under extended-horizon evaluations on Humanoid (two billion transitions under a shared configuration), clipped Symphony demonstrates positive gains over PPO. In a controlled 10M-step Humanoid evaluation within our common implementation, where PPO and SPO configurations were independently selected by mean screening AUC, Symphony achieves higher final returns than both PPO and SPO under each baseline's selected settings, while demonstrating decisive sample efficiency gains over SPO. Experiments across varying geometric orders and clipping ranges confirm that Symphony provides resilience across relaxed clipping boundaries ($\epsilon = 0.4$) and ratio drift, offering a curvature-aware foundation for scalable policy optimization.

1 INTRODUCTION

While on-policy reinforcement learning avoids the divergence hazards of the "Deadly Triad" (Sutton & Barto, 2018) and naturally supports massive parallelization, stabilizing policy optimization remains fundamentally challenging because standard methods optimize linear importance-ratio surrogates that do not account for the curvature of the surrogate under large likelihood-ratio displacements. In multi-dimensional continuous control, this absence of surrogate curvature becomes acute due to the *dimensionality curse of importance sampling*: because the joint likelihood ratio factors as a product across action dimensions, $r(\theta) = \prod_{i=1}^{d} \frac{\pi_\theta(a_i|s)}{\pi_{\text{old}}(a_i|s)}$, even modest per-dimension shifts compound exponentially into extreme ratio outliers. When paired with uncurved linear surrogates, these ratio excursions induce heavy-tailed, explosive parameter updates that shat-

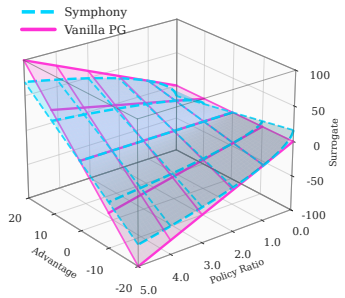
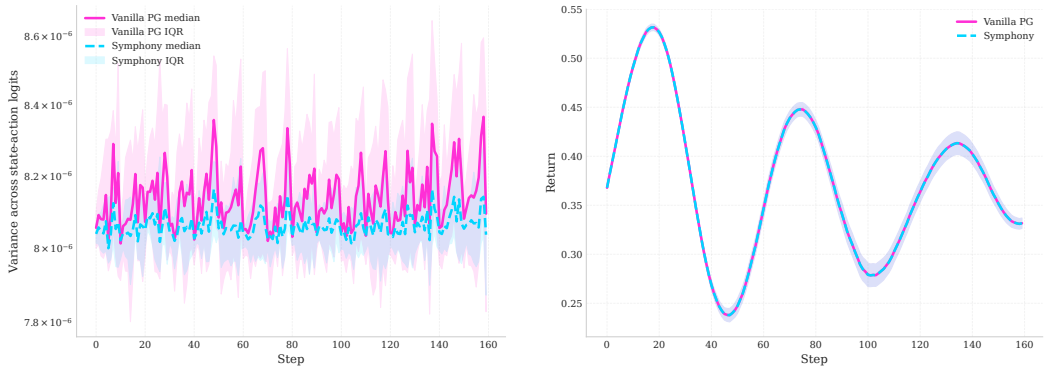


Figure 1: **Surrogate geometry.** Linear surrogate (rA^π, pink) vs. Symphony ($f_2(r)A^\pi$, blue). Tangents match at $r = 1$; Symphony adds negative curvature to dampen ratio spikes.

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.



(a) **Gradient dispersion across logits.** Median dispersion across 32 seeds (IQR shaded; log scale).

(b) **Expected return.** Mean return with 95% CI under identical drift and noisy advantages.

Figure 2: **Brownian multi-state bandit diagnostic** (Appendix C; not an MJX or JAX Atari task). Both methods receive identical drift and noisy advantages. (a) Within-seed logit-gradient dispersion across steps; (b) Expected return. Symphony maintains tighter dispersion while preserving comparable returns, isolating ratio compression from credit assignment.

ter policy stability (Garg et al., 2021)—a vulnerability that directly motivates the curvature-aware, sublinear power-law surrogate transformations introduced in this work.

Historically, reinforcement learning addresses this instability through two main paradigms: analytical trust regions or lighter first-order surrogates.

Analytical Trust Regions. Natural gradients recognize that small parameter shifts can produce large changes in policy distributions. Amari (1998) formalized this via information geometry, using the Fisher Information Matrix as the Riemannian metric on probability distributions. Natural actor-critic methods (Kakade, 2001; Peters & Schaal, 2008) then scale policy updates by the manifold's geometry rather than raw parameter distances.

TRPO made this principle practical for deep policy optimization. Schulman et al. (2015) constrained policy updates using KL divergence (Kullback & Leibler, 1951), resulting in a conservative policy-improvement procedure with monotonic-improvement guarantees closely related to Minorization-Maximization (Lange, 2016). Subsequent work explored richer geometries beyond the standard KL ball: f-divergence regularization for off-policy stability (Touati et al., 2020), forward and reverse KL greedification operators (Chan et al., 2022), and Wasserstein policy optimization that respects the topology of the action space (Zhang et al., 2018). These methods share the same goal: control policy improvement by controlling distributional motion. Their cost is that the geometry is often imposed externally through second-order estimates, constrained optimization, or explicit divergence penalties.

First-Order Approximations. PPO was introduced as a scalable alternative to explicit trust-region optimization. Instead of solving a KL-constrained problem, Schulman et al. (2017) proposed a clipped likelihood-ratio surrogate that uses only first-order gradients while retaining much of TRPO's empirical stability. This simplicity made PPO a default choice in continuous control (Duan et al., 2016; Heess et al., 2017), distributed reinforcement learning (Espeholt et al., 2018), and RLHF for large language models (Ouyang et al., 2022). PPO-style objectives have also influenced recent large-scale reasoning systems through variants such as GRPO (Shao et al., 2024).

However, PPO's stability is not purely a consequence of its clipped surrogate. Engstrom et al. (2020) showed that implementation details, such as reward normalization, advantage normalization, and value-function coupling, explain a large part of the empirical performance of PPO. Andrychowicz et al. (2021) reached a similar conclusion in a large-scale empirical study, showing that auxiliary choices and hyperparameters often dominate final performance. More recently, Simple Policy Optimization (SPO; Xie et al., 2025) has sought to simplify training by removing both the constraint and the clipping mechanism in favor of a penalized first-order surrogate. Beyond linear first-order surrogates, alternative paradigms explore higher-order trajectory expansions of ratio deviations (TayPO; Tang et al., 2020) or meta-discovered objectives grounded in mirror learning (DPO; Lu et al., 2022). Ultimately, standard first-order clipped methods remain scalable, but clipping acts only as a discon-

tinuous, post-hoc threshold: before clipping intervenes, the underlying linear surrogate still scales updates directly with extreme ratio excursions, exposing optimization to fat-tailed gradient variance in high dimensions.

The Symphony Objective. Symphony preserves the first-order scalability and local update direction of standard policy gradients while taming heavy-tailed ratio explosions at the objective level. Standard policy gradients scale proportionally with the likelihood ratio r, causing gradient second moments to grow quadratically as $\mathcal{O}(r^2)$. In continuous control with Gaussian policies over many action dimensions ($d = 17$ in Humanoid), this quadratic dependence amplifies product-ratio outliers into destructive gradient shocks. Symphony instead applies a concave, sublinear power-law transformation to the likelihood ratio: $f_n(r) = nr^{1/n} - (n-1)$ for integer order $n \geq 2$. Because $f_n'(1) = 1$, Symphony preserves the exact policy-gradient direction at the sampling distribution, but bends the surrogate sublinearly away from it. For our practical default $n = 2$, this dampens the dominant second-moment gradient scaling from quadratic $\mathcal{O}(r^2)$ to linear $\mathcal{O}(r)$.

Figure 1 illustrates this concave transform: Symphony matches the standard policy gradient locally around $r = 1$ while growing sublinearly for large r, suppressing extreme gradient spikes without discarding data points. To isolate this ratio-damping mechanism from long-horizon credit assignment, we evaluate a nonstationary multi-state bandit with Brownian policy drift and noisy advantages (Appendix C). Figure 2a shows that Symphony maintains substantially lower and tighter logit-gradient dispersion under drift, while Figure 2b confirms that this damping does not impede expected return. Crucially, this mechanism also clarifies why Symphony's empirical advantages emerge primarily in continuous locomotion (where multidimensional Gaussian product ratios compound into extreme outliers) rather than discrete Atari (which lacks action-dimension product compounding, as discrete actions are single categorical variables rather than multi-dimensional products).

Below, we formalize the sublinear power-law surrogate family and analyze its second-moment variance suppression.

2 Proposed Method: Symphony

Preliminaries and Setup. We consider an infinite-horizon Markov Decision Process (MDP) defined by measurable state and action spaces $(\mathcal{S}, \mathcal{B}(\mathcal{S}))$ and $(\mathcal{A}, \mathcal{B}(\mathcal{A}))$, transition kernel $P(\cdot \mid s, a)$, bounded reward $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ with $\|r\|_\infty \leq R_{\max}$, discount factor $\gamma \in [0, 1)$, and initial state distribution μ (measure-theoretic foundations and trajectory distributions are detailed in Appendix B). A Markov policy π induces discounted state-visitation measure $d_\pi(B) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathbb{P}_\mu^\pi(S_t \in B)$, value function $V^\pi(s) = \mathbb{E}_\mu^\pi[\sum_{t=0}^{\infty} \gamma^t r(S_t, A_t) \mid S_0 = s]$, state-action value $Q^\pi(s,a) = \mathbb{E}_\mu^\pi[\sum_{t=0}^{\infty} \gamma^t r(S_t, A_t) \mid S_0 = s, A_0 = a]$, and advantage $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ satisfying $\int_{\mathcal{A}} A^\pi(s,a)\pi(da \mid s) = 0$. With expected return $\mathcal{J}(\pi) = \mathbb{E}_{S_0 \sim \mu}[V^\pi(S_0)]$, the performance difference lemma (Kakade & Langford, 2002) establishes:

$$\mathcal{J}(\pi') - \mathcal{J}(\pi) = \frac{1}{1-\gamma} \int_{\mathcal{S}} d_{\pi'}(ds) \int_{\mathcal{A}} A^\pi(s,a)\pi'(da \mid s). \tag{1}$$

We use this identity as the starting foundation for the Symphony objective (Appendix B.3).

Multi-Action Likelihood Ratios and the Symphony Objective. In continuous control with independent Gaussian policy distributions over d action dimensions, the likelihood ratio factors multiplicatively as a product across dimensions, $r_{\pi'}(s,a) = \prod_{i=1}^{d} \frac{\pi'(a_i|s)}{\pi(a_i|s)}$. When d is moderate or large (e.g., $d = 17$ in Humanoid), even minor per-dimension policy shifts cause r to compound into extreme tail values ($r > 100$). Because standard policy optimization surrogates (such as PPO) have gradients linear in r ($\nabla_\theta L \propto r \nabla_\theta \log \pi_\theta$), these ratio spikes generate heavy-tailed gradient outliers and steep variance growth ($\mathbb{E}[\|\hat{g}\|_2^2] \propto r^2$).

Starting from Equation (1), assuming absolute continuity $\pi'(\cdot \mid s) \ll \pi(\cdot \mid s)$ for all $s \in \mathrm{supp}(d_{\pi'})$, we have $\mathcal{J}(\pi') - \mathcal{J}(\pi) = \frac{1}{1-\gamma}\mathbb{E}_{s \sim d_{\pi'},\, a \sim \pi}[r_{\pi'}(s,a)A^\pi(s,a)]$. To tame ratio-driven gradient explosions, we derive $f(r)$ from three fundamental consistency principles: (i) **Objective Value Preservation:** $f(1) = 1$; (ii) **First-Order Policy Gradient Alignment:** $f'(1) = 1$ (guaranteeing exact gradient equivalence at θ_{old}); and (iii) **Sublinear Power-Law Attenuation:** $f(r) \sim r^{1/n}$ with $n > 1$ to damp heavy-tailed ratio excursions. Solving $f(r) = ar^{1/n} + b$ subject to (i) and (ii) uniquely

yields $a/n = 1 \implies a = n$ and $a + b = 1 \implies b = -(n-1)$, defining the power-law surrogate family:

$$f_n(r) = nr^{1/n} - (n-1). \tag{2}$$

Valid for any continuous order $n > 1$, this parametric form natively introduces negative curvature $f_n''(1) = -(n-1)/n < 0$. For integer $n \geq 2$, $f_n(r)$ corresponds to the first-order truncation of the binomial identity $r = (1 + (r^{1/n} - 1))^n$, decomposing the likelihood ratio into this primary surrogate and an exact higher-order residual: $r_{\pi'}(s,a) = f_n(r_{\pi'}(s,a)) + \sum_{k=2}^{n} \binom{n}{k} \left(r_{\pi'}(s,a)^{1/n} - 1\right)^k$. We adopt $n = 2$ as the default, reducing quadratic second-moment growth to linear scale (Appendix B.5).

For any probability measure ν on $\mathcal{S}$, define $M_n(\nu) = \mathbb{E}_{s\sim\nu,\, a\sim\pi}[f_n(r_{\pi'}(s,a))A^\pi(s,a)]$ and $R_n(\nu) = \mathbb{E}_{s\sim\nu,\, a\sim\pi}\left[\sum_{k=2}^{n} \binom{n}{k} \left(r_{\pi'}(s,a)^{1/n} - 1\right)^k A^\pi(s,a)\right]$. The exact auxiliary decomposition is $\mathcal{J}(\pi') - \mathcal{J}(\pi) = \frac{1}{1-\gamma}\left(M_n(d_{\pi'}) + R_n(d_{\pi'})\right)$. Localizing this expansion to d_π gives $\mathcal{L}_n(\pi') = \mathcal{J}(\pi) + (1-\gamma)^{-1}(M_n(d_\pi) + R_n(d_\pi))$. This auxiliary surrogate tracks the full binomial expansion. The optimized Symphony surrogate keeps only the primary term: $L_n(\pi') = \mathcal{J}(\pi) + \frac{1}{1-\gamma} M_n(d_\pi) = \mathcal{J}(\pi) + \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\pi,\, a\sim\pi}[f_n(r_{\pi'}(s,a))A^\pi(s,a)]$. Thus, $\mathcal{L}_n$ is a derivational object, while L_n is the update objective.

The difference is the omitted residual, $\mathcal{L}_n(\pi') - L_n(\pi') = \frac{1}{1-\gamma} R_n(d_\pi)$. At the population level, if $|r_{\pi'}(s,a)^{1/n} - 1| \leq \rho_n \leq \bar{\rho}$[1] and $|A^\pi(s,a)| \leq A_{\max}$ hold almost surely under $(s,a) \sim d_\pi \times \pi$ (or over the finite batch support for empirical surrogates), then for fixed $n \geq 2$:

$$\begin{aligned} |\mathcal{L}_n(\pi') - L_n(\pi')| &\leq \frac{A_{\max}}{1-\gamma} \sum_{k=2}^{n} \binom{n}{k} \rho_n^k \\ &= \frac{A_{\max}}{1-\gamma}\left((1+\rho_n)^n - 1 - n\rho_n\right) \leq \frac{A_{\max}}{1-\gamma} C_{n,\bar{\rho}}\, \rho_n^2 = O_n(\rho_n^2). \end{aligned} \tag{3}$$

Equation (3) establishes that the omitted residual scales strictly as $O_n(\rho_n^2)$, where the finite constant $C_{n,\bar{\rho}}$ explicitly controls the higher-order terms (with $C_{2,\bar{\rho}} \equiv 1$ identically for $n = 2$). Note that controlling this localized error under d_π does not automatically bound the true residual $R_n(d_{\pi'})$ in Equation (4), as state-visitation shifts can alter the underlying distribution. Surrogate clipping modifies the objective on sampled transitions but does not enforce a hard bound on the resulting policy ratios.

Let $\epsilon_n = \max_s |\mathbb{E}_{a\sim\pi}[f_n(r_{\pi'}(s,a))A^\pi(s,a)]|$; then $|M_n(d_{\pi'}) - M_n(d_\pi)| \leq \epsilon_n \|d_{\pi'} - d_\pi\|_1$. Using the resolvent form $d_\pi^\top = (1-\gamma)\mu^\top (I - \gamma P_\pi)^{-1}$ gives $d_{\pi'}^\top - d_\pi^\top = \gamma d_{\pi'}^\top (P_{\pi'} - P_\pi)(I - \gamma P_\pi)^{-1}$. Therefore, $\|d_{\pi'} - d_\pi\|_1 \leq \frac{2\gamma}{1-\gamma} \max_s D_{\mathrm{TV}}(\pi'(\cdot \mid s), \pi(\cdot \mid s))$. Together with Pinsker's inequality, $D_{\mathrm{TV}}(\pi'(\cdot \mid s), \pi(\cdot \mid s)) \leq \sqrt{\frac{1}{2} D_{\mathrm{KL}}(\pi'(\cdot|s) \,\|\, \pi(\cdot|s))}$, this yields $|M_n(d_{\pi'}) - M_n(d_\pi)| \leq \frac{\gamma\epsilon_n}{1-\gamma} \max_s \sqrt{2 D_{\mathrm{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s))}$. The localization argument uses the resolvent norm bound for discounted state distributions (see Appendix B.4). Combining the localization and residual terms yields the analytical residual and localization bound in Equation (4) (Proposition B.6):

$$\begin{aligned} \left|\mathcal{J}(\pi') - \left(\mathcal{J}(\pi) + \frac{1}{1-\gamma} M_n(d_\pi)\right)\right| &\leq \frac{\gamma\epsilon_n}{(1-\gamma)^2} \max_s \sqrt{2 D_{\mathrm{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s))} \\ &\quad + \frac{1}{1-\gamma} |R_n(d_{\pi'})|. \end{aligned} \tag{4}$$

Here, $R_n(d_{\pi'})$ denotes the higher-order residual evaluated under the future policy's state distribution $d_{\pi'}$. Equation (4) bounds the analytical approximation error between true expected return $\mathcal{J}(\pi')$ and the localized Symphony surrogate $L_n(\pi')$. We emphasize that this bound quantifies surrogate approximation error rather than claiming monotonic policy improvement for Algorithm 1: the residual penalty term depends on $d_{\pi'}$, which is unknown at optimization time and cannot be computed or

[1] Here, $\bar{\rho} < \infty$ denotes a uniform bound on root-ratio displacement defining a compact policy neighborhood (e.g., $\bar{\rho} = 1$ corresponds to $r \in (0, 2^n]$); $C_{n,\bar{\rho}} = \sum_{k=2}^{n} \binom{n}{k} \bar{\rho}^{k-2} < \infty$ is the finite constant obtained by factoring ρ_n^2 from the residual (with $C_{2,\bar{\rho}} \equiv 1$ identically for $n = 2$); and $O_n(\rho_n^2)$ denotes standard Landau Big-O scaling as $\rho_n \to 0$ with implicit constant depending on n and $\bar{\rho}$ (see Appendix B.5).

minimized directly, and practical updates optimize the first-order clipped surrogate without verifying or enforcing the minorizer pointwise. In implementation, we set $\pi = \pi_{\theta_{\text{old}}}$ and $\pi' = \pi_\theta$. The surrogate objective to maximize is $L_n(\theta) = \mathbb{E}_{s\sim d_{\pi_{\theta_{\text{old}}}},\, a\sim\pi_{\theta_{\text{old}}}}[f_n(r_\theta(s,a))A^{\pi_{\theta_{\text{old}}}}(s,a)]$, where $r_\theta(s,a) = \pi_\theta(a \mid s)/\pi_{\theta_{\text{old}}}(a \mid s)$ and $f_n(r) = nr^{1/n} - (n-1)$. The clipped variant, ***Symphony (Clipped)***, maximizes:

$$L_n^{\text{clip}}(\theta) = \mathbb{E}\left[\min\left\{f_n(r_\theta)A,\ f_n(\text{clip}(r_\theta, 1-\varepsilon, 1+\varepsilon))A\right\}\right]. \tag{5}$$

The practical update procedure is summarized in Algorithm 1 (with standard minimizers optimizing $-L_n$).

For the KL-constrained trust-region variant, ***Symphony (KL)***, we consider the constrained problem over $\mathcal{K}_\delta = \{\theta : \mathbb{E}_{s\sim d_{\pi_{\theta_{\text{old}}}}}[D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot \mid s) \parallel \pi_\theta(\cdot \mid s))] \le \delta\}$. While this formalizes the exact Riemannian geometry under a hard trust-region constraint, practical deep RL follows standard practice (Schulman et al., 2017) by optimizing the unconstrained dual Lagrangian relaxation $\mathfrak{L}(\theta, \lambda_{\text{sym}}) = L_n(\theta) - \lambda_{\text{sym}}\mathbb{E}_{s\sim d_{\pi_{\theta_{\text{old}}}}}[D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot \mid s) \parallel \pi_\theta(\cdot \mid s))]$ with calibrated penalty λ_{sym} (formalized in Algorithm 1). Around θ_{old}, using $L_n(\theta_{\text{old}} + \Delta\theta) \approx L_n(\theta_{\text{old}}) + g_n^\top \Delta\theta$ and $D_{\text{KL}}(\pi_{\theta_{\text{old}}} \| \pi_{\theta_{\text{old}}+\Delta\theta}) \approx \frac{1}{2}\Delta\theta^\top F \Delta\theta$, the local problem is $\max_{\Delta\theta} g_n^\top \Delta\theta$ subject to $\frac{1}{2}\Delta\theta^\top F\Delta\theta \le \delta$.

The Fisher matrix is the local Hessian of the KL, $F = \mathbb{E}_{s,a\sim\pi_{\theta_{\text{old}}}}\left[\nabla_\theta \log \pi_\theta(a \mid s) \nabla_\theta \log \pi_\theta(a \mid s)^\top\right]_{\theta=\theta_{\text{old}}}$. Solving the stationarity condition $g_n - \lambda F \Delta\theta = 0$ and enforcing the active constraint gives $\Delta\theta = \sqrt{2\delta/(g_n^\top F^{-1} g_n)} F^{-1} g_n$, where $g_n = \nabla_\theta L_n(\theta)\big|_{\theta=\theta_{\text{old}}}$. Since $f_n'(r) = r^{1/n-1}$ and $r_\theta = 1$ at θ_{old}, we have $f_n'(1) = 1$. Therefore, $g_n = \mathbb{E}_{s,a\sim\pi_{\theta_{\text{old}}}}[\nabla_\theta \log \pi_\theta(a \mid s) A^{\pi_{\theta_{\text{old}}}}(s,a)]_{\theta=\theta_{\text{old}}}$. Because $f_n'(1) = 1$, Symphony preserves the policy-gradient direction at the anchor θ_{old} and, under local KL geometry, the corresponding natural-gradient direction. Its key advantage lies in regularizing finite-displacement trajectories ($\theta \ne \theta_{\text{old}}$) during multi-step Adam optimization: while linear surrogates diverge under multiplicative ratio compounding, Symphony's sublinear curvature ($f_n''(1) = -(n-1)/n < 0$) inherently constrains gradient scale.

Algorithm 1 Symphony (KL) and Symphony (Clipped)

Input: policy θ_0, value function ϕ_0, order $n \ge 2$, clip range ϵ, penalty λ_{sym}, variant $v \in \{\text{KL}, \text{Clipped}\}$
for $k = 0, 1, 2, \ldots$ **do**
  Roll out π_{θ_k} to collect $\mathcal{D}_k$; compute returns $\hat{R}_t$ and advantages $\hat{A}_t$ using GAE (Schulman et al., 2016)
  Set $\ell_t(\theta) = \log \pi_\theta(a_t \mid s_t) - \log \pi_{\theta_k}(a_t \mid s_t)$ and $s_t(\theta) = n\ \text{expm1}(\ell_t(\theta)/n) + 1$
  if $v = \text{KL}$ **then**
    $q_t(\theta) = s_t(\theta)\hat{A}_t - \lambda_{\text{sym}} D_{\text{KL}}(\pi_{\theta_k}(\cdot \mid s_t) \| \pi_\theta(\cdot \mid s_t))$
  else
    $\bar{\ell}_t(\theta) = \text{clip}(\ell_t(\theta), \log(1-\epsilon), \log(1+\epsilon))$
    $\bar{s}_t(\theta) = n\ \text{expm1}(\bar{\ell}_t(\theta)/n) + 1, \quad q_t(\theta) = \min\{s_t(\theta)\hat{A}_t, \bar{s}_t(\theta)\hat{A}_t\}$
  end if
  $\theta_{k+1} = \arg\max_\theta \mathbb{E}_t[q_t(\theta)]$
  $\phi_{k+1} = \arg\min_\phi \mathbb{E}_t[(V_\phi(s_t) - \hat{R}_t)^2]$
end for

Variance Suppression and Choice of Transformation Order n. The core theoretical mechanism of Symphony is the suppression of heavy-tailed gradient variance induced by multi-action likelihood ratio compounding. Differentiating $f_n(r)$ gives $f_n'(r) = r^{1/n-1}$. Since $\nabla_\theta r_\theta = r_\theta \nabla_\theta \log \pi_\theta(a \mid s)$, the stochastic policy gradient is $\hat{g}_n(s,a) = r_\theta^{1/n} \nabla_\theta \log \pi_\theta(a \mid s) A^\pi(s,a)$. Assuming $C := \max_{s,a} \|\nabla_\theta \log \pi_\theta(a \mid s)\|_2^2 (A^\pi(s,a))^2 < \infty$, the trace second-moment matrix satisfies:

$$\text{Tr}\left(\mathbb{E}[\hat{g}_n \hat{g}_n^\top]\right) \le C \cdot \mathbb{E}_{s,a}\left[r_\theta(s,a)^{2/n}\right]. \tag{6}$$

While total gradient variance depends jointly on advantage estimates and score function norms, Symphony specifically bounds the ratio-dependent growth rate of the trace second-moment matrix from quadratic $\mathcal{O}(r^2)$ (at $n = 1$) to sublinear $\mathcal{O}(r^{2/n})$, and strictly linear $\mathcal{O}(r)$ for $n = 2$. This dampens

Table 1: **Effect of the Symphony transformation order n.** Larger n reduces the dependence of the stochastic gradient on large likelihood ratios, changing the dominant second-moment scaling from $O(r^2)$ at $n = 1$ to $O(r^{2/n})$. This improves stability but increasingly compresses the finite-ratio improvement signal. The default choice $n = 2$ is the first nonlinear member of the family and reduces quadratic growth to linear growth.

n	$f_n(r)$	**Gradient scale**	**2nd moment**	$\mathcal{L}_n - L_n$
1	r	$O(r)$	$O(r^2)$	0
2	$2\sqrt{r} - 1$	$O(r^{1/2})$	$O(r)$	$\frac{1}{1-\gamma}\mathbb{E}[(\sqrt{r}-1)^2 A^\pi]$
4	$4r^{1/4} - 3$	$O(r^{1/4})$	$O(r^{1/2})$	$\frac{1}{1-\gamma}R_4(d_\pi)$
16	$16r^{1/16} - 15$	$O(r^{1/16})$	$O(r^{1/8})$	$\frac{1}{1-\gamma}R_{16}(d_\pi)$
∞	$1 + \log r$	$O(1)$	$O(1)$	limiting residual

the primary driver of heavy-tailed updates in multi-dimensional continuous action spaces (Garg et al., 2021) without discarding samples. Table 1 summarizes this order-dependent scaling.

The bias–stability tradeoff in Table 1 guides our choice: as $n \to \infty$, the surrogate approaches a logarithmic regime $(1 + \log r)$ that heavily compresses improvement signals; $n = 2$ is the first nonlinear order, converting quadratic variance growth into linear scale while preserving ratio signals. As shown in Equation (3), the omitted residual $\frac{1}{1-\gamma}R_n(d_\pi)$ is second order $\mathcal{O}(\rho_n^2)$, remaining tightly controlled under proximal updates.

