FISHER-TRACE STATE SENSITIVITY FOR PROXIMAL POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep reinforcement learning with neural critics often suffers from unstable value estimates and overly aggressive policy updates. Motivated by the intuition that states where the critic is highly sensitive deserve more cautious updates, we derive a state-level sensitivity measure for Proximal Policy Optimization (PPO) from the Fisher-information trace of the critic. We prove that, under the training conditions of standard PPO, this trace reduces to the squared norm of the value gradient with respect to states, and batch-wise standardization removes the unknown global scale. The resulting algorithm, GFIM-PPO, rescales the PPO surrogate loss with a bounded coefficient computed from this measure. On a diverse suite of reinforcement learning benchmarks, GFIM-PPO improves over PPO on the majority of tasks. Our analysis further confirms that it is the state-dependent construction of the weighting coefficient, rather than weighting alone, that drives the gains. Code is available at https://anonymous.4open.science/r/CAAC-F11F.

1 INTRODUCTION

Deep reinforcement learning with neural critics often produces unstable value estimates and overly aggressive policy updates. Proximal Policy Optimization (PPO) (Schulman et al., 2017) mitigates this by clipping the probability ratio between the new and old policies, thereby limiting the size of any single update. Recent surveys review deep reinforcement-learning successes in robotics (Tang et al., 2025). Within this landscape, recent work has revisited the role of entropy in policy optimization: entropy as an exploration signal has been studied for reasoning tasks (Cheng et al., 2026), and entropy regularization has been used to make task representations independent of the behavior policy in offline meta-reinforcement learning (Nakhaeinezhadfard et al., 2025). PPO is the first-order successor to Trust Region Policy Optimization (TRPO) (Schulman et al., 2015a) and the natural policy gradient (Kakade, 2001), both of which use Fisher information as a metric on policy parameters (Amari, 1998).

In this work we extend PPO with a state-level sensitivity signal inspired by the Fisher information on the critic, not on the policy. The intuition is simple: if the critic value changes rapidly under small perturbations of a state, the advantage signal derived from it is easily perturbed, so the policy should place less trust in the directed PPO gradient and explore more. We make this precise by embedding the critic in a Gaussian return model and computing the Fisher information of the state. Under a homoscedastic variance assumption the resulting trace reduces to the squared norm of the critic state-gradient, and a batch-wise normalization step removes the unknown variance, so no variance estimate is needed. Figure 1 illustrates the effect: high-sensitivity states receive small weights and exploration-dominated updates, while smooth states follow the directed gradient.

We call the resulting algorithm GFIM-PPO (Gradient-based Fisher Information Metric for State Sensitivity). It maps the normalized sensitivity to a directed sample weight whose batch average is one, so the directed term keeps the scale of standard PPO. The complementary exploration weight is clipped from the same sensitivity signal: samples that are down-weighted on the directed side get a larger entropy share, and confident samples get none. The per-sample objective is a bounded non-negative mixture of the clipped PPO surrogate and the entropy bonus; the PPO clip structure stays intact.

Our contributions are as follows.

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

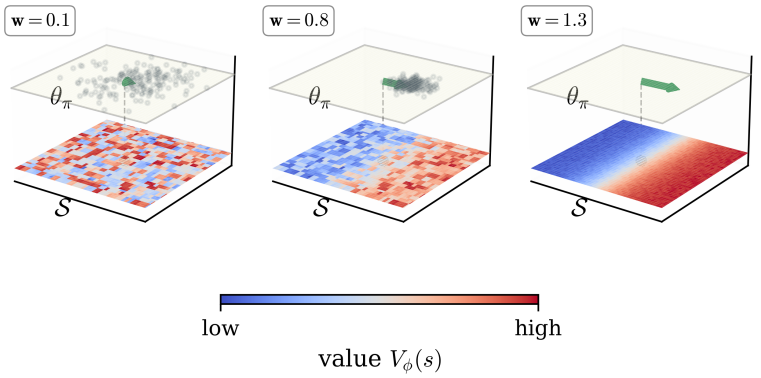


Figure 1: Conceptual illustration of GFIM-PPO. The lower plane is the state space $\mathcal{S}$ colored by the critic value landscape $V_\phi(s)$; the upper plane is the policy-parameter space θ_π. The green arrow is the directed PPO policy gradient and the dark dots are exploration noise. The weight w in each panel (top-left) shows how the normalized sensitivity maps to the trade-off: *Left* ($w = 0.1$), high sensitivity (short arrow, large cloud); *Center* ($w = 0.8$), moderate sensitivity; *Right* ($w = 1.3$), low sensitivity (long arrow, no cloud).