Implicit Trust Region and Curvature Substitution. Let $r = 1+\delta$. Expanding locally around $\delta = 0$ yields $f_n(1+\delta)A^\pi \approx A^\pi(1+\delta) - \left(\frac{n-1}{2n}A^\pi\right)\delta^2 + \mathcal{O}(\delta^3)$. Because the intrinsic quadratic term $-\left(\frac{n-1}{2n}A^\pi\right)\delta^2$ is advantage-weighted, its sign depends on A^π: for positive advantages ($A^\pi > 0$), where expanding probability mass risks ratio spikes, negative quadratic curvature acts as an adaptive local brake. When paired with an external KL penalty $-\frac{\lambda_{\text{sym}}}{2}\delta^2$, total effective second-order braking along positive directions ($A^\pi \approx 1.0$) is $\kappa_{\text{eff}} = \frac{n-1}{2n}A^\pi + \frac{\lambda_{\text{sym}}}{2}$. For linear surrogates ($n = 1$), intrinsic curvature vanishes ($\frac{1-1}{2} = 0$), requiring full external penalization ($\frac{\lambda}{2}$). Under standardized advantages ($A^\pi \approx 1.0$) and $n = 2$, Symphony natively supplies $\frac{2-1}{4}(1.0) = 0.25$. Matching a linear baseline's effective braking of 0.50 ($\lambda = 1.0$) requires $\frac{\lambda_{\text{sym}}}{2} = 0.50 - 0.25 = 0.25 \implies \lambda_{\text{sym}} = 0.5$. Rather than an exact global equivalence to a trust-region penalty, $\lambda_{\text{sym}} = 0.5$ serves as a motivated local calibration under positive advantages, corroborated by empirical penalty sweeps $\beta \in \{0.1, 0.5, 1.0, 5.0\}$ in Appendix D.1.3 (Figure 8). Consequently, the two Symphony variants establish a clear division of labor: *Symphony (KL)* serves as the *theoretical anchor* verifying natural-gradient alignment and local curvature substitution, while *Symphony (Clipped)* serves as the *scalable workhorse* delivering maximum first-order throughput and headline benchmarks (10,137 IQM on Humanoid).

3 Experimentation and Result Analysis

With the theoretical foundation of Symphony established in Section 2, our empirical evaluation interrogates its core mechanism: *does bending the surrogate objective away from linearity via negative curvature provide genuine variance reduction and update stability, or does it over-compress usable improvement signals?* To evaluate this, we examine two regimes with contrasting stochastic structure: continuous locomotion in MuJoCo MJX (Freeman et al., 2021) (up to two billion transitions) and visual control in JAXAtari (Delfosse et al., 2026) (up to 25 million transitions). Following Agarwal et al. (2021), we report Interquartile Mean (IQM) and Probability of Improvement (POI, 64 pairwise seed comparisons) computed over the final 10% of training across 8 random seeds with 95% bootstrap confidence intervals. Across benchmarks, algorithms share an identical network architecture, AdamW optimizer, online running observation normalization (Welford RMS, unclipped), raw environment rewards (unnormalized and unclipped), constant learning rates throughout training (unannealed), and unclipped MSE value loss (see Appendix F); for SPO, we evaluate the SPO surrogate objective within this common implementation rather than reproducing its distinct pipeline engineering (see Appendix E.6 and Table 12 for an itemized comparison).

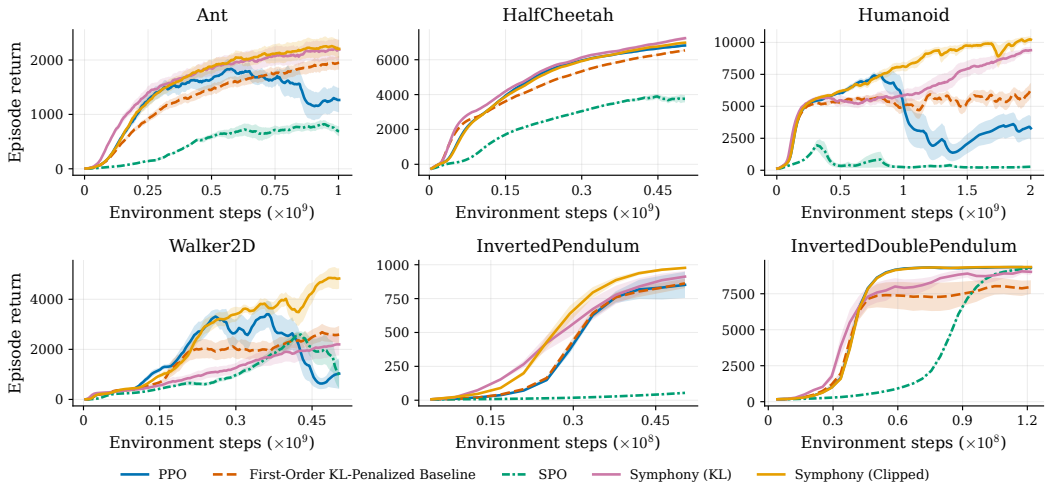


Figure 3: **Learning Curves across MuJoCo MJX Continuous Control Benchmarks.** Symphony (Clipped) demonstrates higher sample efficiency and asymptotic stability on high-dimensional locomotion tasks (*Humanoid*, *Ant*, and *Walker 2D*) while matching baselines on low-dimensional balance tasks. Curves depict mean returns aggregated over 8 independent seeds with 95% bootstrap confidence intervals. Note that on Humanoid and Walker 2D, standard PPO climbs to competitive early returns (≈ 7500 and ≈ 3500, respectively) before experiencing late-stage performance degradation under unannealed constant learning rates; Symphony natively suppresses ratio excursions, maintaining asymptotic stability.

Our evaluation separates three complementary questions. **(i) Long-horizon shared-configuration stress tests** evaluate all methods under the same training setup across MJX tasks, using up to 2B transitions, to characterize late-training behavior and stability. The shared configuration was selected from performance during the initial portion of training, before the later degradation reported below; these experiments therefore characterize long-horizon behavior under a common configuration rather than performance after horizon-specific tuning. **(ii) Baseline-specific Humanoid evaluation** separately screens PPO and SPO over 36 configurations and evaluates the selected settings on fresh seeds for approximately 10M transitions, with Symphony inheriting each selected baseline configuration without additional tuning. **(iii) Implementation-sensitivity evaluation** compares PPO, SPO, and Symphony across our training implementation and an SPO-style MJX pipeline to measure how strongly relative performance depends on surrounding implementation choices.

1. MJX Continuous Control (Long-Horizon Stress Tests). In modern massively parallelized RL (such as Brax/MJX running on GPU accelerators), practitioners aim to train for billions of transitions under constant learning rates. Under this configuration, long-horizon training exposes algorithms to compounding ratio drift: on Humanoid (17 action dimensions) and Walker 2D (6 dimensions), standard PPO initially climbs to competitive returns ($\approx$ 7500 and $\approx$ 3500, respectively) before experiencing late-stage performance degradation down to an IQM of 3749.73 and 439.78. Symphony maintains stable performance across the two-billion-step horizon (10137.06 IQM on Humanoid, 100% POI; 5007.01 on Walker 2D). Because Symphony's $n = 2$ transform bounds dominant second-moment gradient growth to linear scale ($\mathcal{O}(r)$ vs. $\mathcal{O}(r^2)$), long-horizon policy optimization exhibits greater resilience to ratio variance without schedule annealing. On low-dimensional balance tasks (Inverted Pendulum and Inverted Double Pendulum), Symphony matches baselines (990.96 and 9356.19), verifying that power-law damping does not impede benign dynamics.

2. JAXAtari Discrete Visual Control and Boundary Conditions. Visual control across seven Atari games (Part IV of Table 2) evaluates raw pixel frames ($84 \times 84 \times 4$) under a shared convolutional actor–critic architecture (Appendix F) across 8 seeds. Symphony (Clipped) achieves comparable scores to PPO across games (Breakout: 56.34 vs. 55.61; Pong: 20.39 vs. 20.36), with completely overlapping 95% bootstrap confidence intervals. Crucially, this parity reflects a structural boundary condition: discrete Atari environments feature scalar categorical actions without action-dimension product compounding. While single-action ratios satisfy $r \leq 1/\pi_{\text{old}}(a \mid s)$ (which can still be large

Table 2: **Benchmark Results, Environmental Dimensions, and Ablations.** All evaluations report Interquartile Mean (IQM) and Probability of Improvement (POI, 64 pairwise seed comparisons) aggregated over the final 10% of training across 8 seeds with 95% bootstrap confidence intervals. Locomotion shows substantial gains on Humanoid and Walker 2D; JAXAtari exhibits competitive, environment-dependent performance (full distributions and CIs in Appendix D). PPO returns on Humanoid and Walker 2D reflect asymptotic late-training degradation over 2B/500M steps under constant learning rates (as detailed in Figure 3; peak returns prior to degradation were substantially higher).

Part I: MJX Continuous Control Performance & Environmental Complexity

Environment	Humanoid	Ant	Walker 2D	Half Cheetah	Inv. Doub. Pend	Inv. Pend
Obs. Dimension	227	87	17	18	9	4
Act. Dimension (Cont.)	17	8	6	6	1	1
Training Duration	2.00B	1.00B	503.31M	503.31M	121.63M	50.33M
PPO	3749.73	1402.43	439.78	6823.49	9327.01	986.34
First-Order KL Baseline	5870.06	1969.31	2642.99	6445.09	7968.49	876.30
SPO	194.83	788.37	1602.39	3757.87	9282.41	58.46
Symphony (KL)	9464.72	2206.67	1647.37	**7085.84**	9234.15	971.90
Symphony (Clipped)	**10137.06**	**2277.88**	**5007.01**	6905.14	**9356.19**	**990.96**
POI (> PPO)	*100.0%*	*90.6%*	*98.4%*	*67.2%*	*71.9%*	*65.6%*
POI (> SPO)	*100.0%*	*100.0%*	*98.4%*	*100.0%*	*100.0%*	*100.0%*

Part II: Ablation of Symphony Order (8 Seeds, IQM Scores)

Environment	Order 1 (PPO)	Order 2 (Symphony)	Order 4	Order 16	Order 32	Order 64
Humanoid	3749.73	10137.06	10399.70	**10560.10**	10389.30	10217.00
Ant	1402.43	2277.88	**2440.52**	2185.17	2283.22	2174.04

Part III: Ablation of Proximal Clipping Threshold (8 Seeds)

	Clipping Threshold 0.1		Clipping Threshold 0.2		Clipping Threshold 0.4	
Algorithm	Humanoid	Ant	Humanoid	Ant	Humanoid	Ant
PPO	1047.16	1899.19	3749.73	1402.43	568.82	168.82
Symphony (Clipped)	**9262.67**	**2005.43**	**10137.06**	**2277.88**	**9344.21**	**1309.47**
POI (> PPO)	*100.0%*	*56.3%*	*100.0%*	*90.6%*	*100.0%*	*100.0%*

Part IV: JAXAtari Benchmark Performance & Environmental Complexity (IQM Scores)

Environment	Breakout	Freeway	Seaquest	Slot Machine	King Kong	Q*bert	Pong
Obs. Space	$84 \times 84 \times 4$ visual pixel frames						
Act. Dim. (Discrete)	4	3	18	9	6	6	6
Training Duration	10.03M	25.00M	25.00M	25.00M	10.03M	10.03M	10.03M
PPO	55.61	24.95	101.75	**27.91**	26.01	**58.38**	20.36
First-Order KL Baseline	33.61	22.81	91.92	17.81	11.82	50.74	19.84
Symphony (KL)	38.50	22.88	100.57	20.25	11.68	49.15	19.94
Symphony (Clipped)	**56.34**	**26.80**	**107.72**	23.28	**27.25**	58.05	**20.39**
POI (> PPO)	*57.8%*	*57.0%*	*59.4%*	*31.3%*	*57.8%*	*48.4%*	*54.7%*

if an action had low initial probability), discrete tasks avoid the exponential compounding $\prod_{i=1}^{d} r_i$ that drives heavy-tailed gradient variance in high-dimensional continuous control. Consequently, power-law damping matches PPO in discrete control; its primary variance-suppression gains emerge specifically where multi-dimensional action compounding is severe.

3.1 An Analysis of Curvature and Boundaries

Transformation Order. Order n governs the bias–stability tradeoff: power-law damping suppresses tail ratios in continuous control, but higher orders compress usable improvement signals. We adopt $n = 2$ as a conservative default. Table 2 (Part II) shows task-dependent curvature: Ant peaks at order 4 (2440.52), while Humanoid benefits from stronger damping up to order 16 (10560.10), compressing variance toward $\mathcal{O}(1)$.

Clipping Threshold. Relaxing clipping to $\epsilon = 0.4$ collapses PPO to 15.2% return retention on Humanoid (569 vs. 3750) and 12.0% on Ant (169 vs. 1402; Appendix D.1.4). In contrast, Symphony

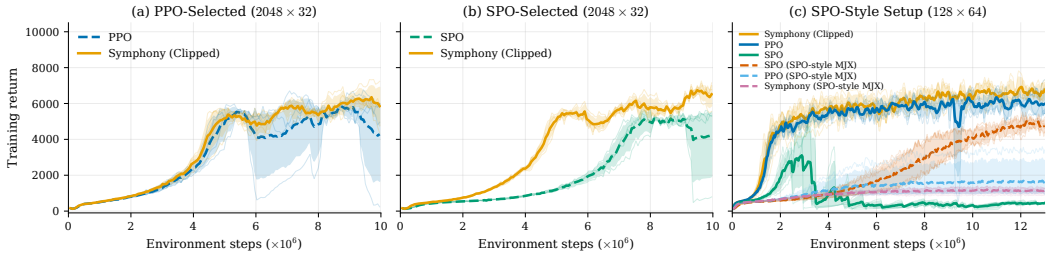


Figure 4: **Humanoid Controlled Confirmation Returns.** Mean returns with 95% bootstrap CI (faint lines: individual seeds): (a) PPO-selected (2048×32), (b) SPO-selected (2048×32), and (c) SPO-style against our implementation (128×64, 13M steps, four seeds, six configurations). Full audits in Appendix E.

retains 92.2% on Humanoid (9344) and 57.5% on Ant (1309), smoothly attenuating outliers via negative curvature before external clipping intervenes.

3.2 Controlled Baseline and Implementation Sensitivity on Humanoid

Baseline-specific selection. The extended-horizon experiments compare algorithms under a shared training configuration. To test whether the observed gains persist after baseline-specific tuning, we separately screen PPO and SPO over 36 configurations varying learning rate, rollout length, the surrogate clipping/penalty parameter, and update epochs. Approximately 5M transitions per configuration provide an affordable screening budget; configurations are ranked by mean training-return AUC across two screening seeds. The selected configurations are then evaluated over approximately 10M transitions on fresh seeds, with Symphony inheriting each baseline's settings without additional tuning. PPO selects $(\alpha, L, \epsilon, E) = (3 \times 10^{-4}, 32, 0.4, 8)$, while SPO selects $(10^{-4}, 32, 0.4, 8)$. This study evaluates performance after baseline-specific selection within our implementation; it complements the long-horizon experiments rather than replacing them. Under PPO-selected settings[2], Symphony achieves 5852 ± 1182 vs. 4290 ± 2746 for PPO ($+1562$ lead; 95% CI: $[+291, +3319]$). Under SPO-selected settings, Symphony achieves 6513 ± 678 vs. 4233 ± 2296 for SPO ($+2280$ lead; 95% CI: $[+574, +5028]$).

Objective–implementation interaction. We evaluate PPO, SPO, and Symphony in two training implementations under matched interaction budgets and common selected hyperparameters (Figure 4c). The observed ranking reverses across implementations: Symphony and PPO achieve high returns in our implementation (6763 ± 549 and 5973 ± 910), whereas SPO performs best in the SPO-style MJX implementation (4768 ± 354). SPO deteriorates after initial learning in our implementation (449 ± 49), while PPO and Symphony plateau at low returns in the SPO-style implementation (1641 ± 1182 and 1139 ± 174). These results demonstrate that relative performance depends on the surrounding training implementation; they do not isolate the individual components responsible (Appendix E.7).

4 Conclusion and Limitations

We introduced Symphony, a concave power-law policy objective $f_n(r) = nr^{1/n} - (n-1)$ that preserves the on-policy gradient while reducing ratio-dependent second-moment growth from $\mathcal{O}(r^2)$ to $\mathcal{O}(r^{2/n})$. Empirically, Symphony improves long-horizon stability in multidimensional continuous control, where likelihood ratios compound across action dimensions, while matching PPO on discrete Atari tasks. **Limitations and Future Work.** Performance remains sensitive to the surrounding training implementation. Future work will study adaptive order selection and these objective–implementation interactions.

[2] α: learning rate, L: rollout steps, ϵ: clip threshold, E: epochs. 128×64: 128 envs $\times$ 64 steps (8,192 steps/batch). The SPO-style setup adapts SPO's 8×256 to 128×64 for GPU execution, sharing $\alpha = 3 \times 10^{-4}$ and batch size across seeds 1–4.

AI USE STATEMENT

In this work, we used browser-accessible generative AI models for manuscript polishing and limited technical assistance. Specifically, GPT-5.4 Thinking[3] and Gemini 3.1 Flash-Lite[4] were used to review author-written mathematical proofs and to assist with writing code. More recently, Gemini 3.8 Flash[5], GPT-5.6 Sol[6], and GPT-6 Astra[7] were used to improve sentence structure and to conduct additional reviews of author-written mathematical proofs. Generative AI tools were not used for research ideation. All AI-assisted output was reviewed by the authors, and all references were manually audited and verified. The authors take full responsibility for the final text, claims, mathematical arguments, code, and conclusions presented in this work.

REPRODUCIBILITY STATEMENT

All algorithmic implementations, training pipelines, hyperparameter configurations, and raw evaluation CSV logs required to reproduce every empirical result, figure, and statistical table in this paper are provided in the accompanying anonymized repository:

- **Continuous Control Logs (`codes/mjx_runs/`):** Complete per-iteration CSV training and evaluation metrics across all six MuJoCo MJX continuous control benchmarks (up to two billion transitions per run across eight random seeds).
- **Discrete Visual Control Logs (`codes/atari_runs/`):** Raw evaluation CSV logs across seven JAXAtari pixel-based games across eight random seeds.
- **Ablation and Parameter Sweep Logs:** Complete CSV metric trajectories for surrogate order ablations (`codes/order_ranges_runs/`), clipping threshold sweeps (`codes/clip_ranges_runs/`), penalty coefficient sweeps (`codes/beta_ranges/`), and learning rate sweeps (`codes/lr_runs/`).
- **Controlled Baseline Evaluation Logs (`codes/control_evaluation/`):** Raw evaluation CSV logs, configuration specifications, and summary tables for the 10M-step Humanoid baseline comparison evaluating Symphony against independently tuned configurations of PPO and SPO (under subdirectories `symphony_vs_ppo/` and `symphony_vs_spo/`), accompanied by protocol documentation in `README.md`.
- **SPO-Style Against Our Implementation Logs (`codes/spo_style_against_our_implementation/`):** Complete per-iteration CSV evaluation logs across seeds 1–4 evaluating PPO, SPO, and Symphony (Clipped) across both our primary pipeline (`ours/`) and the SPO-style MJX pipeline (`spo-style/`).
- **Literal Graph Reproduction Commands in Supplementary READMEs:** The exact standalone Python code and literal command-line instructions required to reproduce all graphs, curves, and diagnostic plots directly from the underlying CSV logs are provided in the `README.md` files of the supplementary material (`codes/README.md`, `codes/control_evaluation/README.md`, and `codes/spo_style_against_our_implementation/README.md`).
- **Statistical Uncertainty Tools:** Automated reproduction scripts to regenerate all manuscript figures (`codes/plot.py`, `codes/plot_control_evaluation.py`) and compute Interquartile Means (IQM), 95% percentile bootstrap confidence intervals, and Probability of Improvement (`codes/poi.py`).

All source code, environment specifications, random seed allocations, and evaluation logs will be released publicly upon publication.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. Pmlr, 2017.

[3] OpenAI, *Introducing GPT-5.4*.
[4] Google, *Gemini 3.1 Flash-Lite*.
[5] Google, *Introducing Gemini 3.8 Flash and 3.8 Flash Cyber*.
[6] OpenAI, *GPT-5.6: Frontier intelligence that scales with your ambition*.
[7] OpenAI, *GPT-6 Astra: A new generation of intelligence*.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016.

Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Leonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, et al. What matters for on-policy deep actor-critic methods? a large-scale study. In *International conference on learning representations*, 2021.

Boris Belousov and Jan Peters. f-divergence constrained policy improvement. *arXiv preprint arXiv:1801.00056*, 2017.

Alan Chan, Hugo Silva, Sungsu Lim, Tadashi Kozuno, A Rupam Mahmood, and Martha White. Greedification operators for policy optimization: Investigating forward and reverse kl divergences. *Journal of Machine Learning Research*, 23(253):1–79, 2022.

Quentin Delfosse, Raban Emunds, Paul Seitz, Sebastian Wette, Jannis Blüml, and Kristian Kersting. Jaxatari: A high-performance framework for reasoning agents. https://github.com/k4ntz/JAXAtari/, 2026.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pp. 1329–1338. PMLR, 2016.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep rl: A case study on ppo and trpo. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1etN1rtPB.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pp. 1407–1416. PMLR, 2018.

C Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax–a differentiable physics engine for large scale rigid body simulation. *arXiv preprint arXiv:2106.13281*, 2021.

Saurabh Garg, Joshua Zhanson, Emilio Parisotto, Adarsh Prasad, Zico Kolter, Zachary Lipton, Sivaraman Balakrishnan, Ruslan Salakhutdinov, and Pradeep Ravikumar. On proximal policy optimization's heavy-tailed gradients. In *International Conference on Machine Learning*, pp. 3610–3619. PMLR, 2021.

Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov):1471–1530, 2004.

Nicolas Heess, Dhruva Tb, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, SM Eslami, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the nineteenth international conference on machine learning*, pp. 267–274, 2002.

Sham M Kakade. A natural policy gradient. In 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/4b86abe48d358ecf194c56c69108433e-Paper.pdf.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

Kenneth Lange. *MM Optimization Algorithms*, volume 147 of *Other Titles in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2016. ISBN 978-1-611974-39-3. doi: 10.1137/1.9781611974409.

Yingzhen Li and Richard E Turner. Rényi divergence variational inference. *Advances in neural information processing systems*, 29, 2016.

Zichuan Lin, Xiapeng Wu, Mingfei Sun, Deheng Ye, Qiang Fu, Wei Yang, and Wei Liu. Sample dropout: A simple yet effective variance reduction technique in deep policy optimization. *arXiv preprint arXiv:2302.02299*, 2023.

Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. *Advances in neural information processing systems*, 31, 2018.

Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. *Advances in Neural Information Processing Systems*, 35:16455–16468, 2022.

Yu Luo, Shuo Han, Yihan Hu, Dong Li, and Jianye Hao. Ratio-variance regularized policy optimization for efficient llm fine-tuning. *arXiv preprint arXiv:2601.03320*, 2026.

Alberto Maria Metelli, Matteo Papini, Francesco Faccio, and Marcello Restelli. Policy optimization via importance sampling. *Advances in neural information processing systems*, 31, 2018.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Trust-pcl: An off-policy trust region method for continuous control. In *International Conference on Learning Representations*, 2018.

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.

Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. Stochastic variance-reduced policy gradient. In *International conference on machine learning*, pp. 4026–4035. PMLR, 2018.

Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

David Pfau, Ian Davies, Diana L Borsa, João Guilherme Madeira Araújo, Brendan Daniel Tracey, and Hado Van Hasselt. Wasserstein policy optimization. In *International Conference on Machine Learning*, pp. 49128–49149. PMLR, 2025.

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

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

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Mingfei Sun, Benjamin Ellis, Anuj Mahajan, Sam Devlin, Katja Hofmann, and Shimon Whiteson. Trust-region-free policy optimization for stochastic policies. *arXiv preprint arXiv:2302.07985*, 2023.

Ziwu Sun, Zhen Gao, Jiyong Zhang, and Jiaheng Li. Beyond importance sampling: Rejection-gated policy optimization. *arXiv preprint arXiv:2604.14895*, 2026.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Yunhao Tang, Michal Valko, and Rémi Munos. Taylor expansion policy optimization. In *International Conference on Machine Learning*, pp. 9397–9406. PMLR, 2020.

Ahmed Touati, Amy Zhang, Joelle Pineau, and Pascal Vincent. Stable policy optimization via off-policy divergence regularization. In Jonas Peters and David Sontag (eds.), *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 124 of *Proceedings of Machine Learning Research*, pp. 1328–1337. PMLR, 03–06 Aug 2020. URL https://proceedings.mlr.press/v124/touati20a.html.

Sharan Vaswani, Olivier Bachem, Simone Totaro, Robert Müller, Shivam Garg, Matthieu Geist, Marlos C Machado, Pablo Samuel Castro, and Nicolas Le Roux. A general class of surrogate functions for stable and efficient reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 8619–8649. PMLR, 2022.

Yuhui Wang, Hao He, and Xiaoyang Tan. Truly proximal policy optimization. In *Uncertainty in artificial intelligence*, pp. 113–122. PMLR, 2020.

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

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

Zhengpeng Xie, Qiang Zhang, Fan Yang, Marco Hutter, and Renjing Xu. Simple policy optimization. In *International Conference on Machine Learning*, pp. 68813–68824. PMLR, 2025.

Ruiyi Zhang, Changyou Chen, Chunyuan Li, and Lawrence Carin. Policy optimization as Wasserstein gradient flows. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5737–5746. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/zhang18a.html.

TABLE OF CONTENTS

Visual Overview: MJX Locomotion Dynamics (PPO vs. Symphony) 15

A. Related Works 16

B. Mathematical Notation and Proofs 19

1. Mathematical Notations 19

2. MDP Trajectory Distribution and Controlled Markov Property 20

3. Performance Difference Lemma 20

4. Resolvent Norm and State-Distribution Localization 21

5. Algebraic Properties of the Symphony Transform 22

6. Analytical Error Bounds on the Surrogate and Omitted Residual 24

C. Nonstationary Brownian Bandit Diagnostic 28

D. Extended Empirical Evaluation 30

1. Statistical Uncertainty Reporting and Bootstrap Confidence Intervals 41

E. Controlled Baseline Sensitivity Study on Humanoid 44

1. Purpose and Relationship to Long-Horizon Evaluations 44

2. Transition Budgets and Screening Criterion 45

3. Confirmation Results and Statistical Comparison 46

4. Detailed Training Diagnostics 47

5. Material Differences: Published SPO Pipeline vs. Common Harness 49

6. Direct Evaluation under the Released SPO Pipeline Configuration 51

F. Implementation Details 53

1. JAXAtari Implementation Details 53

2. MuJoCo MJX Continuous Control Implementation Details 55

3. JAX Implementation of the Symphony Policy Loss 58

G. Frequently Asked Questions 59

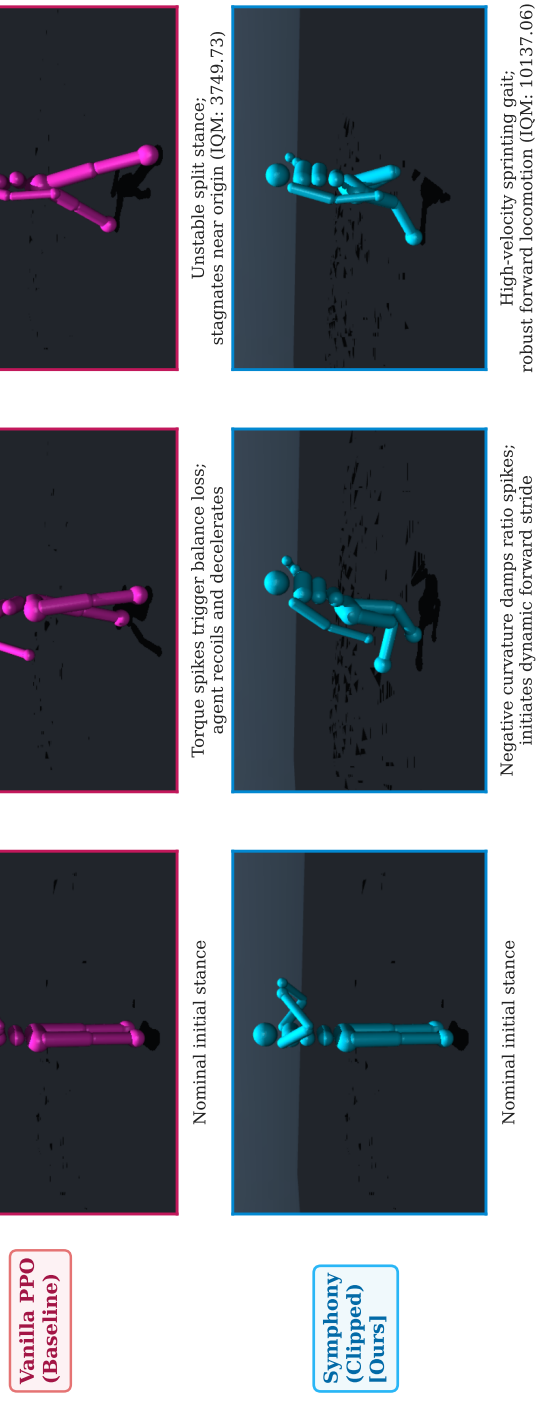


Figure 5: **Qualitative Locomotion Dynamics in MJX Humanoid: Vanilla PPO vs. Symphony (Clipped).** Progression across rollout execution ($t = 0.0\,\text{s}$, $t = 1.0\,\text{s}$, and $t = 2.0\,\text{s}$). While vanilla PPO stumbles under discontinuous foot-ground contact shocks, recoiling and failing to sustain forward velocity (IQM: 3749.73), Symphony (Clipped) natively suppresses ratio-driven variance spikes through its negatively curved surrogate geometry, maintaining dynamic balance and achieving sustained high-velocity forward locomotion (IQM: 10137.06).

A Related Works

A.1 Trust-Region and Natural-Gradient Methods

Trust-region methods provide the main theoretical backdrop for Symphony. Natural-gradient learning views policy updates through the geometry of probability distributions, replacing ordinary steepest descent with Fisher-metric preconditioning (Amari, 1998; 2016). In reinforcement learning, this perspective led to natural policy gradients and natural actor–critic algorithms (Kakade, 2001; Peters & Schaal, 2008). Conservative policy iteration and the performance-difference lemma then connected local surrogate maximization to policy improvement under distributional constraints (Kakade & Langford, 2002). TRPO operationalized this idea for deep policies by optimizing a linear importance-ratio surrogate subject to a mean-KL trust region (Schulman et al., 2015), a construction closely related to minorization–maximization principles (Lange, 2016). Subsequent work has varied the geometry or constraint rather than the pointwise ratio map itself: constrained policy optimization (Achiam et al., 2017), f-divergence and visitation-distribution regularization (Belousov & Peters, 2017; Touati et al., 2020), forward/reverse-KL greedification (Chan et al., 2022), Wasserstein policy flows (Zhang et al., 2018; Pfau et al., 2025), and conservative ratio bounds without an explicit trust-region solve (Sun et al., 2023). These approaches are complementary to Symphony, but their geometric control is imposed through a metric, constraint, or penalty outside the optimized surrogate. Symphony instead places the curvature inside the surrogate by transforming the likelihood ratio as $f_n(r) = nr^{1/n} - (n-1)$, preserving the local policy-gradient direction while tempering finite-ratio deviations.

A.2 First-Order Clipped and Proximal Surrogates

A second line of work seeks trust-region-like behavior using only first-order optimization. PPO's clipped objective became a standard first-order baseline because it is simple, scalable, and effective across continuous control and RLHF-style post-training (Schulman et al., 2017; Duan et al., 2016; Heess et al., 2017; Ouyang et al., 2022; Shao et al., 2024). Related distributed actor–learner systems such as IMPALA instead stabilize stale-policy updates through importance-weighted off-policy correction (Espeholt et al., 2018). At the same time, careful ablations show that PPO's empirical reliability depends heavily on implementation choices such as normalization, initialization, optimizer schedules, and value-function treatment (Engstrom et al., 2020; Andrychowicz et al., 2021). This has motivated objectives that repair or reinterpret the proximal boundary: Truly PPO adds rollback terms and KL-triggered clipping (Wang et al., 2020); SPO incorporates a quadratic ratio penalty derived from total-variation bounds (Xie et al., 2025); and Trust-PCL introduces relative-entropy regularization for off-policy path consistency (Nachum et al., 2018). While these methods introduce penalties or external clipping boundaries to constrain updates, they do not transform the underlying likelihood ratio via concave power-law scaling inside the surrogate itself. Symphony targets this active transformation: the stochastic policy-gradient weight becomes $r^{1/n}\nabla \log \pi_\theta \hat{A}$, damping large likelihood ratios so the ratio-dependent factor in the squared gradient scales as $O(r^{2/n})$ rather than $O(r^2)$, with optional clipping still available as an additional safeguard.

A.3 Variance Reduction and Importance-Ratio Pathologies

Variance reduction in policy optimization has traditionally acted on the advantage estimator, the sampling distribution, or the estimator after gradients have become heavy-tailed. Classical baselines and control variates reduce variance in the return or advantage factor (Williams, 1992; Sutton et al., 1999; Greensmith et al., 2004), while GAE smooths temporal credit assignment through a bias–variance tradeoff (Schulman et al., 2016). On the ratio side, off-policy evaluation highlights how likelihood-ratio products can become unstable over long horizons (Liu et al., 2018). For policy optimization specifically, sample dropout removes high-ratio samples after showing quadratic variance growth in the importance ratio (Lin et al., 2023), and robust-estimator analyses of PPO identify heavy-tailed actor gradients arising from the PPO surrogate, with likelihood ratios and advantage estimates playing central roles as the updated policy drifts from the behavior policy (Garg et al., 2021). Distributed and off-policy algorithms instead truncate or regularize ratios, as in Retrace(λ), V-trace/IMPALA, ACER, and POIS with a Rényi-divergence penalty (Munos et al., 2016; Espeholt et al., 2018; Wang et al., 2017; Metelli et al., 2018; Li & Turner, 2016). SVRPG uses

snapshot/control-variate gradients to reduce stochasticity across policy-gradient iterations (Papini et al., 2018). Symphony is upstream of these mechanisms: it changes the surrogate's ratio dependence before dropout, truncation, robust aggregation, or explicit divergence estimation is needed. Thus existing variance-reduction tools remain compatible, but the baseline moment growth they operate on is reduced from quadratic to $2/n$-power.

A.4 General Surrogate Families and Concurrent Ratio-Control Methods

The closest structural comparison is FMA-PG, which derives policy-gradient surrogate families from functional mirror ascent and mirror maps on policy representations (Vaswani et al., 2022). Symphony is less general but more explicit: the root-ratio family follows from the binomial decomposition of the performance-difference objective and exposes a single order parameter n with direct curvature and variance-scaling meaning. R^{2}VPO regularizes GRPO/PPO-style objectives with a quadratic ratio-variance penalty (Luo et al., 2026), while RGPO replaces importance weighting with a differentiable bounded rejection gate (Sun et al., 2026). Symphony differs by avoiding both an external Lagrange multiplier and a saturating acceptance probability. Its transform satisfies $f_n'(1) = 1$, so the on-policy gradient is exactly preserved; $f_n''(1) = -(n-1)/n$, so local negative curvature is built into the surrogate; and every sample retains a smooth nonzero ratio-dependent gradient contribution rather than being passed through a bounded rejection gate. This places Symphony between broad mirror-map frameworks, explicit variance-regularized PPO/GRPO objectives, and bounded gating approaches: it is a first-order surrogate whose contribution is to reshape the likelihood-ratio geometry itself.

A.5 Higher-Order Taylor Approximations and Meta-Discovered Surrogates: TayPO and DPO

Beyond first-order clipped surrogates and analytical trust regions, two prominent alternative paradigms seek to refine policy optimization objectives: higher-order functional expansions and meta-discovered loss landscapes. We discuss both conceptually to delineate surrogate design paradigms, without implying that we evaluated them empirically in our benchmark suite:

Higher-Order Taylor Expansions (TayPO). Tang et al. (2020) introduced Taylor expansion policy optimization (TayPO), which models non-local policy improvement by performing an arbitrary-order Taylor expansion of the expected return in terms of policy distribution changes. This formulation accounts for state-visitation distribution shifts by expanding trajectory-level returns into multi-step products and interactions of policy-ratio deviations $(r(s_t, a_t) - 1)$ across different time steps along a rollout. Crucially, this expansion operates over policy distributions rather than neural-network parameters: its practical realization optimizes these trajectory-expanded terms via first-order gradient descent without requiring parameter-space Hessians. The essential distinction with Symphony lies in the locus and form of the transformation. TayPO retains multi-step polynomial interactions of ratio deviations along extended horizons to capture cumulative distributional drift. In contrast, Symphony operates as a *pointwise*, single-step surrogate transformation: it applies the concave power-law map $f_n(r) = nr^{1/n} - (n-1)$ directly to the instantaneous importance ratio $r = \pi_\theta(a \mid s)/\pi_{\theta_{\text{old}}}(a \mid s)$ at each transition. By reshaping the immediate likelihood-ratio geometry, Symphony preserves the on-policy gradient direction at $r = 1$ ($f_n'(1) = 1$) while natively attenuating large ratio excursions through negative curvature ($f_n''(r) < 0$), avoiding trajectory-wide cross-product terms.

Meta-Learned and Discovered Objectives (DPO). A parallel line of work uses objective discovery grounded in theoretical frameworks. Discovered Policy Optimisation (DPO; Lu et al., 2022) operates within the Mirror Learning framework, using evolutionary search across sets of training environments to learn effective policy-update behaviors (Learned Policy Optimisation) and subsequently distilling the discovered dynamics into an analytical, closed-form surrogate objective. While DPO demonstrates that effective surrogate forms can be derived by analyzing meta-learned optimization dynamics, its functional form originates from a data-driven search across training environments. In contrast, Symphony is derived analytically from first principles: starting from the performance difference lemma and a finite binomial expansion, it yields a simple family $f_n(r) = nr^{1/n} - (n-1)$ parameterized by an integer order $n \geq 2$. Symphony provides an explicit remainder characterization, matches the on-policy gradient direction at $r = 1$ ($f_n'(1) = 1$) and, under local KL geometry, the

natural-gradient direction, while provably damping ratio-driven squared-gradient growth from $\mathcal{O}(r^2)$ to $\mathcal{O}(r^{2/n})$ without meta-training pipelines.

B Mathematical Notation and Proofs

B.1 Mathematical Notations

This appendix collects the notation used throughout the derivation and implementation of Symphony. Table 3 includes the probability-space notation for the MDP, the value and occupation-measure quantities used in the performance-difference argument, and the Symphony-specific quantities introduced in the power-law surrogate, clipping, and trust-region formulations.

Table 3: **Notation used in the Symphony derivation.** The table includes the main probability-space, MDP, value-function, occupation-measure, surrogate-objective, clipping, and trust-region notation used in the paper.

Notation	Definition
Ω	Space of infinite trajectories.
ω	A trajectory, written as $\omega = (s_0, a_0, s_1, a_1, \ldots)$.
$\mathcal{F}$	Trajectory σ-algebra on Ω.
$\mathbb{P}_\mu^\pi$	Trajectory law induced by initial distribution μ, transition kernel P, and policy π.
$\mathcal{S}$	State space.
$\mathcal{A}$	Action space.
$\mathcal{B}(\mathcal{S})$	Borel σ-algebra on the state space.
$\mathcal{B}(\mathcal{A})$	Borel σ-algebra on the action space.
B	A measurable subset of $\mathcal{S}$, usually $B \in \mathcal{B}(\mathcal{S})$.
$\mathbf{1}_B$	Indicator function of the set B.
S_t	State coordinate map at time t.
A_t	Action coordinate map at time t.
$\mathcal{F}_t$	Natural filtration $\sigma(S_0, A_0, \ldots, S_t, A_t)$.
μ	Initial state distribution.
$P(\cdot \mid s, a)$	Environment transition kernel given state-action pair (s, a).
$r(s, a)$	Bounded reward function.
$R_{\max}$	Uniform reward bound satisfying $\|r\|_\infty \le R_{\max}$.
γ	Discount factor, with $\gamma \in [0, 1)$.
π	Reference policy, typically the behavior or old policy.
π'	Candidate updated policy.
π_θ	Current parameterized policy being optimized.
$\pi_{\theta_{\text{old}}}$	Previous parameterized policy used to collect data.
π_{θ_k}	Policy at iteration k of the algorithm.
P^π	State transition kernel induced by policy π.
P_t^π	t-step transition kernel induced by policy π.
P_π	Matrix or operator form of the policy-induced transition kernel.
$V^\pi(s)$	Value function of policy π at state s.
$Q^\pi(s, a)$	Action-value function of policy π at state-action pair (s, a).
$A^\pi(s, a)$	Advantage function, $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$.
d_π	Discounted state-occupation measure induced by π.
$\mathcal{J}(\pi)$	Expected discounted return of policy π from the initial distribution.
$r_{\pi'}(s, a)$	Policy likelihood ratio $\pi'(a \mid s)/\pi(a \mid s)$.
$r_\theta(s, a)$	Parameterized likelihood ratio $\pi_\theta(a \mid s)/\pi_{\theta_{\text{old}}}(a \mid s)$.
n	Symphony transformation order.
$f_n(r)$	Symphony power-law transform, $f_n(r) = nr^{1/n} - (n-1)$.
$f_n'(r)$	First derivative of the Symphony transform.
$f_n''(r)$	Second derivative of the Symphony transform.
ν	Generic probability measure on $\mathcal{S}$.
$M_n(\nu)$	Primary Symphony term evaluated under state measure ν.
$R_n(\nu)$	Higher-order residual term from the binomial expansion.

Notation	Definition
$\mathcal{L}_n(\pi')$	Auxiliary surrogate retaining both M_n and R_n.
$L_n(\pi')$	Optimized Symphony surrogate retaining only the primary term M_n.
$L_n(\theta)$	Empirical Symphony objective for the parameterized policy π_θ.
$L_n^{\mathrm{clip}}(\theta)$	Clipped Symphony objective using the clipped likelihood ratio.
$L_{n,\mathrm{KL}}$	Local KL-regularized Symphony objective.
ρ_n	Uniform bound on the root-ratio displacement $\lvert r_{\pi'}(s,a)^{1/n} - 1\rvert$.
$A_{\max}$	Uniform bound on the absolute advantage magnitude.
ϵ_n	Bound on the statewise primary-term magnitude used in the localization-error bound.
D_{TV}	Total variation distance.
D_{KL}	Kullback–Leibler divergence.
$\mathcal{K}_\delta$	KL trust-region constraint set with radius δ.
δ	Trust-region radius; locally also used as a small ratio perturbation when writing $r = 1 + \delta$.
λ	Lagrange multiplier or KL-penalty coefficient.
λ_{sym}	KL-penalty coefficient after accounting for Symphony's intrinsic curvature.
$\mathfrak{L}(\theta, \lambda)$	Lagrangian for the KL-constrained Symphony objective.
F	Fisher information matrix induced by the old policy.
g_n	Local gradient of the Symphony objective at θ_{old}.
$\Delta\theta$	Trust-region policy-parameter update direction.
θ_0	Initial policy parameters in the algorithm.
ϕ_0	Initial value-function parameters in the algorithm.
θ_k	Policy parameters at iteration k.
ϕ_k	Value-function parameters at iteration k.
$\mathcal{D}_k$	Rollout dataset collected using π_{θ_k}.
$\hat{R}_t$	Empirical return estimate at time t.
$\hat{A}_t$	Empirical advantage estimate at time t.
ϵ	Clipping range used in the clipped Symphony objective.
v	Algorithm variant, with $v \in \{\mathrm{KL}, \mathrm{Clipped}\}$.
$\ell_t(\theta)$	Log-ratio $\log \pi_\theta(a_t \mid s_t) - \log \pi_{\theta_k}(a_t \mid s_t)$.
$s_t(\theta)$	Numerically stable Symphony ratio transform, $n\ \mathrm{expm1}(\ell_t(\theta)/n) + 1$.
$\bar{\ell}_t(\theta)$	Clipped log-ratio used in the clipped variant.
$\bar{s}_t(\theta)$	Symphony transform applied to the clipped log-ratio.
$q_t(\theta)$	Per-sample surrogate term used in the policy update.
$\mathrm{clip}(\cdot)$	Clipping operator applied to the likelihood ratio or log-ratio.
$\mathrm{expm1}(\cdot)$	Numerically stable function computing $\exp(\cdot) - 1$.

B.2 MDP Trajectory Distribution and Controlled Markov Property

An infinite-horizon Markov Decision Process is defined on trajectory space $\Omega = (\mathcal{S} \times \mathcal{A})^{\mathbb{N}}$ equipped with the product σ-algebra $\mathcal{F} = \bigotimes_{t=0}^{\infty}(\mathcal{B}(\mathcal{S}) \otimes \mathcal{B}(\mathcal{A}))$. The initial state distribution $\mu \in \Delta(\mathcal{S})$, Markov policy kernel $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, and environment transition kernel $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ uniquely induce a probability measure $\mathbb{P}_\mu^\pi$ on $(\Omega, \mathcal{F})$ such that $S_0 \sim \mu$, $A_t \mid S_t \sim \pi(\cdot \mid S_t)$, and $S_{t+1} \mid \mathcal{F}_t \sim P(\cdot \mid S_t, A_t)$, satisfying the standard controlled Markov property for history filtration $\mathcal{F}_t = \sigma(S_0, A_0, \ldots, S_t, A_t)$. Under this law, the discounted state visitation measure is $d_\pi(B) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \mathbb{P}_\mu^\pi(S_t \in B)$, and the expected return is $\mathcal{J}(\pi) = \mathbb{E}_\mu^\pi\left[\sum_{t=0}^{\infty}\gamma^t r(S_t, A_t)\right]$.

B.3 Performance Difference Lemma

The Symphony objective starts from the performance-difference lemma. Intuitively, this lemma says that the improvement from replacing π by π' can be measured by asking a simple question: under the states visited by the new policy π', how much better or worse are the actions chosen by π' according to the old advantage function A^π?

This is useful because $A^\pi(s,a)$ measures whether action a is better than the average action under the reference policy π at state s. If π' tends to place more mass on actions with positive advantage, the return should improve. The lemma makes this statement exact.

Proposition. For any two policies π and π',

$$\mathcal{J}(\pi') - \mathcal{J}(\pi) = \frac{1}{1-\gamma} \int_{\mathcal{S}} d_{\pi'}(ds) \int_{\mathcal{A}} A^\pi(s,a)\pi'(da \mid s). \tag{7}$$

Proof. The advantage can be written in one-step form as: $A^\pi(s,a) = r(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')P(ds' \mid s,a) - V^\pi(s)$. This identity says that the advantage of (s,a) is the immediate reward plus the discounted value of the next state, minus the baseline value $V^\pi(s)$. Now take expectation under trajectories generated by π'. At time t,

$$\mathbb{E}_{\pi'}[A^\pi(S_t,A_t)] = \mathbb{E}_{\pi'}[r(S_t,A_t)] + \gamma \mathbb{E}_{\pi'}[V^\pi(S_{t+1})] - \mathbb{E}_{\pi'}[V^\pi(S_t)]. \tag{8}$$

Multiplying by γ^t and summing over $t \geq 0$ gives a telescoping sum. The positive term involving $V^\pi(S_{t+1})$ at one time step cancels the negative term involving $V^\pi(S_t)$ at the next time step. Thus,

$$\sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi'}[A^\pi(S_t,A_t)] = \mathcal{J}(\pi') - \mathbb{E}_{S_0\sim\mu}[V^\pi(S_0)]. \tag{9}$$

Since $\mathcal{J}(\pi) = \mathbb{E}_{S_0\sim\mu}[V^\pi(S_0)]$, this becomes

$$\sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi'}[A^\pi(S_t,A_t)] = \mathcal{J}(\pi') - \mathcal{J}(\pi). \tag{10}$$

Finally, the discounted occupation measure rewrites the discounted time-sum as an expectation over states sampled from $d_{\pi'}$:

$$\sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi'}[A^\pi(S_t,A_t)] = \frac{1}{1-\gamma} \int_{\mathcal{S}} d_{\pi'}(ds) \int_{\mathcal{A}} A^\pi(s,a)\pi'(da \mid s). \tag{11}$$

Combining the two displays proves the result. The factor $(1-\gamma)^{-1}$ appears because $d_{\pi'}$ is normalized to be a probability measure.

B.4 Resolvent Norm and State-Distribution Localization

The exact performance-difference identity uses $d_{\pi'}$, the discounted state distribution of the updated policy. In practice, however, we optimize using data collected from the old policy π, so the practical surrogate is localized at d_π. This is the usual on-policy approximation: we use samples from the current policy to estimate an update, rather than requiring samples from every candidate new policy.

The key question is therefore: how much error do we create by replacing $d_{\pi'}$ with d_π? The answer is controlled by two ingredients. First, if π' is close to π, then the induced transition kernels $P_{\pi'}$ and P_π are close. Second, the discounted resolvent $(I-\gamma P_\pi)^{-1}$ controls how one-step transition errors accumulate over time.

Following standard perturbation results for discounted Markov decision processes (Kakade & Langford, 2002; Schulman et al., 2015), the discounted resolvent operator $(I-\gamma P_\pi)^{-1} = \sum_{k=0}^{\infty} \gamma^k P_\pi^k$ has total-variation operator norm:

$$\left\|(I-\gamma P_\pi)^{-1}\right\|_{1\to 1} = \frac{1}{1-\gamma}, \tag{12}$$

which captures the maximum cumulative propagation of one-step transition perturbations under geometric discounting.

Proposition. Let $h_n(s) = \mathbb{E}_{a\sim\pi}\left[f_n(r_{\pi'}(s,a))A^\pi(s,a)\right], \quad \epsilon_n = \max_s |h_n(s)| < \infty.$ Then

$$|M_n(d_{\pi'}) - M_n(d_\pi)| \leq \frac{2\gamma\epsilon_n}{1-\gamma} \max_s D_{\mathrm{TV}}(\pi'(\cdot \mid s), \pi(\cdot \mid s)). \tag{13}$$

Consequently, by Pinsker's inequality,

$$|M_n(d_{\pi'}) - M_n(d_\pi)| \le \frac{\gamma\epsilon_n}{1-\gamma} \max_s \sqrt{2D_{\mathrm{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s))}. \tag{14}$$

Proof. The function $h_n(s)$ is the statewise contribution of the Symphony primary term. Therefore, $M_n(d_{\pi'}) - M_n(d_\pi) = \int_{\mathcal{S}} h_n(s)(d_{\pi'} - d_\pi)(ds)$. Since $|h_n(s)| \le \epsilon_n$, we obtain $|M_n(d_{\pi'}) - M_n(d_\pi)| \le \epsilon_n \|d_{\pi'} - d_\pi\|_1$. So it remains to bound $\|d_{\pi'} - d_\pi\|_1$.
The discounted occupation measure satisfies the fixed-point identity $d_\pi^\top = (1-\gamma)\mu^\top + \gamma d_\pi^\top P_\pi$. Subtracting the fixed-point identities for π' and π gives $d_{\pi'}^\top - d_\pi^\top = \gamma(d_{\pi'}^\top - d_\pi^\top)P_\pi + \gamma d_{\pi'}^\top(P_{\pi'} - P_\pi)$. Rearranging,

$$d_{\pi'}^\top - d_\pi^\top = \gamma d_{\pi'}^\top (P_{\pi'} - P_\pi)(I - \gamma P_\pi)^{-1}. \tag{15}$$

Taking norms and applying the resolvent norm identity yields $\|d_{\pi'} - d_\pi\|_1 \le \frac{\gamma}{1-\gamma}\|P_{\pi'} - P_\pi\|_{1\to 1}$.
Now we relate the transition-kernel difference to the policy difference. For every $B \in \mathcal{B}(\mathcal{S})$, $P_{\pi'}(B \mid s) - P_\pi(B \mid s) = \int_{\mathcal{A}} P(B \mid s, a)\{\pi'(da \mid s) - \pi(da \mid s)\}$. Because $0 \le P(B \mid s, a) \le 1$, the induced transition perturbation is bounded by the policy perturbation: $\|P_{\pi'}(\cdot \mid s) - P_\pi(\cdot \mid s)\|_1 \le 2D_{\mathrm{TV}}(\pi'(\cdot \mid s), \pi(\cdot \mid s))$. Therefore,

$$\|d_{\pi'} - d_\pi\|_1 \le \frac{2\gamma}{1-\gamma} \max_s D_{\mathrm{TV}}(\pi'(\cdot \mid s), \pi(\cdot \mid s)). \tag{16}$$

Multiplying by ϵ_n gives the total-variation localization bound. Finally, Pinsker's inequality, $D_{\mathrm{TV}}(\pi'(\cdot \mid s), \pi(\cdot \mid s)) \le \sqrt{\frac{1}{2}D_{\mathrm{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s))}$, gives the KL version. This explains why clipping and KL trust regions are natural companions to Symphony: they keep the policy perturbation small, which keeps the localized surrogate close to the exact $d_{\pi'}$ expression.

B.5 Algebraic Properties of the Symphony Transform

We now isolate the theoretical foundations and algebra specific to Symphony. Rather than an ad-hoc algebraic coincidence, the functional form $f_n(r) = nr^{1/n} - (n-1)$ arises uniquely from three fundamental consistency and stability principles:

1. **Axiom 1 (Objective Value Preservation):** At $r = 1$ (the behavior policy), the surrogate must preserve objective value: $f(1) = 1$.
2. **Axiom 2 (First-Order Policy Gradient Alignment):** At $r = 1$, the surrogate derivative must equal unity: $f'(1) = 1$, guaranteeing exact gradient equivalence with the true policy gradient at θ_{old}.
3. **Axiom 3 (Sublinear Power-Law Attenuation):** To attenuate heavy-tailed ratio excursions, the surrogate must grow sublinearly as r^α with power-law exponent $\alpha = 1/n \in (0, 1)$ for continuous order $n > 1$.

Solving the parametric form $f(r) = ar^{1/n} + b$ subject to Axioms 1 and 2 uniquely yields:

$$a \cdot \frac{1}{n} = 1 \implies a = n, \qquad a + b = 1 \implies b = -(n-1), \tag{17}$$

which uniquely specifies $f_n(r) = nr^{1/n} - (n-1)$ for any continuous float $n > 1$ (as parameterized in code).

For integer orders $n \ge 2$, this functional form admits an exact structural connection to the binomial expansion: setting $x_n = r^{1/n} - 1$, the identity $r = (1 + x_n)^n$ reveals that $f_n(r)$ is the first-order truncation of r, isolating the primary sublinear term from a strictly second-order residual.

Proposition B.5 (Axiomatic Properties and Residual Bounds). For order $n > 1$ and likelihood ratio $r > 0$, let $f_n(r) = nr^{1/n} - (n-1)$. Then:

1. For integer $n \ge 2$, the likelihood ratio admits the exact algebraic decomposition:

$$r = f_n(r) + \sum_{k=2}^{n} \binom{n}{k} (r^{1/n} - 1)^k. \tag{18}$$

2. Let $\bar{\rho} > 0$ define an arbitrary compact neighborhood $[0, \bar{\rho}]$. If $|r_{\pi'}(s,a)^{1/n} - 1| \le \rho_n \le \bar{\rho}$ and $|A^\pi(s,a)| \le A_{\max}$ on the sampled support, then for fixed integer $n \ge 2$:

$$|\mathcal{L}_n(\pi') - L_n(\pi')| \le \frac{A_{\max}}{1-\gamma} C_{n,\bar{\rho}}\, \rho_n^2 = O_n(\rho_n^2) \quad \text{as } \rho_n \to 0, \tag{19}$$

where $C_{n,\bar{\rho}} = \sum_{k=2}^{n} \binom{n}{k} \bar{\rho}^{k-2} < \infty$. In particular:

- For the default order $n = 2$, $C_{2,\bar{\rho}} \equiv 1$ identically for all $\bar{\rho} \ge 0$, yielding a globally quadratic bound.
- For $\bar{\rho} = 1$ (corresponding to $r \in (0, 2^n]$), $C_{n,1} = 2^n - n - 1$.

3. At $\theta = \theta_{\text{old}}$, the Symphony surrogate has the exact same first-order policy-gradient direction as the standard likelihood-ratio surrogate:

$$\nabla_\theta L_n(\theta)\big|_{\theta=\theta_{\text{old}}} = \mathbb{E}_{s\sim d_\pi, a\sim\pi}\left[\nabla_\theta \log \pi_\theta(a \mid s) A^\pi(s,a)\right]. \tag{20}$$

4. The trace second-moment matrix of the stochastic gradient satisfies $\operatorname{Tr}\left(\mathbb{E}[\hat{g}_n \hat{g}_n^\top]\right) \le C \cdot \mathbb{E}\left[r_\theta^{2/n}\right]$ for a problem-dependent constant $C < \infty$, compressing ratio-driven moment growth from quadratic $\mathcal{O}(r^2)$ at $n = 1$ to sublinear $\mathcal{O}(r^{2/n})$ for general n, and to linear $\mathcal{O}(r)$ for $n = 2$.