- We construct a Gaussian return model for the PPO critic and show that the trace of its Fisher information with respect to the state reduces to the squared state-gradient norm under a homoscedastic variance assumption.
- We introduce GFIM-PPO, which turns this trace into batch-renormalized per-sample weights that mix the PPO surrogate with an entropy bonus through a clipped complementary weight. The clipped surrogate is unchanged, the entropy share rescales automatically from the sensitivity signal, and the default configuration performs reasonably on most tasks, though the hyperparameters remain environment-dependent.
- We systematically study the design space around the sensitivity signal: four sensitivity estimators $\nu(s)$ and three ways of mapping the normalized sensitivity to a weight, characterizing how each choice behaves across environments and showing that the default is a robust rather than fragile choice.
- On a 14-task benchmark spanning classic control and MuJoCo (6 seeds), GFIM-PPO improves over PPO on 12 tasks and wins on all 8 MuJoCo tasks (5 of them significant after a Holm correction within each task), at modest computational overhead.

2 Related Work

Policy optimization and Fisher information. Fisher information has long been used in policy optimization as a metric on policy parameters: the natural policy gradient (Amari, 1998; Kakade, 2001) preconditioned the policy update by the inverse Fisher matrix, TRPO (Schulman et al., 2015a) enforced a KL trust region in the same geometry, and Kronecker-factored approximations (Wu et al., 2017) scaled this to deep actor-critics. PPO (Schulman et al., 2017) achieves similar stability with first-order optimization by clipping the probability ratio; its behavior is known to depend on implementation details (Engstrom et al., 2020; Andrychowicz et al., 2021). We use Fisher information in a different place: not on the policy parameters but on the state, as a per-sample sensitivity signal derived from the critic.

Sample weighting and prioritization. Several lines of work reweight individual samples to improve learning. Prioritized experience replay (Schaul et al., 2015) samples transitions by temporal-difference error; emphatic TD (Sutton et al., 2016) corrects the off-policy state distribution. Our weighting signal is the critic's sensitivity to the state rather than a prediction error, and it is applied to a per-sample mixture of the PPO surrogate and an entropy bonus.

Input-gradient sensitivity. The gradient norm $\|\nabla_s V_\phi(s)\|^2$ has been used as an input-Jacobian penalty in double backpropagation (Drucker & Le Cun, 1992), contractive auto-encoders (Rifai et al., 2011), and Sobolev training (Czarnecki et al., 2017). In all of these the gradient norm enters the training loss as a regularizer. In our method it never enters the critic loss; it only produces a per-sample weight for the actor, so critic training is unchanged.

Uncertainty and exploration. The most closely related work is normality-guided distributional RL (Byun & Perrault, 2022), which also assumes a Gaussian value model and reweights the PPO objective per sample. Its uncertainty signal, however, is the deviation of a learned quantile distribution from a Gaussian shape, which requires a distributional critic (Bellemare et al., 2017; Dabney et al., 2018) and a separate variance network. Our weight comes from the Fisher trace of the Gaussian critic, which reduces to $\|\nabla_s V_\phi(s)\|^2$ and needs no extra network. Other exploration methods (value-conditional state entropy (Kim et al., 2023), curiosity (Pathak et al., 2017), risk-aware pessimistic updates (Nauman & Cygan, 2025), and maximum-entropy objectives (Haarnoja et al., 2018)) add a bonus to the reward or the objective. We keep the objective unchanged and instead redistribute each sample between the directed update and exploration through a bounded weight.

3 Preliminaries

We consider a discounted infinite-horizon Markov decision process (MDP) (Sutton & Barto, 2018) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, with state space $\mathcal{S}$, action space $\mathcal{A}$, transition kernel P, reward function r, and discount factor $\gamma \in [0, 1)$. A stochastic policy $\pi_\theta(\cdot \mid s)$ is parameterized by θ, and the return from time t is $R_t := \sum_{i=t}^{\infty} \gamma^{i-t} r(s_i, a_i)$. The state-value and action-value functions are $V^\pi(s) := \mathbb{E}[R_t \mid s_t = s]$ and $Q^\pi(s, a) := \mathbb{E}[R_t \mid s_t = s, a_t = a]$. Actor-critic methods (Mnih et al., 2016) build on the policy gradient theorem (Sutton et al., 1999): the critic $V_\phi(s)$ approximates $V^\pi(s)$ with parameters ϕ, and the actor is updated in a direction suggested by the critic.