Proof. Let $x = r^{1/n} - 1$. Then $r = (1+x)^n$. For integer $n \ge 2$, by the finite binomial theorem,

$$r = 1 + nx + \sum_{k=2}^{n} \binom{n}{k} x^k = n r^{1/n} - (n-1) + \sum_{k=2}^{n} \binom{n}{k} (r^{1/n} - 1)^k. \tag{21}$$

This proves the exact decomposition (Item 1). The first part, $nr^{1/n} - (n-1)$, is precisely $f_n(r)$; the remaining terms form the higher-order residual R_n.
The auxiliary objective $\mathcal{L}_n$ retains the full binomial expansion, while L_n retains only the primary Symphony term. Hence,

$$\mathcal{L}_n(\pi') - L_n(\pi') = \frac{1}{1-\gamma} R_n(d_\pi). \tag{22}$$

Factoring ρ_n^2 from the residual summation gives:

$$\sum_{k=2}^{n} \binom{n}{k} \rho_n^k = \rho_n^2 \sum_{k=2}^{n} \binom{n}{k} \rho_n^{k-2} \le \rho_n^2 \sum_{k=2}^{n} \binom{n}{k} \bar{\rho}^{k-2} = C_{n,\bar{\rho}}\, \rho_n^2. \tag{23}$$

Using the uniform bound on advantage magnitude $|A^\pi(s,a)| \le A_{\max}$, we obtain:

$$|\mathcal{L}_n(\pi') - L_n(\pi')| \le \frac{A_{\max}}{1-\gamma} \sum_{k=2}^{n} \binom{n}{k} \rho_n^k \le \frac{A_{\max}}{1-\gamma} C_{n,\bar{\rho}}\, \rho_n^2. \tag{24}$$

For any fixed $n \ge 2$, $C_{n,\bar{\rho}}$ is a finite constant on the compact interval $\rho_n \in [0, \bar{\rho}]$, proving that the residual scales strictly as $O_n(\rho_n^2)$ as $\rho_n \to 0$ (Item 2).
For the default choice $n = 2$, the residual sum collapses to a single term ($k = 2$):

$$(1+\rho_2)^2 - 1 - 2\rho_2 = \binom{2}{2} \rho_2^2 = \rho_2^2. \tag{25}$$

Thus $C_{2,\bar{\rho}} = 1$ globally for all $\rho_2 \ge 0$, and $|\mathcal{L}_2(\pi') - L_2(\pi')| \le \frac{A_{\max}}{1-\gamma} \rho_2^2$ holds globally without restricting ρ_2 to a bounded neighborhood.
Next, $f_n'(r) = r^{1/n - 1}$ and $\nabla_\theta r_\theta(s,a) = r_\theta(s,a) \nabla_\theta \log \pi_\theta(a \mid s)$. Therefore,

$$\nabla_\theta f_n(r_\theta(s,a)) = f_n'(r_\theta) \nabla_\theta r_\theta(s,a) = r_\theta(s,a)^{1/n} \nabla_\theta \log \pi_\theta(a \mid s). \tag{26}$$

At $\theta = \theta_{\text{old}}$, $r_\theta(s,a) = 1$, which gives $f_n'(1) = 1$ and:

$$\nabla_\theta f_n(r_\theta(s,a))\big|_{\theta=\theta_{\text{old}}} = \nabla_\theta \log \pi_\theta(a \mid s)\big|_{\theta=\theta_{\text{old}}}. \tag{27}$$

Thus, the gradient of the unclipped empirical objective $L_n(\theta)$ at θ_{old} is:

$$\nabla_\theta L_n(\theta)\big|_{\theta=\theta_{\text{old}}} = \mathbb{E}_{s\sim d_{\pi_{\theta_{\text{old}}}}, a\sim\pi_{\theta_{\text{old}}}} \left[\nabla_\theta \log \pi_\theta(a \mid s) A^{\pi_{\theta_{\text{old}}}}(s,a)\right], \tag{28}$$

which is the exact standard policy-gradient direction (Item 3).
Finally, because the stochastic policy gradient is $\hat{g}_n(s,a) = r_\theta(s,a)^{1/n}\nabla_\theta \log \pi_\theta(a \mid s) A^\pi(s,a)$, the outer product satisfies:

$$\hat{g}_n(s,a)\hat{g}_n(s,a)^\top = r_\theta(s,a)^{2/n}(A^\pi(s,a))^2 \nabla_\theta \log \pi_\theta(a \mid s) \nabla_\theta \log \pi_\theta(a \mid s)^\top. \tag{29}$$

Taking the trace and assuming $C := \max_{s,a} \|\nabla_\theta \log \pi_\theta(a \mid s)\|_2^2 (A^\pi(s,a))^2 < \infty$, we obtain:

$$\operatorname{Tr}\left(\mathbb{E}\left[\hat{g}_n \hat{g}_n^\top\right]\right) \le C \cdot \mathbb{E}_{s,a}\left[r_\theta(s,a)^{2/n}\right]. \tag{30}$$

While total gradient variance depends jointly on advantage estimates and score function norms, Symphony specifically bounds the ratio-dependent growth rate of the trace second-moment matrix from quadratic $\mathcal{O}(r^2)$ (at $n=1$) to sublinear $\mathcal{O}(r^{2/n})$ for general n, and to linear $\mathcal{O}(r)$ for $n=2$, directly mitigating heavy-tailed gradient outliers in continuous control (Item 4).
The local negative curvature follows from $f_n''(r) = \frac{1-n}{n} r^{1/n-2}$. At $r=1$, $f_n''(1) = -(n-1)/n$, yielding the second-order expansion:

$$f_n(1+\delta) = 1 + \delta - \frac{n-1}{2n}\delta^2 + O(\delta^3). \tag{31}$$

Thus, Symphony preserves the first-order direction while introducing negative curvature that dampens large finite-ratio moves.

Operating Regime and Coordinate Contraction. We briefly address the practical regime $\rho_n \ll 1$. In on-policy policy optimization, updates are initialized at the behavior policy where $r=1$ and $\rho_n = 0$. While proximal clipping removes the gradient incentive to push ratios beyond $[1-\epsilon, 1+\epsilon]$ along advantageous directions, it does not strictly enforce $|r-1| \le \epsilon$ as a hard coordinate constraint (unfavorable directions remain active outside the interval). Nevertheless, under the operating condition or local trust-region assumption that iterates remain within a neighborhood $r \in [1-\epsilon, 1+\epsilon]$, controlling higher-order terms uniformly across n requires bounding the coordinate displacement $\rho_n = \max_{r\in[1-\epsilon,1+\epsilon]} |r^{1/n} - 1|$. For $t \in [1-\epsilon, 1+\epsilon]$ with $\epsilon \in (0,1)$, the derivative of $g(t) = t^{1/n}$ satisfies $|g'(t)| = \frac{1}{n} t^{1/n-1} \le \frac{1}{n(1-\epsilon)^{(n-1)/n}} \le \frac{C_\epsilon}{n}$, where $C_\epsilon = \frac{1}{1-\epsilon}$. Because $t \mapsto t^{1/n}$ is strictly concave, the maximum absolute displacement on $[1-\epsilon, 1+\epsilon]$ is achieved at the lower boundary $r = 1-\epsilon$:

$$\rho_n = 1 - (1-\epsilon)^{1/n} \le \frac{C_\epsilon \epsilon}{n}. \tag{32}$$

Substituting $\rho_n \le \frac{C_\epsilon \epsilon}{n}$ into the binomial residual expansion yields:

$$\sum_{k=2}^{n} \binom{n}{k} \rho_n^k \le \sum_{k=2}^{n} \frac{n^k}{k!} \left(\frac{C_\epsilon \epsilon}{n}\right)^k = \sum_{k=2}^{n} \frac{(C_\epsilon \epsilon)^k}{k!} \le e^{C_\epsilon \epsilon} - 1 - C_\epsilon \epsilon = O(\epsilon^2). \tag{33}$$

Consequently, the n-th root coordinate contracts policy-ratio deviations by $O(1/n)$. This guarantees that the total omitted residual remains strictly of order $O(\epsilon^2)$ and is uniformly bounded for all orders $n \ge 2$.

B.6 Analytical Residual and Localization Bounds on the Surrogate

We now formally combine the performance difference identity (Section B.3), the state-distribution localization bound (Section B.4), and the algebraic residual expansion of Symphony (Section B.5) to derive the analytical residual and localization bound of Equation (4) and quantify the approximation error of the Symphony surrogate.

Proposition B.6 (Analytical Residual and Localization Bound). Let π and π' be any two policies satisfying $\pi'(\cdot \mid s) \ll \pi(\cdot \mid s)$ on $\operatorname{supp}(d_{\pi'})$, with discounted state-action visitation

distributions d_π and $d_{\pi'}$, and for integer $n \geq 2$, let $\epsilon_n = \max_s |\mathbb{E}_{a\sim\pi}[f_n(r_{\pi'}(s,a))A^\pi(s,a)]| < \infty$. Then:

1. **Analytical Error and Localization Bound (Equation (4)):** The approximation error between the true performance $\mathcal{J}(\pi')$ and the localized Symphony surrogate satisfies the two-sided inequality:

$$\left|\mathcal{J}(\pi') - \left(\mathcal{J}(\pi) + \frac{1}{1-\gamma}M_n(d_\pi)\right)\right| \leq \Delta_{\text{pen}}(\pi', \pi), \tag{34}$$

where the compound divergence and residual penalty is defined as:

$$\Delta_{\text{pen}}(\pi', \pi) = \frac{\gamma\epsilon_n}{(1-\gamma)^2}\max_s \sqrt{2D_{\text{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s))} + \frac{1}{1-\gamma}|R_n(d_{\pi'})|, \tag{35}$$

with $M_n(d_\pi) = \mathbb{E}_{s\sim d_\pi, a\sim\pi}[f_n(r_{\pi'}(s,a))A^\pi(s,a)]$ and $R_n(d_{\pi'})$ the binomial residual.

2. **Analytical Minorizer and Lower Bound:** Consequently, the true policy performance admits the analytical minorization:

$$\mathcal{J}(\pi') \geq \mathcal{J}(\pi) + \frac{1}{1-\gamma}M_n(d_\pi) - \Delta_{\text{pen}}(\pi', \pi). \tag{36}$$

While an idealized policy update satisfying $\frac{1}{1-\gamma}M_n(d_\pi) \geq \Delta_{\text{pen}}(\pi', \pi)$ analytically guarantees $\mathcal{J}(\pi') \geq \mathcal{J}(\pi)$, the penalty $\Delta_{\text{pen}}(\pi', \pi)$ depends on the unknown future policy distribution $d_{\pi'}$ through $R_n(d_{\pi'})$ and cannot be computed or minimized directly during execution; practical updates optimize the clipped surrogate via first-order stochastic gradient methods without verifying or enforcing this condition pointwise.

3. **Upper Bound (Reverse Direction):** Symmetrically, the reverse inequality holds:

$$\mathcal{J}(\pi') \leq \mathcal{J}(\pi) + \frac{1}{1-\gamma}M_n(d_\pi) + \Delta_{\text{pen}}(\pi', \pi), \tag{37}$$

establishing that the true performance $\mathcal{J}(\pi')$ is tightly sandwiched both ways around the localized Symphony surrogate $\mathcal{J}(\pi) + \frac{1}{1-\gamma}M_n(d_\pi)$.

Proof. We prove each statement:

Step 1: Derivation of Equation (34)**.** By the Performance Difference Lemma (Proposition B.3),

$$\mathcal{J}(\pi') - \mathcal{J}(\pi) = \frac{1}{1-\gamma}\int_{\mathcal{S}} d_{\pi'}(ds)\int_{\mathcal{A}} \pi'(da \mid s)A^\pi(s,a). \tag{38}$$

Rewriting the action integral under $\pi(\cdot \mid s)$ via the likelihood ratio $r(s,a) = \pi'(a \mid s)/\pi(a \mid s)$:

$$\mathcal{J}(\pi') - \mathcal{J}(\pi) = \frac{1}{1-\gamma}\int_{\mathcal{S}} d_{\pi'}(ds)\int_{\mathcal{A}} \pi(da \mid s)\, r(s,a)A^\pi(s,a). \tag{39}$$

Substituting the exact algebraic decomposition of Symphony (Proposition B.5, Item 1), $r(s,a) = f_n(r(s,a)) + R_n(s,a)$, we obtain:

$$\begin{aligned}\mathcal{J}(\pi') - \mathcal{J}(\pi) &= \frac{1}{1-\gamma}\int_{\mathcal{S}} d_{\pi'}(ds)\int_{\mathcal{A}} \pi(da \mid s)\left[f_n(r(s,a)) + R_n(s,a)\right]A^\pi(s,a) \\ &= \frac{1}{1-\gamma}M_n(d_{\pi'}) + \frac{1}{1-\gamma}R_n(d_{\pi'}).\end{aligned} \tag{40}$$

Subtracting the localized surrogate $\frac{1}{1-\gamma}M_n(d_\pi)$ from both sides gives:

$$\mathcal{J}(\pi') - \left(\mathcal{J}(\pi) + \frac{1}{1-\gamma}M_n(d_\pi)\right) = \frac{1}{1-\gamma}\left(M_n(d_{\pi'}) - M_n(d_\pi)\right) + \frac{1}{1-\gamma}R_n(d_{\pi'}). \tag{41}$$

Applying the triangle inequality:

$$\left|\mathcal{J}(\pi') - \left(\mathcal{J}(\pi) + \frac{1}{1-\gamma}M_n(d_\pi)\right)\right| \leq \frac{1}{1-\gamma}|M_n(d_{\pi'}) - M_n(d_\pi)| + \frac{1}{1-\gamma}|R_n(d_{\pi'})|. \tag{42}$$

By the state-distribution localization bound (Proposition B.4),

$$|M_n(d_{\pi'}) - M_n(d_\pi)| \le \frac{\gamma\epsilon_n}{1-\gamma} \max_s \sqrt{2D_{\mathrm{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s))}. \tag{43}$$

Substituting this bound immediately yields Equation (34):

$$\begin{aligned}\left|\mathcal{J}(\pi') - \left(\mathcal{J}(\pi) + \frac{1}{1-\gamma} M_n(d_\pi)\right)\right| \le \frac{\gamma\epsilon_n}{(1-\gamma)^2} \max_s \sqrt{2D_{\mathrm{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s))} \\ + \frac{1}{1-\gamma}|R_n(d_{\pi'})| = \Delta_{\mathrm{pen}}(\pi', \pi).\end{aligned} \tag{44}$$

Step 2: Lower Bound and Monotonic Policy Improvement. Expanding the absolute value inequality $|X - Y| \le \Delta_{\mathrm{pen}}$ gives the two-sided relation:

$$-\Delta_{\mathrm{pen}}(\pi', \pi) \le \mathcal{J}(\pi') - \left(\mathcal{J}(\pi) + \frac{1}{1-\gamma} M_n(d_\pi)\right) \le \Delta_{\mathrm{pen}}(\pi', \pi). \tag{45}$$

From the lower bound,

$$\mathcal{J}(\pi') \ge \mathcal{M}_n(\pi'; \pi) := \mathcal{J}(\pi) + \frac{1}{1-\gamma} M_n(d_\pi) - \Delta_{\mathrm{pen}}(\pi', \pi). \tag{46}$$

At $\pi' = \pi$, we verify:

- $r(s,a) = 1 \implies f_n(1) = 1$. Since advantages satisfy $\mathbb{E}_{a\sim\pi}[A^\pi(s,a)] = 0$, $M_n(d_\pi) = \int_{\mathcal{S}} d_\pi(ds) \int_{\mathcal{A}} \pi(da \mid s)[1 \cdot A^\pi(s,a)] = 0$.
- $D_{\mathrm{KL}}(\pi(\cdot \mid s) \,\|\, \pi(\cdot \mid s)) = 0$ for all s.
- $R_n(s,a) = \sum_{k=2}^n \binom{n}{k}(1^{1/n} - 1)^k = 0 \implies R_n(d_\pi) = 0$.

Therefore, $\Delta_{\mathrm{pen}}(\pi, \pi) = 0$, establishing that the lower bound is exact at the current policy:

$$\mathcal{M}_n(\pi; \pi) = \mathcal{J}(\pi). \tag{47}$$

Hence $\mathcal{M}_n(\pi'; \pi)$ is a valid analytical minorizer of $\mathcal{J}(\pi')$. Whenever an idealized candidate policy π' satisfies $\frac{1}{1-\gamma} M_n(d_\pi) \ge \Delta_{\mathrm{pen}}(\pi', \pi)$, the lower bound analytically guarantees $\mathcal{J}(\pi') \ge \mathcal{M}_n(\pi'; \pi) \ge \mathcal{M}_n(\pi; \pi) = \mathcal{J}(\pi)$.

Analytical Error Bound vs. Practical Algorithm. We emphasize that Proposition B.6 is strictly an **analytical residual and localization bound** quantifying the approximation gap between the true expected return $\mathcal{J}(\pi')$ and the localized Symphony surrogate $\mathcal{J}(\pi) + \frac{1}{1-\gamma} M_n(d_\pi)$, rather than an algorithmic monotonic convergence theorem for Algorithm 1. A reviewer or practitioner must recognize two key analytical distinctions:

1. **Residual Evaluated Under Future State Measure $d_{\pi'}$:** The penalty term $\Delta_{\mathrm{pen}}(\pi', \pi)$ contains the residual term $|R_n(d_{\pi'})|$ evaluated under the *future policy's state distribution* $d_{\pi'}$. Because $d_{\pi'}$ is unknown at optimization time, this penalty cannot be computed, verified, or minimized by Algorithm 1.
2. **First-Order Stochastic Updates Without Minorizer Enforcement:** Algorithm 1 optimizes the clipped surrogate via first-order stochastic gradient updates rather than evaluating or enforcing the theoretical minorizer bound pointwise.

Consequently, Proposition B.6 serves to establish a rigorous error-decomposition bound governing the surrogate error and residual truncation, while Algorithm 1 provides its scalable empirical realization.

Step 3: Upper Bound (Reverse Direction). From the right side of the two-sided inequality:

$$\mathcal{J}(\pi') \le \mathcal{U}_n(\pi'; \pi) := \mathcal{J}(\pi) + \frac{1}{1-\gamma} M_n(d_\pi) + \Delta_{\mathrm{pen}}(\pi', \pi). \tag{48}$$

Combining both sides yields the sandwich guarantee:

$$\mathcal{M}_n(\pi'; \pi) \le \mathcal{J}(\pi') \le \mathcal{U}_n(\pi'; \pi). \tag{49}$$

This confirms that the true return $\mathcal{J}(\pi')$ is tightly bounded both ways by the localized Symphony surrogate $M_n(d_\pi)$ and the penalty $\Delta_{\mathrm{pen}}(\pi', \pi)$.

C NONSTATIONARY BROWNIAN BANDIT DIAGNOSTIC

We introduce a controlled diagnostic game to isolate the variance mechanism studied in the main text. The environment is a nonstationary multi-state bandit with Brownian perturbations to the policy logits. It is not intended as a benchmark for long-horizon credit assignment; instead, it removes transition-induced confounds and directly tests whether Symphony suppresses likelihood-ratio-driven gradient variance while preserving return.

The game contains $|\mathcal{S}| = 6$ states and $|\mathcal{A}| = 10$ actions. Each state has a fixed weight w_s, normalized so that $\sum_s w_s = 1$. At each optimization step t, the policy logits are perturbed by Brownian noise,

$$z_{t+1}(s,a) = z_t(s,a) + \sigma \xi_t(s,a), \qquad \xi_t(s,a) \sim \mathcal{N}(0,1), \tag{50}$$

with $\sigma = 0.035$. The old policy logits are refreshed every 10 steps, creating a controlled rollout window in which likelihood ratios can drift away from 1.

The reward landscape is time-varying. For each state, the preferred action moves smoothly over time, while the reward table also includes state-dependent difficulty, global action bonuses, a moving trap, and state-wise cyclic shifts. We then smooth the reward table over neighboring actions and neighboring states. This produces a nonstationary but structured reward process: the optimal action changes over time, but the landscape remains locally coherent.

Given the reward table $R_t(s,a)$, we construct noisy advantages independently across seeds:

$$\widetilde{A}_t^{(i)}(s,a) = R_t(s,a) + 0.35\, \eta_t^{(i)}(s,a), \qquad \eta_t^{(i)}(s,a) \sim \mathcal{N}(0,1). \tag{51}$$

For each seed and state, advantages are centered and normalized across actions:

$$A_t^{(i)}(s,a) = \frac{\widetilde{A}_t^{(i)}(s,a) - \frac{1}{|\mathcal{A}|}\sum_{a'} \widetilde{A}_t^{(i)}(s,a')}{\mathrm{Std}_{a'}[\widetilde{A}_t^{(i)}(s,a')] + \varepsilon_{\mathrm{num}}}. \tag{52}$$

This normalization keeps the reward scale fixed while preserving noisy action preferences.

We compare Vanilla Policy Gradient and Symphony under the same Brownian logit drift. Let z_θ and z_{old} denote the current and old logits, with associated softmax policies $\pi_\theta(a \mid s) \propto \exp(z_\theta(s,a))$ and $\pi_{\mathrm{old}}(a \mid s) \propto \exp(z_{\mathrm{old}}(s,a))$. The exact log-likelihood ratio incorporates the log-partition normalization:

$$\ell_t(s,a) = \log \frac{\pi_\theta(a \mid s)}{\pi_{\mathrm{old}}(a \mid s)} = \left(z_\theta(s,a) - \log \sum_{a'} e^{z_\theta(s,a')} \right) - \left(z_{\mathrm{old}}(s,a) - \log \sum_{a'} e^{z_{\mathrm{old}}(s,a')} \right). \tag{53}$$

Vanilla Policy Gradient uses the standard likelihood-ratio surrogate:

$$\mathcal{L}_{\mathrm{VPG}}(\theta) = -\sum_s w_s \sum_a \pi_{\mathrm{old}}(a \mid s) \exp(\ell_t(s,a)) A_t(s,a). \tag{54}$$

Symphony uses the stable log-ratio implementation of the power-law surrogate:

$$s_n(\ell) = n\ \mathrm{expm1}(\ell/n) + 1, \tag{55}$$

and optimizes

$$\mathcal{L}_{\mathrm{Sym}}(\theta) = -\sum_s w_s \sum_a \pi_{\mathrm{old}}(a \mid s) s_n(\ell_t(s,a)) A_t(s,a). \tag{56}$$

In this diagnostic, expectations across states and actions are evaluated under the behavior distribution $\sum_s w_s \sum_a \pi_{\mathrm{old}}(a \mid s)$ using the normalized softmax probabilities and log-ratios. We set $n = 2$, so the surrogate corresponds to $2\sqrt{r} - 1$ in ratio space, evaluated via the numerically stable log-ratio form $s_2(\ell) = 2\ \mathrm{expm1}(\ell/2) + 1$. All empirical curves in Figure 2 reflect these exact policy objectives across the 32 random seeds.

We run 32 independent seeds for 160 optimization steps with learning rate 0.06. At each step, we record two quantities. First, Figure 2a reports the median, across seeds, of the within-seed dispersion of gradient components over state-action logits (this measures within-seed logit gradient dispersion in this controlled setting, and is not a minibatch-level estimator variance measure):

$$\mathrm{VarGrad}_t^{(i)} = \mathrm{Var}_{s,a}\left[\nabla_{z(s,a)} \mathcal{L}_t^{(i)} \right]. \tag{57}$$

Second, we measure the expected return under the current softmax policy:

$$J_t^{(i)} = \sum_s w_s \sum_a \pi_{\theta_t}^{(i)}(a \mid s) R_t(s, a). \tag{58}$$

This construction directly targets the phenomenon studied by Symphony. Brownian perturbations make the likelihood ratios drift, the nonstationary reward landscape creates changing advantage directions, and the injected advantage noise produces seed-level stochasticity. Both methods receive the exact same stochastic drift and noisy advantages. Symphony shows lower, tighter dispersion in this diagnostic while preserving comparable expected returns. This isolates the proposed ratio-compression effect from long-horizon credit-assignment confounders.

D Extended Empirical Evaluation

D.1 Ablation Studies

In this section, we provide the learning curves corresponding to the ablation studies discussed in Section 3.1, isolating the effects of the transformation order (n), the proximal clipping boundary (ϵ), the intrinsic curvature penalty (λ), and the learning rate.

D.1.1 Impact of Transformation Order (n)

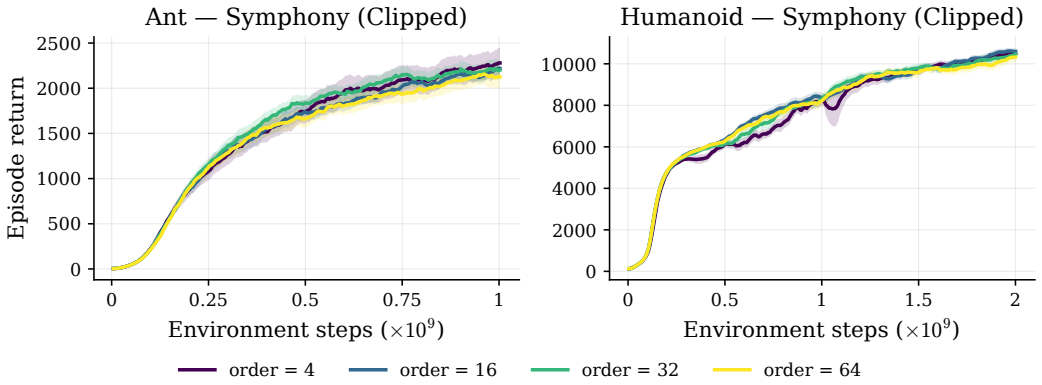


Figure 6: Impact of the transformation order (n) on sample efficiency for the Ant and Humanoid environments using Symphony (Clipped). The optimal curvature is environment-dependent: mid-complexity tasks like Ant favor lower orders ($n = 4$), while high-dimensional action spaces like Humanoid benefit from stronger damping ($n = 16$). This confirms that while curvature reduces ratio-driven variance, excessive damping can over-compress the learning signal.

Justification of Order $n = 2$ as a Practical Default. Throughout our primary evaluations, we adopt $n = 2$ as the *smallest nonlinear integer order and a practical default*, rather than claiming it is universally optimal across all environments. Order $n = 2$ represents the minimal integer power that introduces intrinsic negative curvature ($f_2''(1) = -1/2$) to attenuate large likelihood ratios and scale dominant squared-gradient variance as $\mathcal{O}(r)$ (in contrast to PPO's unattenuated $\mathcal{O}(r^2)$), while possessing an exact quadratic residual identity $r = (2\sqrt{r} - 1) + (\sqrt{r} - 1)^2$ globally.

As demonstrated by the empirical sweeps in Table 2 (Part II) and Figure 6, the preferred curvature is task-dependent:

- On Ant (8 continuous action dimensions), an intermediate order of $n = 4$ achieves peak performance (2440.52 IQM), with higher orders ($n \geq 16$) degrading returns (2174.04 at $n = 64$) due to excessive compression of usable policy gradients.

- On Humanoid (17 continuous action dimensions), where compounding variance across action dimensions is substantially more severe, stronger curvature damping yields higher returns, peaking at $n = 16$ (10560.10 IQM) before stabilizing (10217.00 at $n = 64$).

We emphasize that the multi-dimensional interactions between transformation order n, learning rate α, update epochs E, and training horizon are not exhaustively studied in this work. Nonetheless, order $n = 2$ provides a consistent, robust improvement over linear surrogates across all evaluated tasks without requiring environment-specific order tuning.

D.1.2 Sensitivity to Learning Rate

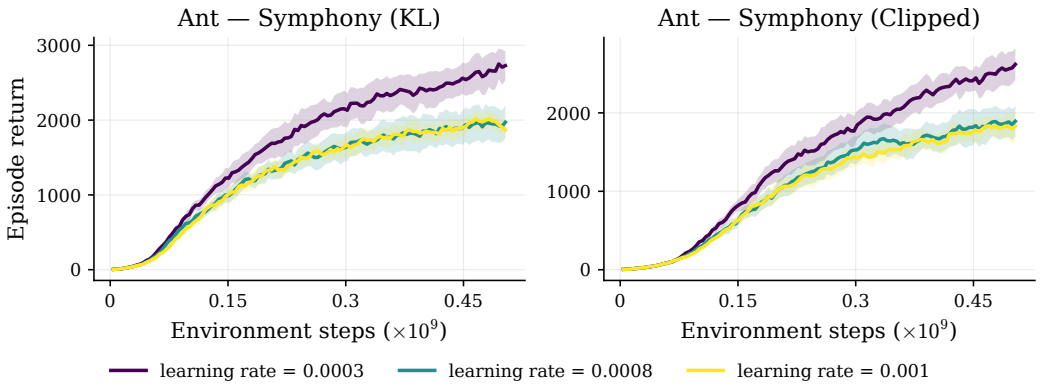


Figure 7: Sensitivity analysis of the learning rate ($\alpha \in \{0.0003, 0.0008, 0.001\}$) on the Ant environment for both standard and clipped Symphony for 4 seeds. The results demonstrate Symphony's robustness across various learning rates, with the clipped variant maintaining stable and monotonic improvements without catastrophic collapse even at higher step sizes.

D.1.3 Effect of Intrinsic Curvature Penalty

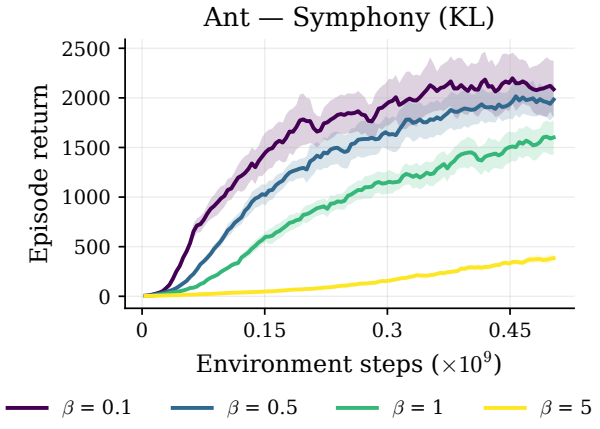


Figure 8: Effect of the intrinsic curvature penalty coefficient (λ_{sym}, parameterized as β) on the unclipped Symphony objective in the Ant environment for 8 seeds. The curves illustrate the bias-stability tradeoff: while appropriate curvature provides essential variance damping for unstable updates, excessively high values over-compress the finite-ratio improvement signal.

D.1.4 Robustness to Proximal Clipping Boundary

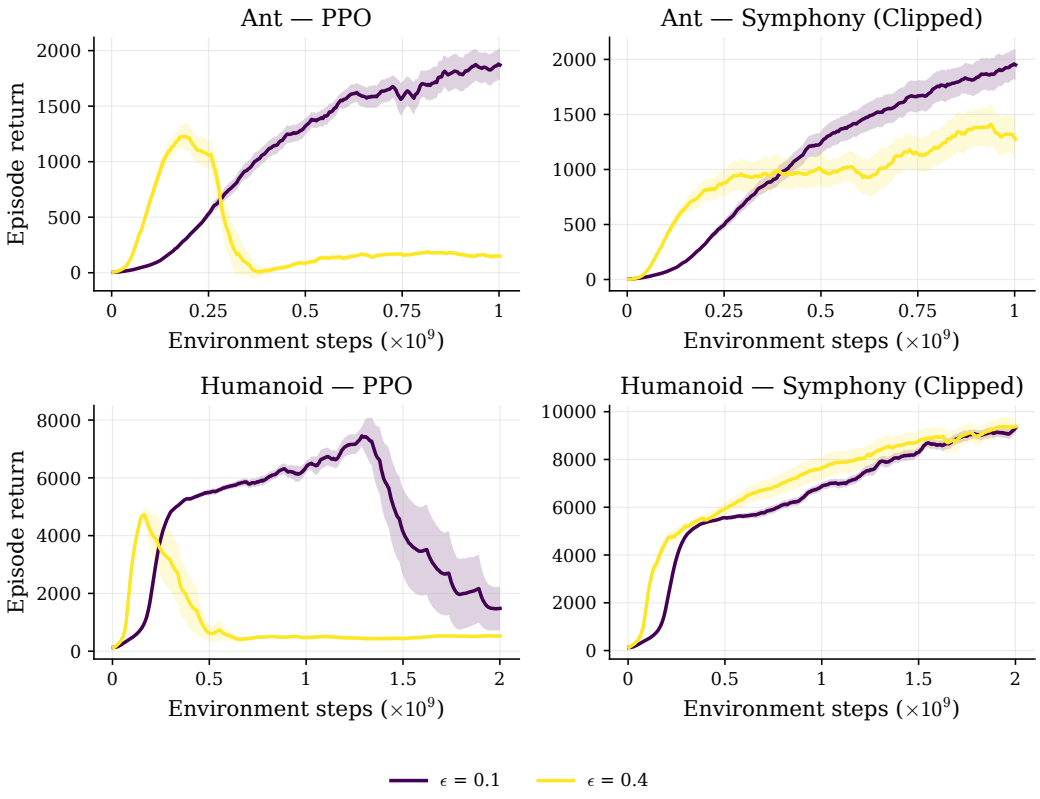


Figure 9: Robustness to the proximal clipping boundary (ϵ) for 8 seeds. Symphony demonstrates significantly greater stability than PPO when the clipping constraint is relaxed to clipping range $\epsilon = 0.4$. Because Symphony natively damps large likelihood ratios via its negatively curved surrogate geometry, it remains highly resilient even when external hard clipping boundaries are loosened.

Within-Algorithm Sensitivity Relative to Standard $\epsilon = 0.2$ Setting. In Section 3.1 and Table 2 (Part III), we reported cross-method Probability of Improvement (POI) scores across clipping thresholds $\epsilon \in \{0.1, 0.2, 0.4\}$. While cross-method POI establishes pairwise superiority at each fixed threshold, it answers a comparative question rather than evaluating each algorithm's intrinsic sensitivity to its own clipping parameter.

To directly quantify algorithmic sensitivity under clipping boundary variations, Table 4 evaluates the *within-algorithm return retention* relative to each method's own standard baseline configuration at $\epsilon = 0.2$:

$$\text{Retention}(\epsilon) = \frac{\text{IQM}(\epsilon)}{\text{IQM}(\epsilon = 0.2)} \times 100\%. \tag{59}$$

Table 4: **Within-Algorithm Clipping Sensitivity Relative to Each Method's Own $\epsilon = 0.2$ Setting.** Returns report Interquartile Mean (IQM) across 8 seeds. Retention percentages measure performance relative to each algorithm's own standard setting ($\epsilon = 0.2 = 100\%$). While PPO collapses by 85%–88% when clipping is relaxed to $\epsilon = 0.4$, Symphony preserves 92.2% on Humanoid and 57.5% on Ant, demonstrating lower sensitivity than PPO in the tested settings.

Environment	Algorithm	**Tight** ($\epsilon = 0.1$)	**Standard** ($\epsilon = 0.2$)	**Relaxed** ($\epsilon = 0.4$)
Humanoid	PPO	1047.16 (27.9%)	3749.73 (100.0%)	568.82 (15.2%)
	Symphony (Clipped)	9262.67 (91.4%)	10137.06 (100.0%)	9344.21 (92.2%)
Ant	PPO	1899.19 (135.4%)	1402.43 (100.0%)	168.82 (12.0%)
	Symphony (Clipped)	2005.43 (88.0%)	2277.88 (100.0%)	1309.47 (57.5%)

This within-algorithm analysis yields two primary insights:

1. **Severe PPO degradation under relaxed clipping ($\epsilon = 0.4$):** When the clipping threshold is widened to $\epsilon = 0.4$, PPO suffers catastrophic collapse, retaining only 15.2% of its return on Humanoid (568.82 vs. 3749.73, an 84.8% reduction) and only 12.0% on Ant (168.82 vs. 1402.43, an 88.0% collapse). Because PPO's surrogate objective is linear in the likelihood ratio r, loosening external boundary clipping permits unattenuated gradient contributions from extreme ratios, inducing destructive policy updates. In sharp contrast, Symphony (Clipped) retains 92.2% of its performance on Humanoid (9344.21, a mere 7.8% variation) and 57.5% on Ant (1309.47 vs. PPO's 168.82). Symphony's negative curvature ($f_n''(r) < 0$) inherently compresses large ratios, preventing update destabilization even when hard boundaries are relaxed.
2. **Asymmetric sensitivity under tight clipping ($\epsilon = 0.1$):** On Humanoid, overly aggressive clipping ($\epsilon = 0.1$) reduces PPO to 27.9% retention (1047.16) due to severe step-size starvation in high dimensions, whereas Symphony retains 91.4% of its baseline return (9262.67). On Ant, PPO achieves higher return under tighter clipping $\epsilon = 0.1$ (1899.19, 135.4% retention), indicating that contact-rich dynamics benefit from conservative updates when surrogate curvature is absent; Symphony achieves > 2000 IQM under both $\epsilon = 0.1$ (2005.43) and $\epsilon = 0.2$ (2277.88), exhibiting lower sensitivity than PPO across the tested settings despite experiencing a return decline under relaxed clipping $\epsilon = 0.4$ (1309.47, 57.5% retention).

D.2 Detailed Training Metrics

To provide a deeper understanding of the optimization dynamics and support the theoretical claims of variance reduction, we present comprehensive training metrics for the highest-complexity continuous control tasks. Also, all algorithms used same *fps* for the benchmarking.

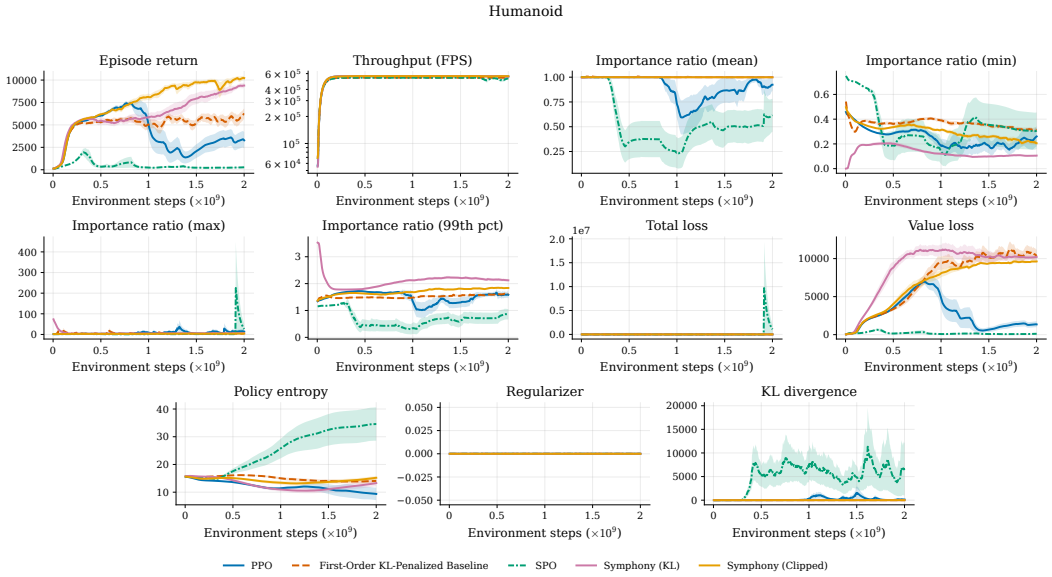


Figure 10: Detailed training diagnostics for the Humanoid environment. The plots track the evolution of the importance ratio (mean, min, max, 99th percentile), KL divergence, policy entropy, and respective optimization losses. Noticeably, Symphony (Clipped) maintains tighter control over the maximum importance ratio and KL divergence spikes compared to unclipped baselines.

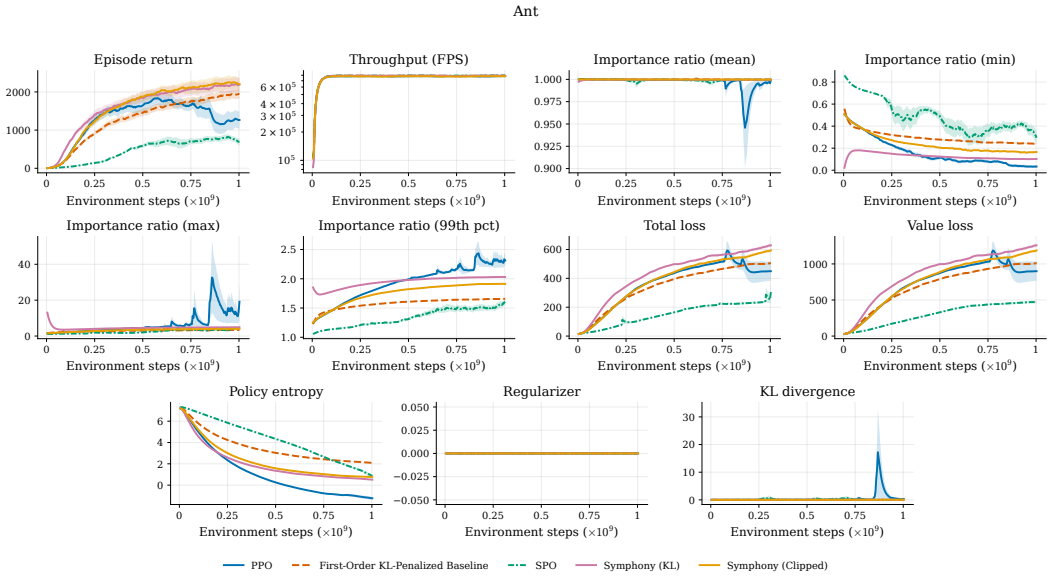


Figure 11: Detailed training diagnostics for the Ant environment. The plots track the evolution of the importance ratio (mean, min, max, 99th percentile), KL divergence, policy entropy, and respective optimization losses. These auxiliary metrics illustrate how Symphony's intrinsic curvature effectively suppresses unstable updates driven by discontinuous contacts, corroborating the sample efficiency gains observed in the primary learning curves.

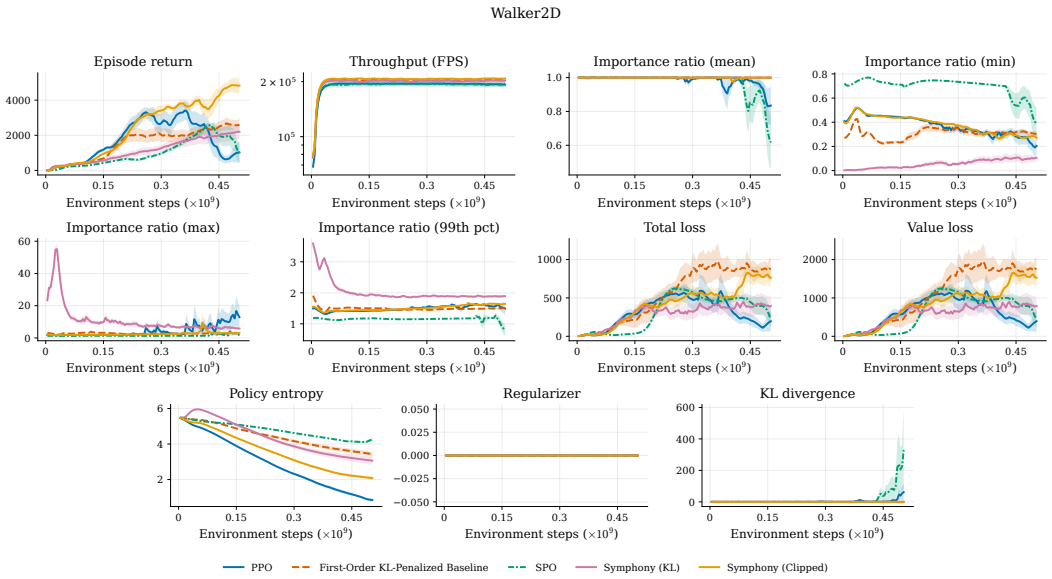


Figure 12: Detailed training diagnostics for the Walker 2D environment. The plots track the evolution of the importance ratio (mean, min, max, 99th percentile), KL divergence, policy entropy, and optimization losses. The bipedal locomotion task introduces contact-driven variance across baselines, but Symphony (Clipped) maintains stronger return performance while suppressing extreme ratio and KL excursions. The gap between clipped and unclipped Symphony highlights the value of the proximal boundary in controlling residual variance under unstable contact dynamics.

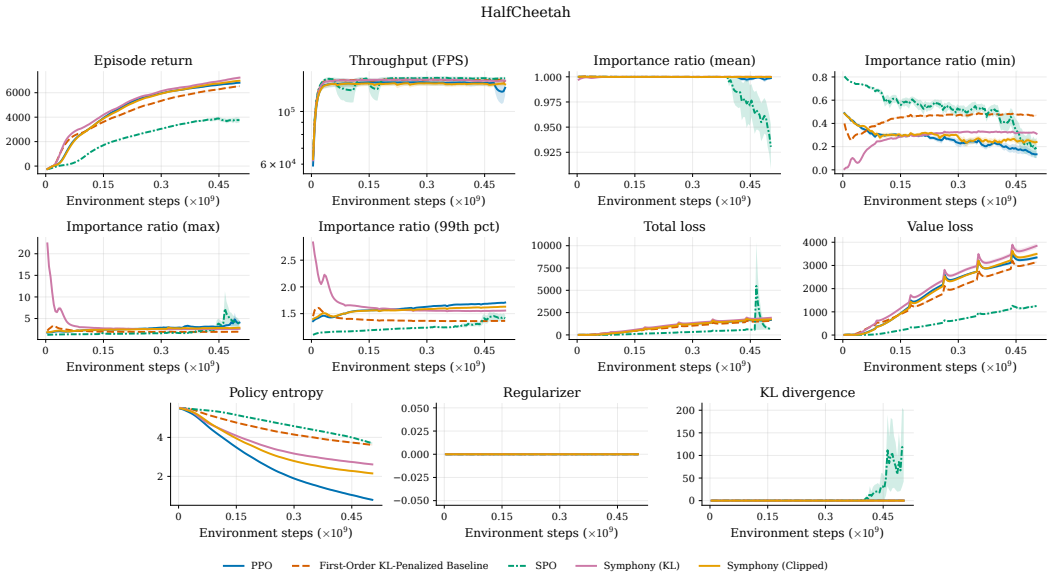


Figure 13: Detailed training diagnostics for the Half Cheetah environment. The plots track the evolution of the importance ratio (mean, min, max, 99th percentile), KL divergence, policy entropy, and optimization losses. HalfCheetah exhibits smoother, contact-light dynamics, keeping policy-gradient variance moderate across methods. In this benign regime, unclipped Symphony slightly benefits from productive ratio excursions, showing that the negatively curved surrogate can improve performance without imposing unnecessary conservatism.

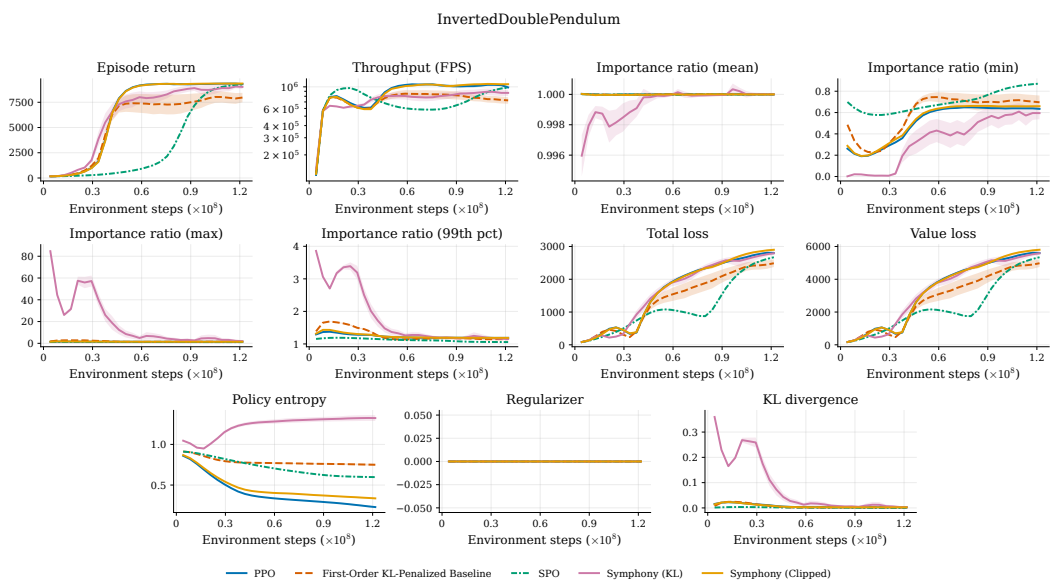


Figure 14: Detailed training diagnostics for the Inverted Double Pendulum environment. The plots track the evolution of the importance ratio (mean, min, max, 99th percentile), KL divergence, policy entropy, and optimization losses. The low-dimensional action space and smooth dynamics yield well-conditioned updates for all methods, with importance ratios concentrated near unity and KL divergence remaining bounded. These diagnostics show that Symphony preserves stable optimization behavior in low-variance regimes without degrading the learning signal.

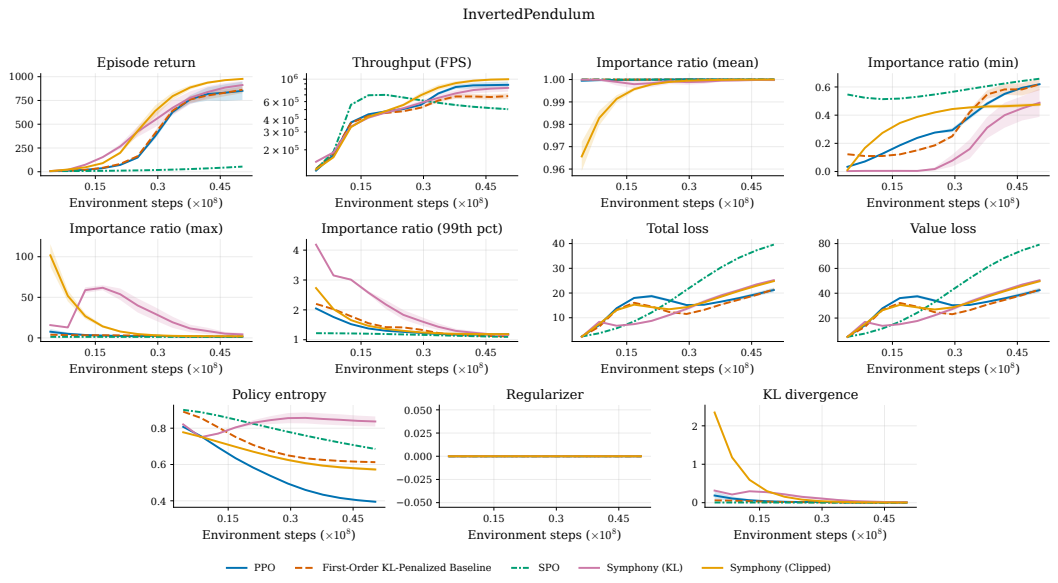


Figure 15: Detailed training diagnostics for the Inverted Pendulum environment. The plots track the evolution of the importance ratio (mean, min, max, 99th percentile), KL divergence, policy entropy, and optimization losses. As the simplest benchmark, InvertedPendulum produces minimal importance-weight stochasticity and rapid convergence toward near-ceiling returns. The close agreement among Symphony, Symphony (Clipped), and the baselines indicates that Symphony remains effectively neutral when surrogate-geometry corrections have little practical impact.

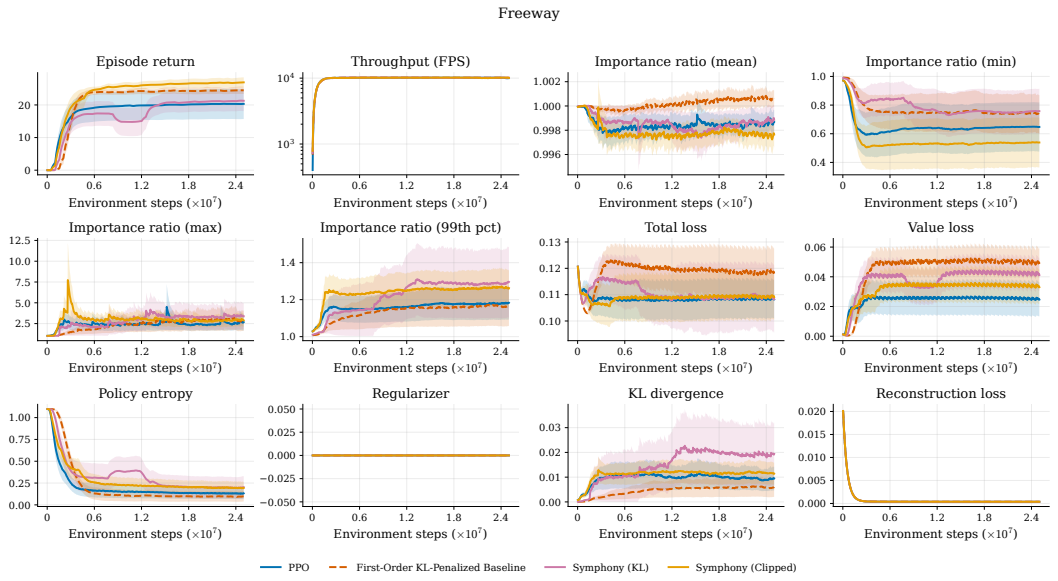


Figure 16: Detailed training diagnostics for the Freeway environment. The plots track the evolution of the importance ratio statistics, KL divergence, policy entropy, reconstruction loss, and optimization losses. Symphony (Clipped) achieves the strongest return trajectory while keeping KL divergence and ratio statistics well controlled. The stability of the clipped variant highlights the benefit of combining the Symphony surrogate with a hard proximal boundary in sparse-reward visual control.

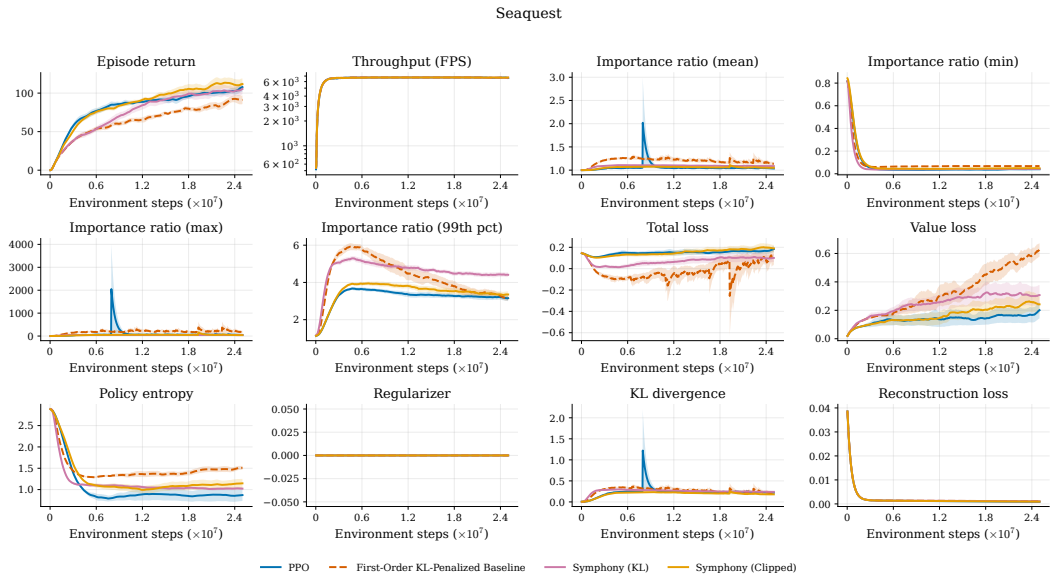


Figure 17: Detailed training diagnostics for the Seaquest environment. The plots track importance ratio behavior, KL divergence, policy entropy, reconstruction loss, and optimization losses throughout training. Symphony (Clipped) obtains the best final return while avoiding the large KL and ratio spikes observed in PPO. These diagnostics show that clipping the curved Symphony surrogate improves stability in visually complex, high-variance Atari dynamics.

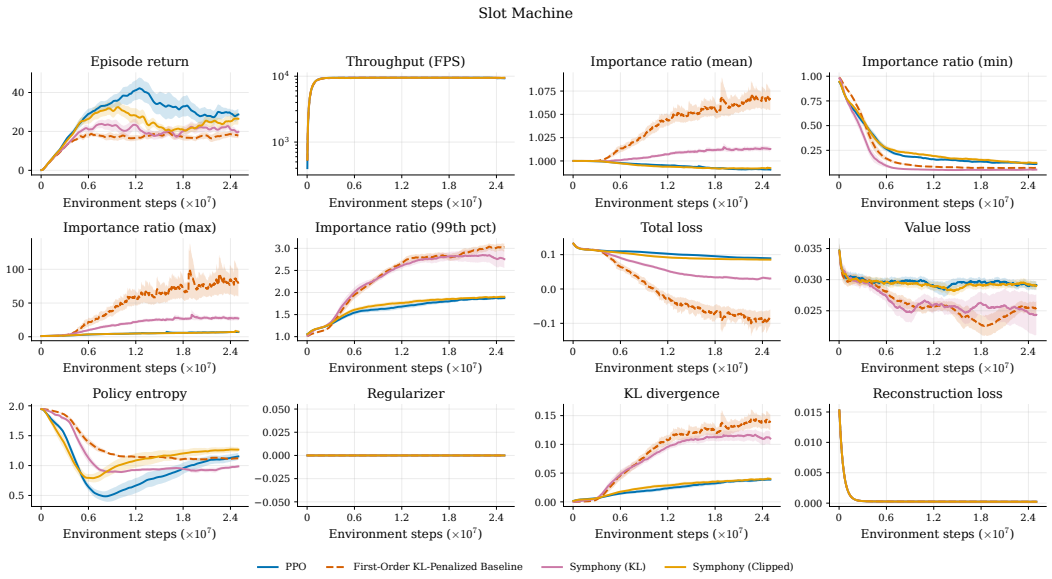


Figure 18: Detailed training diagnostics for the Slot Machine environment. The plots track the importance ratio statistics, KL divergence, policy entropy, reconstruction loss, and optimization losses. PPO reaches a higher peak return, while Symphony (Clipped) maintains lower KL divergence and more conservative importance-ratio growth. This figure illustrates a regime where stronger update control improves stability but does not translate into the best return.