3.1 Proximal Policy Optimization

PPO (Schulman et al., 2017) is a first-order alternative to trust-region policy optimization (Schulman et al., 2015a), which itself builds on the natural policy gradient (Kakade, 2001). It optimizes the clipped surrogate objective

$$\ell_i^{\mathrm{PPO}}(\theta) := \min\Big\{ r_i(\theta) A_i,\ \mathrm{clip}\big(r_i(\theta), 1-\epsilon, 1+\epsilon\big) A_i \Big\}, \tag{1}$$

where

$$r_i(\theta) := \frac{\pi_\theta(a_i \mid s_i)}{\pi_{\theta_{\mathrm{old}}}(a_i \mid s_i)} \tag{2}$$

is the probability ratio and A_i is an advantage estimate, typically computed with generalized advantage estimation (Schulman et al., 2015b). The clip operator keeps the ratio inside $[1-\epsilon, 1+\epsilon]$, which bounds how far the new policy can move from the old policy on this sample. The critic is trained by minimizing the mean squared error

$$L^{\mathrm{VF}}(\phi) := \mathop{\mathbb{E}}_t \Big[\big(V_\phi(s_t) - V_t^{\mathrm{target}} \big)^2 \Big], \tag{3}$$

where V_t^{target} is a target value, often computed by bootstrapping or from an n-step return.

4 Method

The method augments PPO with a state-level sensitivity signal derived from the Fisher information of a Gaussian critic model. We first define the model and derive the Fisher trace for a general state-dependent variance. Under the homoscedastic assumption that matches standard PPO training, the trace simplifies to the squared state-gradient norm divided by a constant variance, and batch-wise standardization removes this constant. The normalized sensitivity is mapped to per-sample weights by a softmax-renormalized (SRN) mapping whose weights average to one, and a clipped complementary weight assigns the entropy share of each sample. A family of modulated sensitivity estimators beyond the plain Fisher trace is defined in the appendix and evaluated in the ablation studies.

4.1 Gaussian Critic Model

Let Y denote the random return observed when starting from state s. We embed the critic in a Gaussian model

$$Y \mid s \sim \mathcal{N}\big(V_\phi(s), \sigma^2(s)\big), \tag{4}$$

where $V_\phi(s)$ is the critic output and $\sigma^2(s) > 0$ is the conditional variance. This is a modeling choice: the critic output is treated as the mean of a distribution over returns, and the variance captures the intrinsic randomness of future returns from s.

We measure how sharply the log-likelihood of this model changes under small state perturbations using the Fisher information matrix (FIM) (Fisher, 1922; Yang, 2008)

$$\mathcal{I}(s) \coloneqq \mathop{\mathbb{E}}_{Y\sim p(\cdot|s)}\Big[\big(\nabla_s \log p(Y \mid s)\big)\big(\nabla_s \log p(Y \mid s)\big)^\top\Big]. \tag{5}$$

Classically, the Fisher information is defined with respect to the parameters of a statistical model. Here we differentiate with respect to the state, treating s as a pseudo-parameter of the observation model. The result is an input-space sensitivity: $\mathcal{I}(s)$ measures how much the return distribution responds to small changes in s. We use its trace as a scalar signal for the actor update.

The motivation is a local first-order argument. A perturbation δs changes the critic output by $\Delta V \approx \nabla_s V_\phi(s)^\top \delta s$, so $|\Delta V| \le \|\nabla_s V_\phi(s)\|\,\|\delta s\|$: the same state noise induces a larger error in the value estimate exactly where the trace is large, and the advantage computed on top of that value inherits this error. Down-weighting the directed surrogate at such states limits the influence of these noise-amplified samples; the complementary entropy share keeps their contribution from being zero. This is a heuristic argument, not a global guarantee. We also note the classical reading of Fisher information points the other way—high-information samples would be emphasized. In the on-policy setting, however, the advantage is bootstrapped from the critic itself, so a high trace is more about noise amplification than useful information. The ablations test this choice empirically.