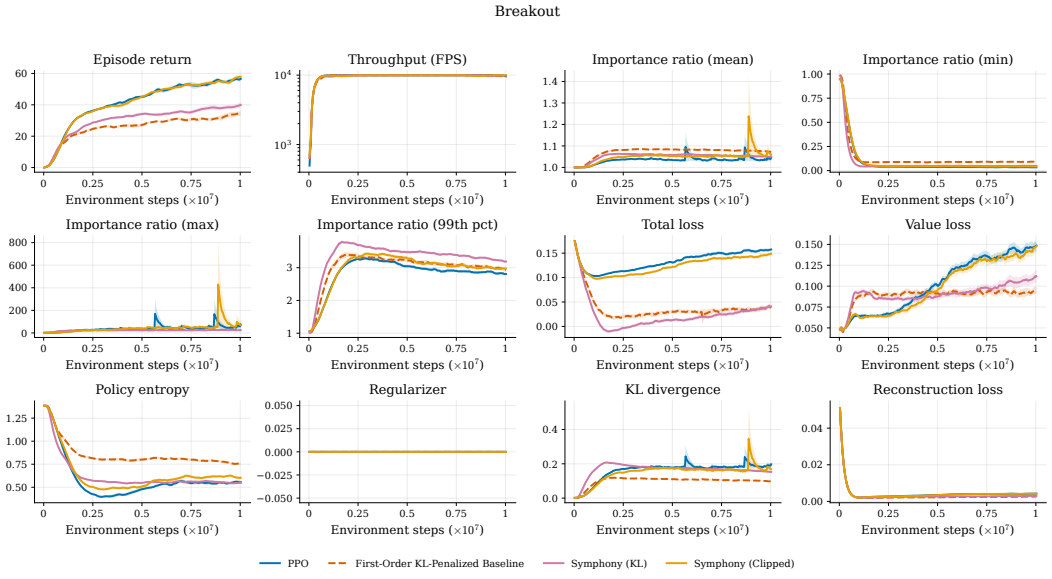


Figure 19: Detailed training diagnostics for the Breakout environment. The plots track importance ratios, KL divergence, policy entropy, reconstruction loss, and optimization losses. PPO and Symphony (Clipped) achieve the strongest returns, with Symphony (Clipped) closely matching or slightly exceeding PPO while keeping ratio statistics broadly controlled. The occasional ratio and KL spikes indicate that visual-control tasks still induce high-variance updates even under clipping.

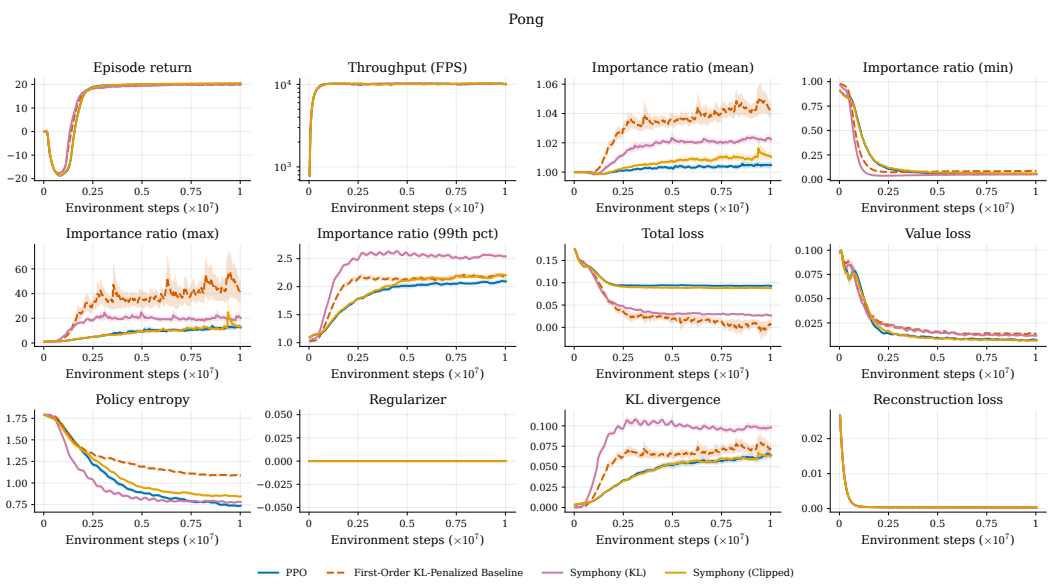


Figure 20: Detailed training diagnostics for the Pong environment. The plots track the evolution of importance ratio statistics, KL divergence, policy entropy, reconstruction loss, and optimization losses. All methods rapidly converge to near-ceiling returns, making performance differences small. Symphony (Clipped) preserves this fast convergence while maintaining lower ratio growth than the first-order KL-penalized baseline and unclipped Symphony.

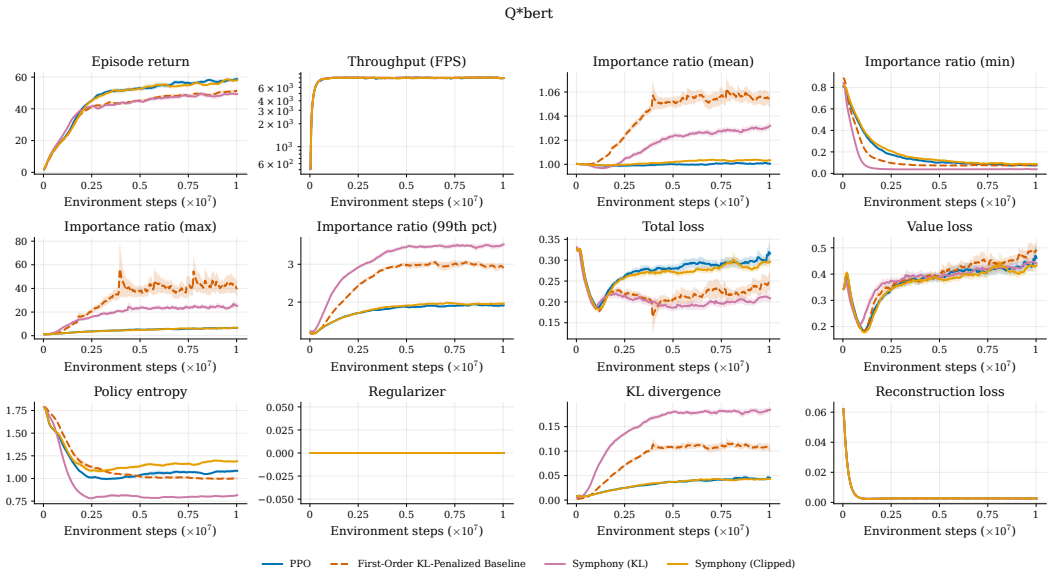


Figure 21: Detailed training diagnostics for the Q*bert environment. The plots track importance ratios, KL divergence, policy entropy, reconstruction loss, and optimization losses. PPO attains a slight return advantage, while Symphony (Clipped) remains competitive and keeps KL divergence substantially lower than the unclipped Symphony baseline. These results suggest that stronger curvature and clipping improve update stability, though the added conservatism can slightly limit performance in this task.

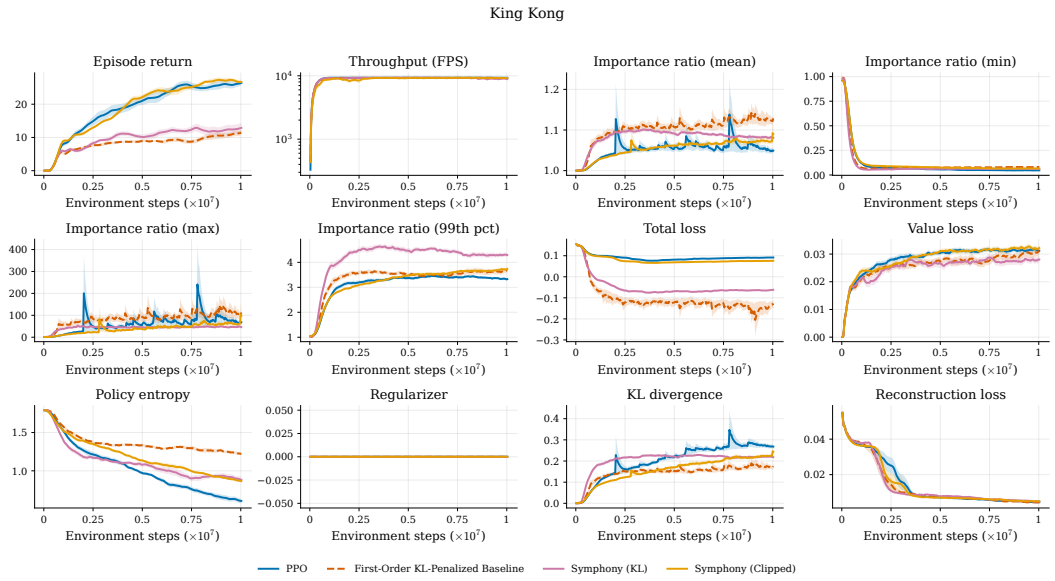


Figure 22: Detailed training diagnostics for the King Kong environment. The plots track importance ratio statistics, KL divergence, policy entropy, reconstruction loss, and optimization losses. PPO and Symphony (Clipped) achieve the strongest return trajectories, with the clipped Symphony variant maintaining competitive performance while avoiding the larger ratio excursions seen in the first-order KL-penalized baseline and PPO. The results support the role of the clipped surrogate in stabilizing visually driven policy updates.

D.3 Statistical Uncertainty Reporting and Bootstrap Confidence Intervals

To provide complete statistical transparency and address empirical uncertainty, Table 5 and Table 6 report the Interquartile Mean (IQM), Mean $\pm$ Standard Error (SE $= s/\sqrt{8}$), 95% Percentile Bootstrap Confidence Intervals for the IQM, and Probability of Improvement (POI vs. PPO) with 95% Percentile Bootstrap Confidence Intervals.

Aggregation and Bootstrap Protocol. Each random seed's evaluation is first summarized as a single scalar observation by computing the temporal mean return across its final 10% of training steps ($\bar{R}_i = \frac{1}{|T_{10\%}|} \sum_{t \in T_{10\%}} R_{i,t}$), yielding $N = 8$ independent observations per algorithm-environment pair. Confidence intervals for the IQM are constructed using standard percentile bootstrap by resampling the $N = 8$ seed-level observations with replacement across $B = 2{,}000$ bootstrap replicates and extracting the 2.5th and 97.5th percentiles.

Furthermore, while POI point estimates are evaluated across all $8 \times 8 = 64$ pairwise seed comparisons ($\frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\mathbb{I}[X_i > Y_j] + 0.5 \cdot \mathbb{I}[X_i = Y_j])$), these pairs share underlying seeds and are not 64 independent observations. To properly account for this shared-seed dependency, we construct 95% cluster-bootstrap confidence intervals for POI by resampling the seed indices of both the target and baseline algorithms with replacement ($B = 2{,}000$ resamples) and evaluating the pairwise comparison matrix on each resampled set.

Note on Degenerate POI Intervals. When every observed seed of one algorithm strictly outperforms all observed seeds of the baseline across the finite sample (as occurs on Humanoid for Symphony vs. PPO, or SPO on Inverted Pendulum), every bootstrap resample likewise exhibits complete pairwise dominance. This produces degenerate empirical intervals such as 100.0% $[100.0\%, 100.0\%]$ or 0.0% $[0.0\%, 0.0\%]$. These collapsed intervals reflect a known limitation of the empirical bootstrap under complete sample separation rather than an assertion of mathematical certainty over unobserved future runs.

Analysis of Uncertainty and Comparisons. In MJX continuous control (Table 5), high-dimensional locomotion environments demonstrate decisive, non-overlapping IQM performance separations between Symphony (Clipped) and PPO (e.g., on Humanoid: IQM 10137.06 with 95% CI $[9724.31, 10481.90]$ vs. PPO 3749.73 with $[1085.10, 5223.04]$; on Walker 2D: $[3663.17, 5667.94]$ vs. $[281.63, 2015.68]$). On Ant, Symphony achieves a substantially higher point estimate (2277.88 vs. 1402.43) and 90.6% POI [71.9%, 100.0%], though high seed dispersion results in narrow CI overlap ($[1738.10, 2659.55]$ vs. $[622.82, 1789.78]$). Near-ceiling tasks (Inverted Pendulum, Inverted Double Pendulum, Half Cheetah) perform comparably across methods, with Symphony (KL) slightly leading Half Cheetah (7085.84 IQM).

Conversely, in JAXAtari visual control (Table 6), performance differences between Symphony (Clipped) and PPO are modest and exhibit substantially overlapping confidence intervals across all games (e.g., Breakout: $[54.09, 58.37]$ vs. $[53.72, 57.89]$; Pong: $[20.19, 20.57]$ vs. $[20.10, 20.54]$), with POI hovering near 50% (e.g., 57.8% [25.0%, 85.9%] on Breakout; 54.7% [23.4%, 81.2%] on Pong; 48.4% [18.8%, 79.7%] on Q*bert). However, on games like Breakout, both clipped methods separate decisively from the unclipped KL-penalized baselines ($[30.85, 37.67]$ and $[35.88, 43.39]$). These intervals confirm that while Symphony matches or slightly edges PPO in five of seven games, it does not dominate visual control, reflecting the environment-dependent utility of ratio curvature.

Table 5: **MJX Continuous Control Statistical Uncertainty (8 Seeds, Final 10% of Training).** Degenerate POI intervals (e.g., 100.0% $[100.0\%, 100.0\%]$ or 0.0% $[0.0\%, 0.0\%]$) arise under complete sample separation across the 8 observed seeds, reflecting empirical bootstrap limits rather than absolute certainty over unobserved runs.

Algorithm	IQM	Mean	Std Error	95% Bootstrap CI (IQM)	POI vs. PPO [95% CI]
Ant					
PPO	1402.43	1250.02	± 222.32	[622.82, 1789.78]	—
First-Order KL Baseline	1969.31	1932.47	± 113.33	[1624.25, 2232.46]	87.5% [67.2%, 100.0%]
SPO	788.37	759.21	± 62.79	[575.70, 917.10]	25.0% [0.0%, 50.0%]
Symphony (KL)	2206.67	2178.58	± 180.98	[1665.00, 2667.38]	87.5% [67.2%, 100.0%]
Symphony (Clipped)	**2277.88**	**2230.91**	± 163.83	[1738.10, 2659.55]	**90.6% [71.9%, 100.0%]**
Half Cheetah					
PPO	6823.49	6767.62	± 72.09	[6562.69, 6934.65]	—
First-Order KL Baseline	6445.09	6465.12	± 31.02	[6392.31, 6556.07]	12.5% [0.0%, 37.5%]
SPO	3757.87	3736.00	± 132.70	[3350.90, 4086.78]	0.0% [0.0%, 0.0%]
Symphony (KL)	**7085.84**	**7146.65**	± 97.36	[6905.91, 7428.82]	**85.9% [62.5%, 100.0%]**
Symphony (Clipped)	6905.14	6915.76	± 64.16	[6729.46, 7105.54]	67.2% [34.4%, 92.2%]
Humanoid					
PPO	3749.73	3311.63	± 755.04	[1085.10, 5223.04]	—
First-Order KL Baseline	5870.06	5883.00	± 382.19	[4863.79, 6912.89]	85.9% [64.1%, 100.0%]
SPO	194.83	250.73	± 72.92	[93.47, 463.88]	6.2% [0.0%, 20.4%]
Symphony (KL)	9464.72	9232.07	± 256.74	[8421.51, 9685.61]	100.0% [100.0%, 100.0%]
Symphony (Clipped)	**10137.06**	**10124.03**	± 137.55	[9724.31, 10481.90]	**100.0% [100.0%, 100.0%]**
Inverted Double Pendulum					
PPO	9327.01	9314.82	± 18.29	[9265.13, 9356.18]	—
First-Order KL Baseline	7968.49	7910.99	± 474.38	[6529.96, 9312.03]	15.6% [0.0%, 39.1%]
SPO	9282.41	9283.23	± 14.41	[9239.52, 9317.91]	32.8% [7.8%, 65.6%]
Symphony (KL)	9234.15	9031.09	± 168.65	[8541.80, 9336.18]	23.4% [4.7%, 48.4%]
Symphony (Clipped)	**9356.19**	**9356.01**	± 0.25	[9355.26, 9356.56]	**71.9% [37.5%, 100.0%]**
Inverted Pendulum					
PPO	986.34	858.34	± 105.12	[532.56, 994.38]	—
First-Order KL Baseline	876.30	876.55	± 31.74	[787.43, 961.61]	34.4% [7.8%, 64.1%]
SPO	58.46	58.30	± 0.72	[56.30, 60.20]	0.0% [0.0%, 0.0%]
Symphony (KL)	971.90	935.80	± 34.92	[826.93, 989.41]	39.1% [12.5%, 68.8%]
Symphony (Clipped)	**990.96**	**989.89**	± 3.22	[980.00, 997.81]	**65.6% [35.9%, 92.2%]**
Walker 2D					
PPO	439.78	825.27	± 371.60	[281.63, 2015.68]	—
First-Order KL Baseline	2642.99	2609.66	± 308.05	[1804.05, 3376.64]	90.6% [67.2%, 100.0%]
SPO	1602.39	1571.49	± 501.52	[-5.60, 2956.54]	56.2% [23.4%, 87.5%]
Symphony (KL)	1647.37	2147.26	± 448.01	[1281.54, 3541.60]	89.1% [62.5%, 100.0%]
Symphony (Clipped)	**5007.01**	**4819.59**	± 358.00	[3663.17, 5667.94]	**98.4% [90.6%, 100.0%]**

Table 6: **JAXAtari Benchmark Statistical Uncertainty (8 Seeds, Final 10% of Training).** Degenerate POI intervals (e.g., 0.0% $[0.0\%, 0.0\%]$) arise under complete sample separation across the 8 observed seeds, reflecting empirical bootstrap limits rather than absolute certainty over unobserved runs.

Algorithm	IQM	Mean	Std Error	95% Bootstrap CI (IQM)	POI vs. PPO [95% CI]
Breakout					
PPO	55.61	55.76	± 0.75	[53.72, 57.89]	—
First-Order KL Baseline	33.61	33.98	± 1.25	[30.85, 37.67]	0.0% [0.0%, 0.0%]
Symphony (KL)	38.50	39.05	± 1.37	[35.88, 43.39]	0.0% [0.0%, 0.0%]
Symphony (Clipped)	**56.34**	**56.29**	± 0.79	[54.09, 58.37]	**57.8% [25.0%, 85.9%]**
Freeway					
PPO	24.95	20.30	± 4.64	[7.61, 31.27]	—
First-Order KL Baseline	22.81	24.45	± 1.10	[22.73, 27.45]	37.5% [9.4%, 65.6%]
Symphony (KL)	22.88	21.24	± 3.14	[11.43, 26.50]	52.3% [21.9%, 84.4%]
Symphony (Clipped)	**26.80**	**26.85**	± 1.51	[22.85, 30.87]	**57.0% [23.4%, 87.5%]**
King Kong					
PPO	26.01	26.12	± 0.88	[23.68, 28.72]	—
First-Order KL Baseline	11.82	11.09	± 0.70	[8.96, 12.51]	0.0% [0.0%, 0.0%]
Symphony (KL)	11.68	12.59	± 1.43	[9.16, 16.50]	0.0% [0.0%, 0.0%]
Symphony (Clipped)	**27.25**	**26.96**	± 0.54	[25.49, 28.37]	**57.8% [29.7%, 87.5%]**
Pong					
PPO	20.36	20.33	± 0.08	[20.10, 20.54]	—
First-Order KL Baseline	19.84	19.86	± 0.08	[19.61, 20.09]	7.8% [0.0%, 25.0%]
Symphony (KL)	19.94	19.90	± 0.12	[19.54, 20.21]	15.6% [0.0%, 40.6%]
Symphony (Clipped)	**20.39**	**20.39**	± 0.07	[20.19, 20.57]	**54.7% [23.4%, 81.2%]**
Q*bert					
PPO	**58.38**	**58.16**	± 0.95	[55.30, 60.66]	—
First-Order KL Baseline	50.74	51.02	± 0.67	[49.30, 52.99]	0.0% [0.0%, 0.0%]
Symphony (KL)	49.15	49.31	± 1.01	[46.62, 52.20]	0.0% [0.0%, 0.0%]
Symphony (Clipped)	58.05	58.10	± 0.72	[56.18, 60.15]	48.4% [18.8%, 79.7%]
Seaquest					
PPO	101.75	104.53	± 4.65	[94.52, 119.29]	—
First-Order KL Baseline	91.92	91.74	± 4.76	[79.56, 106.20]	21.9% [0.0%, 48.4%]
Symphony (KL)	100.57	103.41	± 6.32	[87.02, 122.83]	48.4% [18.8%, 79.7%]
Symphony (Clipped)	**107.72**	**111.66**	± 7.06	[95.07, 134.54]	**59.4% [25.0%, 85.9%]**
Slot Machine					
PPO	**27.91**	**28.96**	± 2.14	[24.04, 35.64]	—
First-Order KL Baseline	17.81	18.18	± 1.22	[14.99, 21.63]	3.1% [0.0%, 14.1%]
Symphony (KL)	20.25	21.11	± 1.03	[19.03, 24.41]	6.2% [0.0%, 21.9%]
Symphony (Clipped)	23.28	25.62	± 2.51	[20.13, 32.77]	31.2% [4.7%, 60.9%]

Table 7: **Environment Complexity and Training Budgets across Benchmarks.** Observation dimensions, action dimensions, and total training transitions for all evaluated environments.

MJX Continuous Control				JAXAtari Benchmarks	
Environment	Obs Dim	Act Dim	Total Steps	Environment	Total Steps
Humanoid	227	17	2.00B	Freeway	25.002M
Ant	87	8	1.00B	Seaquest	25.002M
Walker 2D	17	6	503.31M	Slot Machine	25.002M
Half Cheetah	18	6	503.31M	Breakout	10.027M
Inv. Double Pendulum	9	1	121.63M	Pong	10.027M
Inverted Pendulum	4	1	50.33M	Q*bert	10.027M
				King Kong	10.027M

E CONTROLLED BASELINES

E.1 PURPOSE AND RELATIONSHIP TO LONG-HORIZON EVALUATIONS

Our primary benchmark evaluations across continuous control (MJX) and visual control (JAXAtari) stress-test extended training over up to 2 billion transitions within a shared architectural and hyperparameter configuration. While these long-horizon experiments evaluate asymptotic stability and drift resilience under fixed training conditions, they leave open a distinct and critical question: *do Symphony's empirical advantages persist when competitor baselines (PPO and SPO) receive their own dedicated hyperparameter search?*

To address this question without confounding algorithmic mechanisms with divergent implementation choices, we conduct a controlled hyperparameter sensitivity and confirmation study on the Humanoid benchmark ($9,961,472$ transitions, ≈ 10M steps). Configurations are selected independently for each baseline from a systematic screening sweep, and Symphony (Clipped) is subsequently evaluated strictly under those baseline-selected settings, inheriting the exact hyperparameters chosen for its competitor.

We explicitly distinguish the scope of these two empirical studies:

- **Extended-Horizon Asymptotic Evaluations (up to 2B transitions):** Evaluate long-horizon stability, asymptotic returns, and compounding ratio drift under a standardized, shared configuration across continuous control (MJX) and visual control (JAXAtari) environments.
- **Controlled Baseline-Selected Study on Humanoid (10M transitions):** Tests whether Symphony (Clipped)'s performance advantage persists when competitor baselines receive dedicated hyperparameter tuning within our common implementation. This study addresses tuning specifically on Humanoid; our experiments establish improvements under a shared training implementation, including baseline-specific tuning on Humanoid, but do not establish uniform superiority across independently optimized training pipelines.

Furthermore, we clarify that SPO is evaluated by implementing the SPO surrogate objective (Xie et al., 2025) directly inside our shared Flax/JAX training pipeline (sharing the exact same MLP architecture, AdamW optimizer, running observation normalization, and per-batch advantage standardization; see Section E.6 and Table 12 for an itemized comparison with the published SPO pipeline). We do not describe this as reproducing SPO's published training pipeline or external codebase from Xie et al. (2025); our empirical comparison isolates the mathematical surrogate objective under identical, controlled implementation conditions, addressing the narrower, supported claim that Symphony (Clipped) outperforms the independently tuned SPO objective within this common implementation.

E.2 CONTROLLED FACTORIAL SWEEP DESIGN

To explore baseline sensitivity in a rigorous, controlled manner, we systematically varied four fundamental optimization hyperparameters while holding all remaining architectural, reward-processing, and optimization choices strictly fixed within our shared normalized training stack:

1. **Initial Learning Rate ($\alpha \in \{1 \times 10^{-4}, 3 \times 10^{-4}, 8 \times 10^{-4}\}$):** Controls the magnitude of parameter updates along the policy gradient direction.
2. **Rollout Length per Environment ($L \in \{32, 256\}$ transitions):** Determines trajectory chunking, update frequency, and surrogate batch size across 2048 parallel environments (65,536 versus 524,288 transitions per rollout buffer).
3. **Clipping / Penalty Boundary ($\epsilon \in \{0.1, 0.2, 0.4\}$):** Sets the proximal trust-region boundary for PPO and the curvature regularizer penalty boundary for SPO.
4. **Number of Update Epochs ($E \in \{2, 8\}$):** Dictates data reuse and the degree of optimization per collected rollout buffer.

This sweep forms a 36-configuration factorial grid per baseline ($3 \times 2 \times 3 \times 2$), supplemented by a reference configuration ($\text{lr} = 8 \times 10^{-4}, L = 32, \epsilon = 0.2, E = 4$). Selection was performed strictly over the 36 factorial configurations; the reference configuration served purely as an anchor against default settings and did not participate in baseline configuration selection. All algorithms share an

identical training infrastructure: 2-layer MLP actor and critic (2×256 units), AdamW optimizer, minibatch size 2048, gradient clipping at 1.0, discount factor $\gamma = 0.99$, GAE parameter $\lambda = 0.95$, value function coefficient 0.5, policy entropy coefficient 0.0, running observation normalization, and batch advantage normalization (raw environment rewards are left unnormalized).

E.3 Two-Stage Budget Allocation: Screening vs. Confirmation

The study employs two distinct transition budgets:

- **Screening Stage (≈ 5M transitions):** Evaluating 36 configurations per baseline over extended horizons is computationally intensive. Screening configurations over a nominal budget of approximately 5,000,000 transitions across two screening seeds (seeds 11, 22) per configuration makes a broad search computationally affordable while providing a reliable metric for early learning speed. Configurations are ranked by the *mean normalized area under the return curve (AUC)* across screening seeds, defined as the trapezoidal time-average of training episode return over the transition budget: AUC $= \frac{1}{T}\int_0^T R(t)dt$, where $R(t)$ denotes the mean return of recently completed training episodes logged at each iteration via `return_mean`. This division by the transition horizon T normalizes the temporal scale to return units (yielding an interpretable score on the original return scale) without modifying raw environment rewards. The screening stage is strictly a configuration selection procedure; it is not intended to establish long-horizon asymptotic stability or replace the 2-billion-step evaluations.

- **Confirmation Stage (≈ 10M transitions; $9,961,472$ steps):** Each baseline's top-performing configuration identified during screening is subsequently evaluated over a longer confirmation budget across 4 fresh, independent random seeds: seeds 101, 202, 303, 404 for the PPO comparison, and seeds 1101, 1202, 1303, 1404 for the SPO comparison. Final return is defined as the mean return of recently completed training episodes at the completion of the confirmation horizon ($9,961,472$ transitions) across seeds, recorded via `return_mean`. Symphony (Clipped) is evaluated at the exact same confirmation budget under each baseline's selected configuration, inheriting its competitor's settings.

Table 8: **Baseline-Selected Configurations under the Screening Criterion.** Top-performing configurations identified within the 36-configuration screening search space ranked by mean screening AUC. These represent the best configurations within the tested grid rather than globally optimal hyperparameters. In confirmation, Symphony (Clipped) inherits the settings selected for its competitor in each pairwise comparison.

Target Baseline	Learning Rate (α)	Rollout Length (L)	Clipping Boundary (ϵ)	Update Epochs (E)
PPO-Selected Setting	3×10^{-4}	32	0.4	8
SPO-Selected Setting	1×10^{-4}	32	0.4	8

Table 9: **Complete 36-Configuration Factorial Screening Grid on Humanoid with Baseline Screening AUC Scores.** The factorial sweep varies initial learning rate α, rollout length L, clipping/penalty boundary ϵ, and update epochs E. All 36 configurations were evaluated over a nominal 5M transition budget across two independent screening seeds (seeds 11, 22) and ranked by mean normalized screening AUC. PPO and SPO configurations were independently selected under this criterion: Configuration 18 achieved the highest screening AUC for PPO (3312.8), and Configuration 6 achieved the highest screening AUC for SPO (2123.8). The full CSV table, seed-level scores, and run manifests are documented in `codes/control_evaluation/screening_grid_configurations.csv`.

Config	α	L	ϵ	E	PPO	SPO	Config	α	L	ϵ	E	PPO	SPO
1	1×10^{-4}	32	0.1	2	991.0	776.1	19	3×10^{-4}	256	0.1	2	529.5	273.3
2	1×10^{-4}	32	0.1	8	1330.7	1059.2	20	3×10^{-4}	256	0.1	8	739.2	340.0
3	1×10^{-4}	32	0.2	2	1577.1	1223.0	21	3×10^{-4}	256	0.2	2	878.1	391.7
4	1×10^{-4}	32	0.2	8	2175.0	1676.8	22	3×10^{-4}	256	0.2	8	1178.0	586.3
5	1×10^{-4}	32	0.4	2	1949.0	1519.4	23	3×10^{-4}	256	0.4	2	1087.7	510.2
6 (SPO)	1×10^{-4}	32	0.4	8	2706.6	**2123.8**	24	3×10^{-4}	256	0.4	8	1503.6	726.0
7	1×10^{-4}	256	0.1	2	449.7	334.0	25	8×10^{-4}	32	0.1	2	1047.5	390.2
8	1×10^{-4}	256	0.1	8	596.8	470.1	26	8×10^{-4}	32	0.1	8	1444.3	577.9
9	1×10^{-4}	256	0.2	2	704.7	577.6	27	8×10^{-4}	32	0.2	2	1685.5	678.6
10	1×10^{-4}	256	0.2	8	954.0	768.9	28	8×10^{-4}	32	0.2	8	2304.4	915.2
11	1×10^{-4}	256	0.4	2	864.8	686.1	29	8×10^{-4}	32	0.4	2	2128.5	858.7
12	1×10^{-4}	256	0.4	8	1219.3	949.8	30	8×10^{-4}	32	0.4	8	2914.9	1183.2
13	3×10^{-4}	32	0.1	2	1181.9	599.5	31	8×10^{-4}	256	0.1	2	500.4	169.2
14	3×10^{-4}	32	0.1	8	1689.8	814.6	32	8×10^{-4}	256	0.1	8	642.1	286.6
15	3×10^{-4}	32	0.2	2	1879.5	895.5	33	8×10^{-4}	256	0.2	2	736.4	290.1
16	3×10^{-4}	32	0.2	8	2636.6	1280.8	34	8×10^{-4}	256	0.2	8	1027.8	429.8
17	3×10^{-4}	32	0.4	2	2378.2	1144.9	35	8×10^{-4}	256	0.4	2	953.4	390.2
18 (PPO)	3×10^{-4}	32	0.4	8	**3312.8**	1591.8	36	8×10^{-4}	256	0.4	8	1320.4	518.6

E.4 Confirmation Results and Statistical Comparison

Confirmation runs were executed across 4 fresh random seeds per baseline comparison: seeds 101, 202, 303, 404 for PPO vs. Symphony (Clipped), and seeds 1101, 1202, 1303, 1404 for SPO vs. Symphony (Clipped). Crucially, within each pairwise comparison, Symphony (Clipped) and the target baseline share the exact same evaluation seeds and rollout conditions, enabling direct seed-paired trajectory comparisons. Performance metrics report mean $\pm$ sample standard deviation along with 95% bootstrap confidence intervals computed over 10,000 resamples across seeds. All difference intervals (Symphony (Clipped) $-$ Baseline) are computed via paired bootstrap resampling over these shared seeds (Table 10 and Figure 4).

Table 10: **Humanoid 10M Confirmation Evaluation Results.** Evaluated over $9,961,472$ transitions. Part I reports the PPO comparison under the PPO-selected configuration (lr $= 3 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$) across 4 fresh seeds (seeds 101, 202, 303, 404). Part II reports the SPO comparison under the SPO-selected configuration (lr $= 1 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$) across 4 fresh seeds (seeds 1101, 1202, 1303, 1404). Symphony (Clipped) inherits the competitor's configuration in each comparison. Reported metrics include mean $\pm$ sample SD and 95% bootstrap confidence intervals for completed training episode returns and normalized learning AUC.

Method	Final Return (Mean $\pm$ SD)	Final Return [95% CI]	Normalized AUC (Mean $\pm$ SD)	Normalized AUC [95% CI]
Part I: PPO-Selected Setting (lr $= 3 \times 10^{-4}$, Rollout 32, $\epsilon = 0.4$, 8 Epochs; 4 Fresh Seeds: 101, 202, 303, 404)				
PPO	4290.21 ± 2745.66	$[1663.46, 5982.36]$	3207.76 ± 543.57	$[2665.13, 3623.26]$
Symphony (Clipped)	$\mathbf{5852.32 \pm 1181.60}$	$[4744.48, 6902.97]$	$\mathbf{3602.23 \pm 17.95}$	$[3589.65, 3618.73]$
Difference (Symphony (Clipped) $-$ PPO)	$+\mathbf{1562.11}$ **[95% CI:** $+290.86, +3318.82$**]**		$+394.46$ [95% CI: $-35.64, +955.80$]	
Part II: SPO-Selected Setting (lr $= 1 \times 10^{-4}$, Rollout 32, $\epsilon = 0.4$, 8 Epochs; 4 Fresh Seeds: 1101, 1202, 1303, 1404)				
SPO	4232.56 ± 2295.89	$[1927.06, 5464.13]$	2110.19 ± 106.30	$[2019.15, 2201.23]$
Symphony (Clipped)	$\mathbf{6512.53 \pm 678.36}$	$[5868.18, 7003.40]$	$\mathbf{3579.81 \pm 54.44}$	$[3547.96, 3632.93]$
Difference (Symphony (Clipped) $-$ SPO)	$+\mathbf{2279.97}$ **[95% CI:** $+574.08, +5027.93$**]**		$+\mathbf{1469.62}$ **[95% CI:** $+1354.68, +1584.55$**]**	

Table 11: **Seed-Level Humanoid 10M Confirmation Performance.** Exact completed training episode returns and normalized evaluation AUC over $9,961,472$ transitions. Part I reports the PPO comparison under the PPO-selected configuration (lr $= 3 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$; seeds 101, 202, 303, 404). Part II reports the 4 confirmation seeds for the SPO comparison under the SPO-selected configuration (lr $= 1 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$; seeds 1101, 1202, 1303, 1404). Symphony (Clipped) inherits the target baseline's selected settings in each pairwise comparison. Individual seed logs, manifests, and command lines are fully recorded in `codes/control_evaluation/`.

Setting / Comparison	Evaluation Seed	Baseline Final Return	Symphony (Clipped) Return	Baseline AUC	Symphony (Clipped) AUC
Part I: PPO-Selected Setting ($\alpha = 3 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$)					
PPO vs. Symphony (Clipped)	Seed 101	235.87	4350.34	2439.66	3627.06
PPO vs. Symphony (Clipped)	Seed 202	4960.23	5892.09	3341.52	3602.55
PPO vs. Symphony (Clipped)	Seed 303	6018.51	7239.93	3328.28	3593.72
PPO vs. Symphony (Clipped)	Seed 404	5946.22	5926.89	3721.59	3585.58
PPO Aggregate (Mean ± SD)	**4 Seeds**	$\mathbf{4290.21 \pm 2745.66}$	$\mathbf{5852.32 \pm 1181.60}$	$\mathbf{3207.76 \pm 543.57}$	$\mathbf{3602.23 \pm 17.95}$
Part II: SPO-Selected Setting ($\alpha = 1 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$)					
SPO vs. Symphony (Clipped)	Seed 1101	5281.71	5564.94	2183.31	3549.32
SPO vs. Symphony (Clipped)	Seed 1202	5524.94	6554.06	2011.94	3546.61
SPO vs. Symphony (Clipped)	Seed 1303	5331.27	6777.92	2219.15	3562.51
SPO vs. Symphony (Clipped)	Seed 1404	792.32	7153.19	2026.36	3660.79
SPO Aggregate (Mean ± SD)	**4 Seeds**	$\mathbf{4232.56 \pm 2295.89}$	$\mathbf{6512.53 \pm 678.36}$	$\mathbf{2110.19 \pm 106.30}$	$\mathbf{3579.81 \pm 54.44}$

E.5 Detailed Training Diagnostics for Controlled Baselines

To provide an exhaustive diagnostic view analogous to Figure 10 in Appendix D.2, we plot the full suite of optimization metrics across the 10-million-step confirmation budget for both controlled settings. Each diagnostic figure tracks 11 key quantities: episode return, throughput (FPS), importance ratio statistics (mean, min, max, 99th percentile), total loss, value loss, policy entropy, regularizer / penalty term, and approximate KL divergence.

PPO-Selected Setting Diagnostics (Figure 23). Under PPO's top-performing configuration ($\alpha = 3 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$), PPO initially matches Symphony (Clipped)'s learning trajectory through approximately 4 million transitions. However, as shown in Figure 23, PPO subsequently encounters substantial optimization turbulence:

- **Ratio Excursions and KL Spikes:** PPO experiences sharp spikes in maximum importance ratios ($\max r > 100$) and dramatic bursts in approximate KL divergence (exceeding 10^1 and 10^2). These destabilizing excursions lead to policy degradation in multiple seeds.

- **Policy Entropy Collapse:** PPO's policy entropy declines sharply from ≈ 15.6 down to ≈ 6.0 by step 10M, indicating premature policy determinism and restricted exploration. In contrast, Symphony (Clipped) maintains higher policy entropy (≈ 8.1) alongside smoothly bounded KL divergence ($\approx 3 \times 10^{-2}$), yielding smaller ratio excursions and more stable policy updates.

- **Statistical Separation:** Symphony (Clipped) achieves a higher observed mean AUC (3602.23 vs. 3207.76), but the AUC difference 95% bootstrap confidence interval ($[-35.64, 955.80]$) includes zero, indicating uncertain early sample-efficiency separation. Conversely, the final-return difference ($+1562.11$, 95% CI: $[290.86, 3318.82]$) is strictly positive, corroborating Symphony (Clipped)'s superior late-stage stability under PPO's own preferred hyperparameter setting.

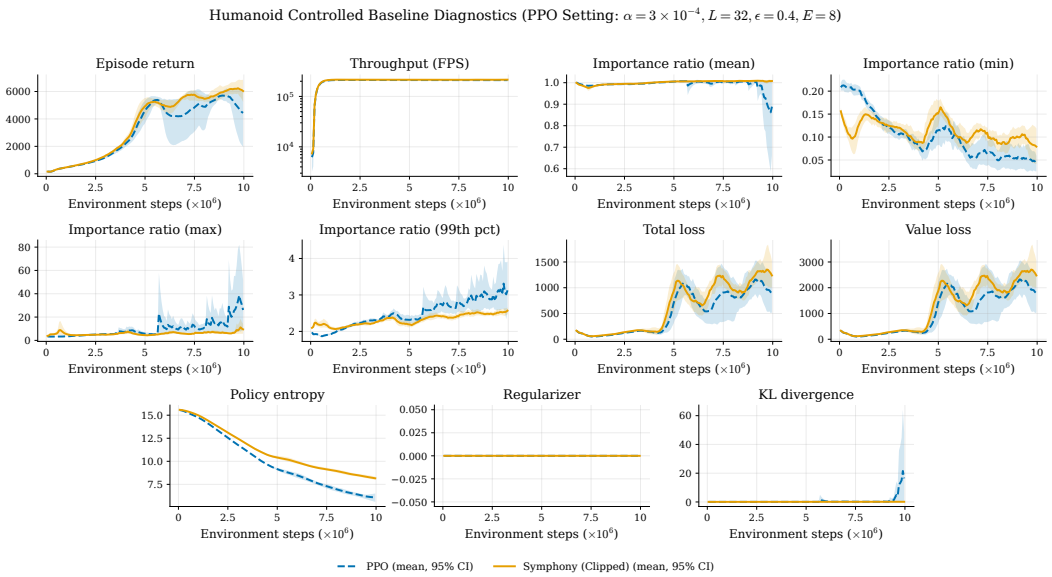


Figure 23: **Detailed Training Diagnostics for Humanoid under the PPO-Selected Setting** ($\alpha = 3 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$)**.** Plots track 11 optimization metrics across 4 fresh random seeds over 10M transitions: episode return, throughput (FPS), importance ratios (mean, min, max, 99th pct), total loss, value loss, policy entropy, regularizer, and approximate KL divergence. Curves report mean values with 95% bootstrap confidence interval shading.

SPO-Selected Setting Diagnostics (Figure 24). Under SPO's top-performing configuration ($\alpha = 1 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$), Symphony (Clipped) demonstrates higher performance and decisive sample efficiency separation across the 4 confirmation seeds (Figure 24):

- **Accelerated Return Convergence:** Symphony (Clipped) reaches an evaluation return of ≈ 4500 by 5 million transitions, whereas SPO requires nearly 8 million transitions to reach 5000 before experiencing late-training degradation in seed 1404. Across the 4 confirmation seeds, Symphony (Clipped) finishes at a mean return of 6512.53 ± 678.36 versus 4232.56 ± 2295.89 for SPO ($+2279.97$ lead, 95% CI: $[+574.08, +5027.93]$).
- **Decisive Cumulative AUC Lead:** Symphony (Clipped) maintains a strictly positive normalized AUC advantage of $+1469.62$ (95% CI: $[+1354.68, +1584.55]$), confirming superior sample efficiency from early training onwards.
- **Well-Conditioned Surrogate Geometry:** While SPO exhibits substantial variance in value loss and importance ratio growth ($\max r \approx 15$), Symphony (Clipped)'s intrinsic negative curvature smoothly conditions updates, maintaining stable importance ratios ($\max r \approx 3$), well-behaved policy entropy, and bounded KL divergence.

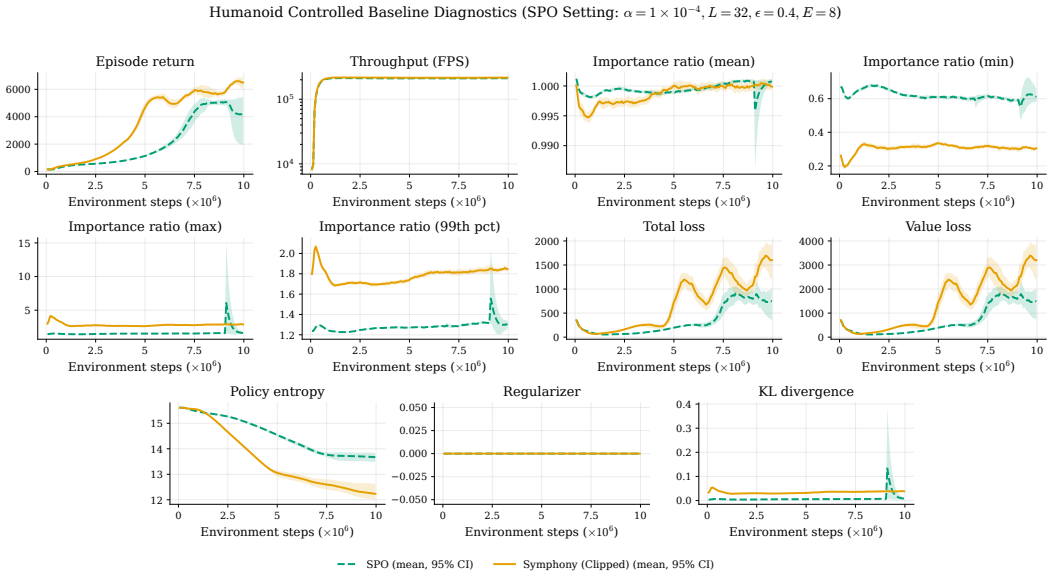


Figure 24: **Detailed Training Diagnostics for Humanoid under the SPO-Selected Setting** ($\alpha = 1 \times 10^{-4}, L = 32, \epsilon = 0.4, E = 8$)**.** Plots track 11 optimization metrics across 4 fresh random seeds over 10M transitions: episode return, throughput (FPS), importance ratios (mean, min, max, 99th pct), total loss, value loss, policy entropy, regularizer, and approximate KL divergence. Curves report mean values with 95% bootstrap confidence interval shading.

E.6 Material Differences: Published SPO Pipeline vs. Common Experimental Implementation

To ensure complete scientific transparency and auditability, Table 12 provides an exhaustive, code-level comparison between the published SPO implementation (Xie et al., 2025) (from their reference codebase consisting of `trainer.py`, `main.py`, and `agent.py` in the `mujoco/` directory of the released repository) and our standardized, common experimental harness implemented in `codes/run_mjx.py`.

Surrogate-Centric Evaluation vs. Pipeline-Level Engineering. A central methodological priority of our empirical study is to rigorously isolate the mathematical mechanics of the policy surrogate objective from confounding pipeline-level engineering choices. In empirical reinforcement learning, auxiliary heuristics—such as running reward scaling, reward clipping, learning-rate decay schedules,

value-loss clipping, and deep non-standard network architectures—frequently interact with or mask the stability of policy updates. When an algorithm is evaluated exclusively within an idiosyncratic pipeline containing multiple stabilizing heuristics, it is difficult to determine whether observed performance stems from the core surrogate objective or from the external scaffolding (such as forcing the learning rate to zero or squashing reward variance).

In our experimental framework (`codes/run_mjx.py`), we deliberately adopt a **surrogate-centric evaluation**:

1. **Exact Policy Objective Equivalence:** Both codebases evaluate the identical mathematical SPO loss:
$$L^{\text{SPO}}(\theta) = -\hat{\mathbb{E}}_t \left[r_t(\theta)\hat{A}_t - \frac{|\hat{A}_t|}{2\epsilon}(r_t(\theta) - 1)^2 \right]. \tag{60}$$
In SPO's `mujoco/trainer.py`, this is implemented as:
```
policy_loss = -(mb_advantages * ratios - torch.abs(mb_advantages) *
    torch.pow(ratios - 1, 2) / (2 * self.args.epsilon)).mean()
```
which identically matches line 237 of our `codes/run_mjx.py`:
```
policy_loss = -(ratio * adv - (jnp.abs(adv) * jnp.square(ratio - 1.0))
                    / (2 * epsilon)).mean()
```
Both implementations compute the advantage-weighted quadratic penalty regularizer.
2. **Raw, Unmodified Reward Dynamics:** The published SPO pipeline wraps Gymnasium environments with `NormalizeReward(gamma=0.99)` and `TransformReward(clip=[-10, 10])`. In `run_mjx.py`, we leave raw environment rewards completely unnormalized and unclipped. This directly exposes each surrogate objective to the true scale and variance of the task return signal, preventing artificial reward compression from masking optimization instability.
3. **Constant Learning Rate (No Annealing):** Published SPO enables linear learning-rate decay by default (`learning_rate_decay=True`), which forces late-stage convergence by monotonically reducing step sizes toward zero. In our benchmarks, we hold the learning rate strictly constant throughout training, directly stress-testing whether negative curvature and surrogate bounds can prevent asymptotic ratio divergence without relying on step-size decay.
4. **Standard Architecture and Principled Action Handling:** Rather than employing SPO's 7-layer Tanh MLP actor and post-hoc action truncation (`ClipAction`), our common runner uses standard 2-layer ReLU MLPs (256 hidden units) and continuous action squashing via $\tanh(u)$ with exact analytic Jacobian log-determinant correction.
5. **Strict Common Harness Across All Baselines:** All evaluated algorithms—PPO, SPO, First-Order KL, and Symphony (Clipped)—are evaluated within this exact same JAX harness. This ensures that every observed performance difference is directly attributable to the surrogate objective's intrinsic curvature and bounding mechanics rather than external software differences.

Table 12: **Comprehensive Comparison of Published SPO Codebase (Xie et al., 2025) and Our Common Implementation (`run_mjx.py`).** Both setups evaluate the identical regularized surrogate policy loss, but differ across system-level training pipeline choices. Our study evaluates all algorithms (PPO, SPO, First-Order KL, and Symphony) within this common harness to strictly isolate surrogate objective mechanics from pipeline-level engineering confounding.

Pipeline Component	Published SPO Codebase (Xie et al., 2025)	Our Common Implementation (`run_mjx.py`)	Surrogate-Centric Isolation Rationale
Surrogate Policy Loss	$-(r\hat{A} - \frac{\lvert\hat{A}\rvert}{2\epsilon}(r-1)^2)$	Identical: $-(r\hat{A} - \frac{\lvert\hat{A}\rvert}{2\epsilon}(r-1)^2)$	Exact mathematical equivalence; evaluates identical objective.
Actor Architecture	7-layer MLP ($6 \times$ Tanh hidden: 256, 256, 128, 128, 64, 64)	Standard 2-layer MLP (2×256 ReLU hidden)	Standard continuous-control MLP; avoids deep architectural bias.
Critic Architecture	2-layer MLP with 64 hidden units (2×64 Tanh hidden)	Standard 2-layer MLP (2×256 ReLU hidden)	Expressive, symmetric baseline capacity across high-dimensional tasks.
Initial Policy Log-Std	$\log \sigma = 0.0$ ($\sigma = 1.0$) unclipped	$\log \sigma = -0.5$ ($\sigma \approx 0.606$), clipped $[-20.0, 2.0]$	Standard initial exploration scale; bounds numerical stability.
Action Space & Bounding	Unbounded Gaussian $\rightarrow$ post-sample hard clip $[-1, 1]$	Gaussian $u \sim \mathcal{N} \rightarrow a = \tanh(u) \in [-1, 1]$	Principled squashing with exact analytic Jacobian log-det correction.
Reward Normalization	Running return scaling + clip $[-10, 10]$	Raw environment rewards unnormalized & unclipped	Exposes surrogate directly to true task reward scale and variance.
Observation Processing	Running mean/var norm + clip $[-10, 10]$	Online running mean/var norm (Welford RMS), unclipped	Standard continuous-control state scaling without artificial clipping.
Learning Rate Schedule	Linear decay schedule annealing $\alpha \rightarrow 0$	Constant learning rate throughout training	Stress-tests asymptotic stability without forced step-size shrinkage.
Value Loss Formulation	Clipped MSE $\frac{1}{2}\max((V-\hat{R})^2, (V_{\text{clip}}-\hat{R})^2)$ ($c_1 = 0.5$)	Unclipped MSE $(V-\hat{R})^2$ ($c_{vf} = 0.5$)	Eliminates value-clipping confounder; isolates policy surrogate.
Advantage Normalization	Standardized per sampled minibatch	Standardized per sampled minibatch	Identical per-minibatch standardization across both codebases.
Optimizer & Epsilon	PyTorch Adam ($\epsilon_{\text{opt}} = 1 \times 10^{-8}$, max norm 0.5)	Flax/Optax AdamW ($\epsilon_{\text{opt}} = 1 \times 10^{-5}$, max norm 1.0)	Modern standard optimizer settings for vectorized execution.
Rollout Buffer Size	8 envs $\times 256$ steps $= 2048$ transitions/buffer	2048 envs $\times 32$ steps $= 65{,}536$ transitions/buffer	Vectorized parallel simulation isolating sample efficiency at scale.
Minibatch Size & Epochs	4 minibatches of size 512, 10 update epochs	Minibatch size 2048, 8 update epochs (confirmed setting)	Efficient large-batch SGD optimization on modern GPU hardware.
Execution Framework	PyTorch CPU/GPU hybrid with Python multiprocessing	Pure JAX/Flax compiled on GPU (`jit`, `lax.scan`)	Eliminates CPU-GPU transfers; ensures strict determinism and speed.

We highlight two crucial methodological principles regarding this comparison:

1. **No unverified causal claims:** We do *not* claim that these pipeline differences explain SPO's empirical performance gap in our benchmarks without direct ablation testing. Isolating the individual effect of each auxiliary component (e.g., reward scaling vs. learning-rate annealing vs. 7-layer Tanh architecture) would require an exhaustive factorial ablation across all external pipeline components that was not tested.
2. **Preservation of experimental integrity:** All primary continuous and discrete benchmark comparisons (Sections 3 and E.1–E.5) evaluate algorithms within our shared common implementation. We do not describe our experiments as a reproduction of the published SPO software package; our empirical comparisons in those sections isolate the mathematical surrogate objective under identical, controlled implementation conditions. Section E.7 below evaluates transfer under an alternative SPO-style pipeline.

E.7 Direct Evaluation under the SPO-Style Implementation (128 Envs × 64 Steps)

To test whether performance differences persist under an alternative training pipeline and evaluate how Symphony transfers to that setting, we adopt the SPO-style configuration from the reference codebase (Xie et al., 2025) (`trainer.py`, `agent.py`, `main.py` in `mujoco/`). While the reference implementation uses 8 environments × 256 rollout steps (2,048 steps/batch), our evaluation adapts this to 128 parallel environments × 64 rollout steps (8,192 steps/batch) for accelerated GPU execution in Brax/MJX, using a faithful translation of SPO to JAX/Flax. Training runs over 13M environment steps (12,992,512 transitions) with Adam ($\alpha = 3 \times 10^{-4}$ linearly decaying to zero), running reward normalization clipped to $[-10, 10]$, clipped value loss ($\epsilon_V = 0.2$), clipping threshold $\epsilon = 0.2$, 10 update epochs, and gradient clipping at 0.5 across 4 identical seeds (seeds 1–4).

We evaluate six configurations across these identical four seeds:

1. **Symphony (Clipped):** Symphony ($n = 2, \epsilon = 0.2$) in our primary pipeline (2×256 ReLU, unnormalized rewards, constant LR, squashed actions).
2. **PPO:** Standard PPO ($\epsilon = 0.2$) in our primary pipeline under identical architecture and settings.
3. **SPO:** SPO ($L^{\mathrm{SPO}}, \epsilon = 0.2$) in our primary pipeline under identical architecture and settings.
4. **SPO (SPO-style MJX):** SPO ($L^{\mathrm{SPO}}, \epsilon = 0.2$) with 7-layer Tanh network (256-256-128-128-64-64), reward normalization, linear LR decay, and hard-clipped actions.
5. **PPO (SPO-style MJX):** Standard PPO ($\epsilon = 0.2$) substituted into the SPO-style pipeline with identical 7-layer architecture.
6. **Symphony (SPO-style MJX):** Symphony ($n = 2, \epsilon = 0.2$) substituted into the SPO-style pipeline with identical 7-layer architecture.

Table 13: **Implementation Comparison Across Evaluated Pipelines in the 13M-Step Study.** All configurations evaluate 128 parallel environments collecting 64 rollout steps (8,192 transitions per buffer) with minibatch size 2048 and 10 update epochs over 13M transitions across seeds 1–4. Symphony (Clipped), PPO, and SPO evaluate within our primary pipeline; Symphony (SPO-style MJX), PPO (SPO-style MJX), and SPO (SPO-style MJX) evaluate within the SPO-style MJX pipeline.

Component	Primary Evaluation Pipeline (Ours)	SPO-Style MJX Pipeline
Evaluated Surrogates	Symphony (Clipped), PPO, SPO	Symphony (Clipped), PPO, SPO
Actor Architecture	Standard 2-layer MLP (2×256 ReLU)	7-layer MLP ($6 \times$ Tanh: $256, 256, 128, 128, 64, 64$)
Critic Architecture	Standard 2-layer MLP (2×256 ReLU)	2-layer MLP (2×64 Tanh)
Optimizer	AdamW ($\epsilon_{\mathrm{opt}} = 1 \times 10^{-5}$, max norm 0.5)	Adam ($\epsilon_{\mathrm{opt}} = 1 \times 10^{-8}$, max norm 0.5)
Learning Rate	$\alpha = 3 \times 10^{-4}$, constant	$\alpha = 3 \times 10^{-4}$, linear decay to 0
Reward Processing	Raw environment rewards (unclipped)	Running return scaling + clip $[-10, 10]$
Observation Processing	Online Welford RMS (unclipped)	Running mean/var norm + clip $[-10, 10]$
Value Loss	Unclipped MSE $(V - \hat{R})^2$ ($c_{\mathrm{vf}} = 0.5$)	Clipped MSE with internal $\frac{1}{2}$ ($c_1 = 0.5$)
Action Bounding	Squashed $\tanh(u)$ with log-det	Gaussian → hard clip $[-1, 1]$

Confirmation Results and Analysis. As detailed in Table 14 and Figure 25:

- **Symphony (Clipped):** Reaches $\mathbf{6763.00 \pm 548.68}$ mean final return (95% bootstrap CI: $[6284.64, 7168.92]$) and AUC $\mathbf{5446.90 \pm 453.80}$ ($[4999.47, 5780.89]$).
- **PPO:** Reaches 5972.79 ± 910.14 mean final return (95% CI: $[5437.89, 6855.83]$) and AUC 5083.75 ± 314.68 ($[4813.81, 5353.69]$).
- **SPO:** Reaches 449.24 ± 48.81 mean final return (95% CI: $[405.93, 490.71]$) and AUC 702.47 ± 60.70 ($[656.13, 760.06]$).
- **SPO (SPO-style MJX):** Reaches 4768.91 ± 353.85 mean final return (95% CI: $[4511.00, 5084.06]$) and AUC 2354.14 ± 427.45 ($[1984.62, 2723.65]$).
- **PPO (SPO-style MJX):** Reaches 1640.59 ± 1181.57 mean final return (95% CI: $[974.43, 2830.90]$) and AUC 1281.60 ± 722.25 ($[885.26, 1993.95]$).
- **Symphony (SPO-style MJX):** Reaches 1138.61 ± 174.43 mean final return (95% CI: $[995.81, 1272.24]$) and AUC 968.05 ± 113.94 ($[868.07, 1050.77]$).

These six-way confirmation runs provide two key insights: First, within our primary codebase, both Symphony (6763.00 ± 548.68) and PPO (5972.79 ± 910.14) learn effectively, with Symphony holding a $+790.21$ return advantage over PPO under identical 128×64 conditions, whereas SPO collapses to 449.24 ± 48.81 without its external training heuristics. Second, when transferred to the SPO-style MJX pipeline without retuning, both standard surrogates suffer steep collapses: PPO drops by 72.5% to 1640.59 (with high variance, $SD = 1181.57$), and Symphony drops to 1138.61. In contrast, SPO achieves 4768.91. This shows reciprocal sensitivity: each surrogate's empirical behavior is tightly coupled to its native pipeline choices (network depth, reward normalization, action clipping).

Table 14: **Audit under the SPO-Style Implementation** (128 **Envs** $\times$ 64 **Steps, 13M Transitions).** Evaluated across seeds 1, 2, 3, 4. All metrics report mean $\pm$ sample SD along with 95% bootstrap confidence intervals across seeds.

Method	Final Return (Mean $\pm$ SD)	Final Return [95% CI]	Normalized AUC (Mean $\pm$ SD)	Normalized AUC [95% CI]
Primary Implementation Pipeline (2 $\times$ 256 ReLU, raw rewards, unclipped Welford, squashed actions)				
Symphony (Clipped)	$\mathbf{6763.00 \pm 548.68}$	[6284.64, 7168.92]	$\mathbf{5446.90 \pm 453.80}$	[4999.47, 5780.89]
PPO	5972.79 ± 910.14	[5437.89, 6855.83]	5083.75 ± 314.68	[4813.81, 5353.69]
SPO	449.24 ± 48.81	[405.93, 490.71]	702.47 ± 60.70	[656.13, 760.06]
SPO-Style MJX Pipeline (6 $\times$ Tanh actor, reward norm, clipped value, hard-clipped actions)				
SPO (SPO-style MJX)	4768.91 ± 353.85	[4511.00, 5084.06]	2354.14 ± 427.45	[1984.62, 2723.65]
PPO (SPO-style MJX)	1640.59 ± 1181.57	[974.43, 2830.90]	1281.60 ± 722.25	[885.26, 1993.95]
Symphony (SPO-style MJX)	1138.61 ± 174.43	[995.81, 1272.24]	968.05 ± 113.94	[868.07, 1050.77]

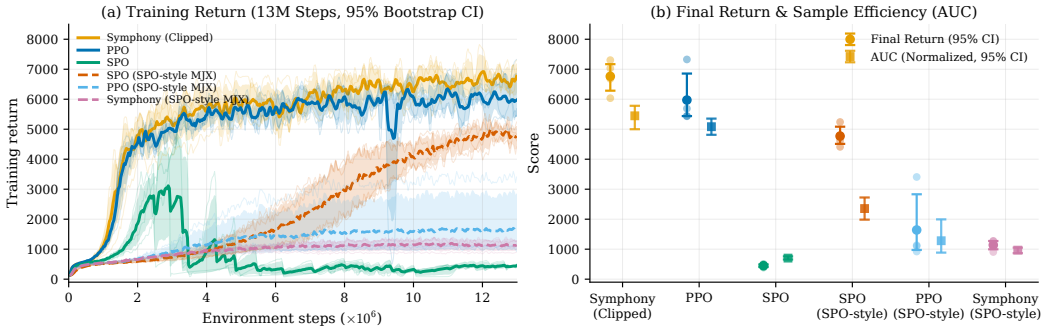


Figure 25: **Direct Evaluation under the SPO-Style Implementation** (128 **Envs** $\times$ 64 **Steps, 13M Transitions).** Six-configuration evaluation comparing Symphony (Clipped), PPO, and SPO across both our primary pipeline and the SPO-style MJX pipeline across seeds 1–4. *Left:* Combined training curves over 13M steps with solid/dashed mean lines, 95% bootstrap confidence interval shading, and faint individual seed trajectories. *Right:* Final evaluation return and normalized area under the curve (AUC) with 95% bootstrap confidence interval error bars and individual seed scatter points.

F IMPLEMENTATION DETAILS

The primary benchmark evaluations (two-billion-step continuous control on MuJoCo MJX and 25-million-step visual control on JAXAtari) were conducted on a compute cluster utilizing NVIDIA RTX 6000 Ada Generation GPUs with 48GB of VRAM. The 10M-step controlled baseline confirmation experiments on Humanoid were executed on dedicated NVIDIA GPU instances in Google Colab (with execution times and hardware environments documented in each trial manifest).

F.1 JAXATARI IMPLEMENTATION DETAILS

For the JAXAtari discrete control benchmarks, we process the environment observations by converting them to grayscale, resizing them to 84×84 pixels, applying max-pooling, and stacking the 4 most recent frames. To introduce stochasticity into the environment dynamics, we apply a 25% probability of sticky actions (repeating the previous action).

Neural Network Architecture (`ConvVAEActorCritic`). Our visual control agent employs a Convolutional Variational Autoencoder Actor-Critic (`ConvVAEActorCritic`) implemented in Flax/JAX. The network processes the preprocessed 4-frame stack $x \in [0,1]^{84\times84\times4}$ through three dedicated functional submodules:

1. **Convolutional Encoder:** The observation is processed through three convolutional layers with decreasing kernel sizes and strides (8×8 stride 4, 4×4 stride 2, and 3×3 stride 1) with orthogonal kernel initialization (gain $\sqrt{2}$) and ReLU activations. The flattened spatial features ($7 \times 7 \times 64 = 3136$) are projected through a 512-unit fully connected layer. Two parallel linear heads then project this feature representation into the latent distribution parameters: the latent mean $\mu \in \mathbb{R}^{128}$ and latent log-variance $\log \sigma^2 \in \mathbb{R}^{128}$ (clamped to $[-10.0, 10.0]$ for numerical stability).

2. **Actor and Critic Heads:** To eliminate stochastic sampling noise from corrupting policy and value updates, the actor and critic heads condition directly on the *deterministic* latent mean $z_{\text{det}} = \mu$. The actor head routes z_{det} through a 256-unit dense layer with ReLU and projects to unnormalized action logits $\in \mathbb{R}^{|\mathcal{A}|}$ (orthogonal initialization with gain 0.01). The critic head similarly routes z_{det} through an independent 256-unit dense layer with ReLU and projects to a scalar baseline value $V(s) \in \mathbb{R}$ (orthogonal initialization with gain 1.0).

3. **Observation Reconstruction Decoder (Auxiliary Decodability Probe):** To monitor whether the learned latent representation preserves sufficient spatial visual features without allowing pixel reconstruction loss to distort or compete with policy-gradient updates, an auxiliary decoder reconstructs the most recent observation frame from a stochastic latent sample $z_{\text{stoch}} = \mu + \epsilon \odot \exp(0.5 \log \sigma^2)$, where $\epsilon \sim \mathcal{N}(0, I)$. Crucially, as implemented in code via `jax.lax.stop_gradient(`z_{stoch}`)`, reconstruction gradients do *not* backpropagate into the convolutional encoder; the decoder functions strictly as an auxiliary readout probe (with an auxiliary latent consistency penalty $0.1\|\mu - \text{stop_gradient}(z_{\text{stoch}})\|_2^2$ regularizing the latent space, and global gradient clipping at 0.5). The decoder expands the detached latent sample through a 512-unit dense layer and a 7744-unit dense layer (reshaped to $11 \times 11 \times 64$), followed by three transposed convolutional layers with SAME padding. The final stride-4 transposed convolution generates an $88 \times 88 \times 1$ spatial map, which is center-cropped to $84 \times 84 \times 1$ with sigmoid activation to produce the reconstructed frame $\hat{x}$, whose MSE loss is tracked as a diagnostic metric across training (Figures 16–22).

Table 15 details the exact layer-by-layer specification of the `ConvVAEActorCritic` architecture, while Table 16 provides the training and optimization hyperparameters. All hyperparameters were held strictly identical across all benchmarked algorithms.

Table 15: JAXAtari Conv-VAE Actor-Critic Neural Network Architecture (`ConvVAEActorCritic`)

Component / Layer	Type	Output Shape	Kernel / Stride	Activation	Initialization
Shared Visual Encoder					
Input Observation	Input	$84 \times 84 \times 4$	—	—	—
Encoder Conv 1	Conv2D	$20 \times 20 \times 32$	8×8 / 4	ReLU	Orthogonal ($\sqrt{2}$)
Encoder Conv 2	Conv2D	$9 \times 9 \times 64$	4×4 / 2	ReLU	Orthogonal ($\sqrt{2}$)
Encoder Conv 3	Conv2D	$7 \times 7 \times 64$	3×3 / 1	ReLU	Orthogonal ($\sqrt{2}$)
Encoder Dense	Dense	512	—	ReLU	Orthogonal ($\sqrt{2}$)
Latent Mean Head (μ)	Dense	128	—	Linear	Orthogonal (0.01)
Latent Log-Var Head ($\log \sigma^2$)	Dense	128	—	Linear (clip $[-10, 10]$)	Orthogonal (0.01)
Policy Head (from $z_{\text{det}} = \mu$)					
Policy Hidden	Dense	256	—	ReLU	Orthogonal ($\sqrt{2}$)
Action Logits Head	Dense	$\lvert\mathcal{A}\rvert$	—	Linear	Orthogonal (0.01)
Value Head (from $z_{\text{det}} = \mu$)					
Value Hidden	Dense	256	—	ReLU	Orthogonal ($\sqrt{2}$)
Value Scalar Head	Dense	1	—	Linear	Orthogonal (1.0)
Decoder (reparameterized sample, stop-gradient)					
Decoder Dense 1	Dense	512	—	ReLU	Orthogonal ($\sqrt{2}$)
Decoder Dense 2	Dense	7744	—	ReLU (reshape $11 \times 11 \times 64$)	Orthogonal ($\sqrt{2}$)
Transposed Conv 1	ConvTranspose	$11 \times 11 \times 64$	3×3 / 1 (SAME)	ReLU	Orthogonal ($\sqrt{2}$)
Transposed Conv 2	ConvTranspose	$22 \times 22 \times 32$	4×4 / 2 (SAME)	ReLU	Orthogonal ($\sqrt{2}$)
Transposed Conv 3	ConvTranspose	$88 \times 88 \times 1$	8×8 / 4 (SAME)	Linear	Orthogonal ($\sqrt{2}$)
Center Crop & Activation	Crop	$84 \times 84 \times 1$	Center 84×84	Sigmoid	—

Table 16: JAXAtari Comprehensive Hyperparameters and Training Configuration

Hyperparameter / Setting	Value
Environment & Rollout	
Observation Resolution	84x84 Grayscale
Frame Stack	4
Sticky Actions Probability	0.25
Number of Environments	256
Steps per Rollout	128
Network Architecture Summary	
Encoder Channels	32, 64, 64
Encoder Kernel Sizes	8x8, 4x4, 3x3
Encoder Strides	4, 2, 1
Latent Dimension	128
Policy / Value MLP Hidden Units	256
Decoder Transposed Conv Channels	64, 32, 1
Algorithm & Optimization	
Optimizer	AdamW
Learning Rate	8×10^{-4}
Optimizer Epsilon	1×10^{-5}
Minibatch Size	2048
Update Epochs	4
Discount Factor (γ)	0.99
GAE Parameter (λ_{GAE})	0.95
Max Gradient Norm	0.5
Objective Coefficients & Symphony	
Symphony Order (n)	2
Symphony Penalty (λ_{sym})	0.5
Clipping Parameter (ϵ)	0.2
First-Order KL-Penalized Baseline λ	1.0
Value Function Coefficient	0.5
Entropy Coefficient	0.0
Reconstruction Loss Coefficient	1.0
KL Penalty Coefficient (β)	0.1
Latent Consistency Scale	0.1

F.2 MuJoCo MJX Continuous Control Implementation Details

For the continuous control benchmarks, we utilize the Brax physics engine with the MuJoCo MJX backend. Unlike the visual Atari domains, these environments provide low-dimensional state vectors. To ensure stable optimization across varying state scales, we apply running observation normalization, tracking the mean and variance of the state distribution online via Welford's algorithm.

Neural Network Architecture (`ActorCritic`). The continuous control agent is parameterized by an `ActorCritic` module implemented in Flax/JAX consisting of two separate Multi-Layer Perceptron (MLP) towers:

1. **Actor (Policy Network):** The actor receives the normalized state vector $s \in \mathbb{R}^{\dim(\mathcal{S})}$ and routes it through two hidden layers of 256 units each, with ReLU activations and orthogonal weight initialization (gain $\sqrt{2}$). A linear output head with orthogonal initialization (gain 0.01) outputs the mean action vector $\mu(s) \in \mathbb{R}^{\dim(\mathcal{A})}$. The action log standard deviation $\log \sigma \in \mathbb{R}^{\dim(\mathcal{A})}$ is parameterized as a state-independent learnable parameter vector, initialized to constant -0.5 and clipped to $[-20.0, 2.0]$ during evaluation. Actions are sampled as $u \sim \mathcal{N}(\mu(s), \sigma^2)$ and squashed via $\tanh$ into $[-1, 1]^{\dim(\mathcal{A})}$. The exact log-likelihood is corrected via the log-determinant of the Jacobian:

$$\log \pi(a \mid s) = \log \mathcal{N}(u; \mu(s), \sigma^2) - \sum_{i=1}^{\dim(\mathcal{A})} \left[2\left(\log 2 - u_i - \mathrm{softplus}(-2u_i)\right)\right]. \tag{61}$$

2. **Critic (Value Network):** The value network receives the identical normalized state $s \in \mathbb{R}^{\dim(\mathcal{S})}$ and processes it through an independent two-layer MLP with 256 units per hidden layer, ReLU activations, and orthogonal initialization (gain $\sqrt{2}$). A final linear head with orthogonal initialization (gain 1.0) outputs the scalar baseline prediction $V(s) \in \mathbb{R}$.

Table 17 details the layer-by-layer breakdown of the continuous control architecture, and Table 18 summarizes the corresponding training hyperparameters.

Table 17: MuJoCo MJX Actor-Critic Neural Network Architecture (`ActorCritic`)

Component / Layer	Type	Output Shape	Activation	Weight Initialization	Bias Init
Actor (Policy MLP)					
Input State	Input	$\dim(\mathcal{S})$	Running Mean/Var	—	—
Policy Hidden 1	Dense	256	ReLU	Orthogonal ($\sqrt{2}$)	Zero
Policy Hidden 2	Dense	256	ReLU	Orthogonal ($\sqrt{2}$)	Zero
Action Mean Head	Dense	$\dim(\mathcal{A})$	Linear	Orthogonal (0.01)	Zero
Action Log-Std ($\log \sigma$)	Parameter	$\dim(\mathcal{A})$	Clip $[-20, 2]$	Constant (-0.5)	—
Action Bounding	Squashing	$\dim(\mathcal{A})$	$\tanh$	Analytic Jacobian Corr.	—
Critic (Value MLP)					
Input State	Input	$\dim(\mathcal{S})$	Running Mean/Var	—	—
Value Hidden 1	Dense	256	ReLU	Orthogonal ($\sqrt{2}$)	Zero
Value Hidden 2	Dense	256	ReLU	Orthogonal ($\sqrt{2}$)	Zero
Value Scalar Head	Dense	1	Linear	Orthogonal (1.0)	Zero

Table 18: MuJoCo MJX Comprehensive Hyperparameters and Training Configuration

Hyperparameter / Setting	**Value**
Environment & Rollout	
Physics Backend	Brax (MJX)
Observation Processing	Running Mean/Variance Normalization
Number of Environments	2048
Steps per Rollout	2048
Network Architecture Summary	
Policy MLP Hidden Units	256, 256
Value MLP Hidden Units	256, 256
Activation Function	ReLU
Weight Initialization	Orthogonal ($\sqrt{2}$ for hidden, 0.01/1.0 for heads)
Action Distribution	Gaussian (State-independent learned $\log \sigma$)
Action Bounding	Tanh Squashing
$\log \sigma$ Bounds	$[-20.0, 2.0]$
Algorithm & Optimization	
Optimizer	AdamW
Learning Rate	8×10^{-4}
Optimizer Epsilon	1×10^{-5}
Minibatch Size	2048
Update Epochs	4
Discount Factor (γ)	0.99
GAE Parameter (λ_{GAE})	0.95
Max Gradient Norm	1.0
Objective Coefficients & Symphony	
Symphony Order (n)	2
Symphony Penalty (λ_{sym})	0.5
Clipping Parameter (ϵ)	0.2
First-Order KL-Penalized Baseline λ	1.0
Value Function Coefficient	0.5
Entropy Coefficient	0.0
KL Penalty Coefficient (β)	0.0

F.3 Standardized Objective Implementations and Baseline Scope

To strictly isolate algorithmic mechanisms from confounding implementation differences, all benchmarked algorithms are evaluated within our standardized, common Flax/JAX training codebase. They share identical network architectures, orthogonal weight initializations, AdamW optimizer hyperparameters, observation normalization (running mean and variance updated online), and batch advantage standardization:

- **Proximal Policy Optimization (PPO):** Implements the standard clipped likelihood-ratio objective $L^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right]$ (Schulman et al., 2017).

- **Simple Policy Optimization (SPO Objective):** Implements the first-order surrogate objective of Xie et al. (2025) directly inside our shared training pipeline. We explicitly clarify that our study evaluates the *SPO objective function* under identical training infrastructure, rather than reproducing the external published pipeline, custom heuristics, or third-party codebase of Xie et al. (2025) (see Section E.6 and Table 12 for an itemized comparison). Our findings address the narrower, well-supported claim that Symphony outperforms the independently tuned SPO objective within this common implementation.

- **First-Order KL-Penalized Baseline:** Implements the unclipped likelihood-ratio objective regularized by an explicit KL divergence penalty: $L^{\text{KL}}(\theta) = \hat{\mathbb{E}}_t \left[r_t(\theta)\hat{A}_t - \beta \, \text{KL}\left(\pi_{\theta_{\text{old}}}(\cdot \mid s_t) \parallel \pi_\theta(\cdot \mid s_t)\right) \right]$.

- **Symphony (Clipped and KL Variants):** Implements the n-th root power-law surrogate $f_n(r) = nr^{1/n} - (n-1)$, combined either with proximal boundary clipping at $[1-\epsilon, 1+\epsilon]$ or with the analytical KL divergence constraint as formulated in Section 2.

F.4 JAX Implementation of the Symphony Policy Loss

For completeness, Listing 1 gives the minimal JAX policy-loss branch corresponding to Algorithm 1. The implementation computes the Symphony (KL) and Symphony (Clipped) surrogates directly from the log-ratio for numerical stability; the unclipped Symphony (KL) branch also includes an optional standard KL penalty, while the clipped branch clips the log-ratio before applying the Symphony transform.

Listing 1: Minimal JAX implementation of the Symphony (KL) and Symphony (Clipped) policy-loss branches.

```python
from __future__ import annotations

from typing import Literal

import jax
import jax.numpy as jnp


PolicyVariant = Literal[
    "symphony_kl",
    "symphony_clipped",
]


def symphony_policy_loss(
    log_ratio: jax.Array,
    adv: jax.Array,
    *,
    variant: PolicyVariant,
    symphony_order: float = 2.0,
    epsilon: float = 0.2,
    kl_coef: float = 0.0,
) -> jax.Array:
    """Compute the Symphony policy loss.

    Args:
        log_ratio:
            Per-sample log probability ratio,
            log pi_theta(a_t | s_t) - log pi_old(a_t | s_t).
        adv:
            Per-sample advantage estimates.
        variant:
            Either 'symphony_kl' or 'symphony_clipped'.
        symphony_order:
            Transformation order n. The paper uses n = 2, with order n >= 2.
        epsilon:
            Clipping parameter for Symphony (Clipped).
        kl_coef:
            Coefficient for the standard KL penalty in the
            Symphony (KL) objective. Set kl_coef = 0.0 to disable it.

    Returns:
        Scalar policy loss to minimize.
    """

    pure_surrogate = (
        symphony_order * jnp.expm1(log_ratio / symphony_order) + 1.0
    )

    if variant == "symphony_kl":
        surrogate_loss = -(pure_surrogate * adv).mean()
        ratio = jnp.exp(log_ratio)
        approx_kl = jnp.mean((ratio - 1.0) - log_ratio)
        policy_loss = surrogate_loss + kl_coef * approx_kl

    elif variant == "symphony_clipped":
        clip_low = jnp.log(1.0 - epsilon)
        clip_high = jnp.log(1.0 + epsilon)
        clipped_log_ratio = jnp.clip(log_ratio, clip_low, clip_high)

        clipped_surrogate = (
            symphony_order * jnp.expm1(clipped_log_ratio / symphony_order) + 1.0
        )

        policy_term = jnp.minimum(
            pure_surrogate * adv,
            clipped_surrogate * adv,
        )
        policy_loss = -policy_term.mean()

    else:
        raise ValueError(f"Unknown policy-loss variant: {variant}")

    return policy_loss
```

G FREQUENTLY ASKED QUESTIONS

1. **Is Symphony just PPO with a nonlinear ratio transform?**
   ⇒ The practical clipped surrogate can indeed be viewed as applying the power-law transformation $f_n(r) = nr^{1/n} - (n-1)$ to PPO's importance-ratio surrogate. However, Symphony's core contribution lies in the theoretical and empirical framework built around this construction: (1) we derive this transformation from first-order consistency axioms and a finite binomial decomposition, explicitly bounding the discarded higher-order residual; (2) we prove that the power-law transformation preserves the policy-gradient direction at the anchor $r = 1$ and, under local KL geometry, the corresponding natural-gradient direction, while reducing the ratio-dependent factor in the gradient's trace second-moment matrix from $O(r^2)$ to $O(r^{2/n})$; (3) the framework provides a common root-ratio surrogate usable with either clipping or KL penalization via curvature substitution; and (4) our extensive empirical evaluation demonstrates that intrinsic negative curvature stabilizes high-dimensional continuous control (e.g., nearly tripling PPO's return on Humanoid) and offers resilience under relaxed clipping boundaries where standard clipping alone suffers from severe ratio variance.
2. **Why does Symphony preserve the policy-gradient direction?**
   ⇒ Symphony preserves the policy-gradient direction because $f_n(1) = 1$ and $f_n'(1) = 1$. At the old policy, $r_\theta = 1$, so the gradient of the Symphony surrogate reduces to the standard policy-gradient estimator. The method therefore does not change the local update direction at θ_{old}; it regularizes finite-displacement trajectories away from the anchor point.
3. **What exactly is gained by using the n-th root transform?**
   ⇒ The main gain is variance damping. In the standard likelihood-ratio surrogate, large ratios contribute with dominant second-moment scaling $O(r^2)$. Under Symphony, the ratio-dependent multiplier becomes $r^{1/n}$, so the corresponding trace second-moment scaling becomes $O(r^{2/n})$. For the default choice $n = 2$, this reduces quadratic growth to linear growth, which is especially useful in high-dimensional continuous control where multiplicative likelihood-ratio spikes are common.
4. **Does omitting the higher-order residual introduce bias?**
   ⇒ Yes, but the omitted term is explicitly bounded. The practical Symphony surrogate keeps the primary power-law term and drops the higher-order residual from the binomial expansion. When the root-ratio displacement $\left|r^{1/n} - 1\right|$ is small, the residual is second order in that displacement. This is why Symphony pairs naturally with clipping or KL penalties: both mechanisms keep policy ratios near unity, ensuring the omitted residual remains tightly controlled.
5. **Why use $n = 2$ as the default?**
   ⇒ The choice $n = 2$ is the first nonlinear member of the Symphony family. It gives a clean stability benefit by changing the dominant ratio-driven second-moment growth from $O(r^2)$ to $O(r)$, while still preserving much of the original improvement signal. Larger values of n damp variance more strongly, but they can also over-compress useful learning signals.
6. **If larger n gives stronger damping, why not always use a large n?**
   ⇒ The transformation order n controls a bias–stability tradeoff. Larger n values make the surrogate more conservative and reduce the influence of extreme likelihood ratios, but excessive damping can weaken useful improvement signals. The ablations show that the best order is environment-dependent: high-dimensional tasks such as Humanoid benefit from stronger damping, while mid-complexity tasks such as Ant can perform better with smaller orders.
7. **Why does Symphony still need clipping?**
   ⇒ Symphony and clipping stabilize different aspects of the optimization. Surrogate clipping removes the gradient incentive to push likelihood ratios beyond $[1-\epsilon, 1+\epsilon]$ when advantages align, but it leaves the unclipped surrogate completely linear within that interval. Symphony introduces negative curvature directly inside the active region where updates occur, moderating gradient contributions from moderately large ratios before they dominate. The clipped variant is complementary: clipping eliminates gradient incentives for advantageous overshoots, while Symphony introduces intrinsic negative curvature to dampen ratio-driven variance across the active update path.
8. **Why does Symphony help more on Humanoid and Ant than on smoother tasks?**
   ⇒ Symphony is most useful when likelihood-ratio variance is a major bottleneck. Humanoid has a 17-dimensional continuous action space, while Ant introduces contact-driven stochasticity. These settings create multiplicative ratio compounding where Symphony's intrinsic curvature is especially valuable. In smoother or discrete tasks, the gains are smaller because bounded likelihood ratios prevent extreme variance explosions.
9. **Is Symphony a replacement for TRPO-style trust regions?**
   ⇒ Not exactly. Symphony provides trust-region-like stability natively through first-order surrogate curvature without requiring second-order conjugate-gradient solves. Under the KL-penalized local analysis, Symphony preserves natural-gradient alignment while substituting external divergence penalization with intrinsic power-law damping.

10. **What are the main limitations reviewers should keep in mind?**
⇒ Symphony introduces an additional hyperparameter, the transformation order n, and the optimal amount of curvature is environment-dependent. Furthermore, Symphony operates natively on raw reward scales; dropping it into external pipelines with aggressive reward squashing or decaying learning rates degrades performance.

11. **How should the scope of the 2B extended-horizon evaluations versus the 10M Humanoid controlled study be interpreted?**
⇒ They evaluate complementary questions under distinct experimental scopes: (1) The extended-horizon evaluations (up to 2B transitions in MJX locomotion and 25M in JAXAtari) assess asymptotic stability, drift resilience, and sample efficiency under a standardized, shared configuration across continuous and discrete domains. (2) The 10M Humanoid study investigates whether Symphony (Clipped)'s empirical advantages persist when competitor baselines (PPO and SPO) receive dedicated hyperparameter tuning within our common implementation (screening over a 36-configuration factorial grid). This study addresses baseline-specific tuning on Humanoid within our common codebase; it does not claim tuned superiority across every benchmark, nor does it reproduce external published pipelines for baselines such as SPO.