4.2 Fisher Trace for a Heteroscedastic Gaussian Critic

Theorem 1 (Fisher trace for Gaussian critic)**.** *For the model $Y \mid s \sim \mathcal{N}(V_\phi(s), \sigma^2(s))$ with $\sigma^2(s) > 0$,*

$$\operatorname{tr}\big(\mathcal{I}(s)\big) = \frac{\|\nabla_s V_\phi(s)\|^2}{\sigma^2(s)} + \frac{\|\nabla_s \sigma^2(s)\|^2}{2\sigma^4(s)}. \tag{6}$$

The two terms measure sensitivity of the critic mean and variance to the state. A full proof is given in Appendix A.

4.3 Homoscedastic Variance in Standard PPO

Standard PPO critics output a single scalar and are trained with the squared-error loss $L^{\mathrm{VF}}(\phi) = \mathbb{E}_t[(V_\phi(s_t) - V_t^{\mathrm{target}})^2]$ (Schulman et al., 2017). Minimizing this loss is equivalent to maximum-likelihood estimation under additive homoscedastic Gaussian noise (Abramson et al., 2006). We adopt the homoscedastic model

$$\sigma^2(s) \equiv \sigma^2 \tag{7}$$

because it is the simplest case under which the Fisher trace is computable without a variance head. Heteroscedastic regression would require a separate variance prediction.

Under this assumption the variance term in Theorem 1 vanishes, since a constant has zero gradient:

$$\nabla_s \sigma^2(s) = \nabla_s \sigma^2 = \mathbf{0}. \tag{8}$$

Corollary 1 (Fisher trace under homoscedastic Gaussian critic)**.** *Under Eq.* (7)*,*

$$\operatorname{tr}\big(\mathcal{I}(s)\big) = \frac{\|\nabla_s V_\phi(s)\|^2}{\sigma^2}. \tag{9}$$

4.4 BATCH-NORMALIZED SENSITIVITY

Corollary 1 gives the raw sensitivity $\nu_i^{\mathrm{FIM}} := \|\nabla_s V_\phi(s_i)\|^2/\sigma^2$. For a batch of N states (in our implementation, the full rollout buffer of one update), dividing by the batch mean $\bar{\nu}^{\mathrm{FIM}}$ yields

$$\frac{\nu_i^{\mathrm{FIM}}}{\bar{\nu}^{\mathrm{FIM}} + \delta_{\mathrm{stab}}} = \frac{\|\nabla_s V_\phi(s_i)\|^2}{\frac{1}{N}\sum_{j=1}^{N} \|\nabla_s V_\phi(s_j)\|^2 + \sigma^2 \delta_{\mathrm{stab}}}, \tag{10}$$

where $\delta_{\mathrm{stab}} > 0$ is a small numerical-stability constant (we use δ rather than ε to avoid confusion with the PPO clip coefficient ϵ). Because σ^2 is a shared positive constant, it cancels between numerator and denominator up to the δ_{stab} term, so the normalized sensitivity can be computed without knowing σ^2. We therefore define the raw sensitivity as

$$\nu_i := \|\nabla_s V_\phi(s_i)\|^2 \tag{11}$$

and the normalized sensitivity as

$$\tilde{\nu}_i := \frac{\nu_i}{\bar{\nu} + \delta_{\mathrm{stab}}} = \frac{\|\nabla_s V_\phi(s_i)\|^2}{\frac{1}{N}\sum_{j=1}^{N} \|\nabla_s V_\phi(s_j)\|^2 + \delta_{\mathrm{stab}}}. \tag{12}$$

This operation normalizes a batch statistic; it is analogous to batch normalization (Ioffe & Szegedy, 2015) in spirit but adds no normalization layer and needs no variance head. The weights depend only on the relative sensitivity of the critic across the batch.

4.5 SAMPLE WEIGHT GENERATION

The normalized sensitivity is converted to a sample weight by a decreasing mapping, so that higher sensitivity leads to a smaller directed weight. We use a normalized softmax mapping. With scaling $\lambda \geq 0$ and temperature $\tau > 0$,

$$\begin{aligned} z_i &:= -\lambda \tilde{\nu}_i / \tau, \\ \hat{w}_i &:= N \cdot \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}, \\ w_i &:= \max\{\hat{w}_i, w_{\min}\}, \end{aligned} \tag{13}$$

where $w_{\min} \geq 0$ is a lower-bound hyperparameter. The factor N makes the weights average to one over the batch,

$$\frac{1}{N}\sum_{i=1}^{N} \hat{w}_i = 1, \tag{14}$$

so the directed term keeps the same overall scale as in standard PPO. The softmax concentrates the weight differences on the high-sensitivity tail; the floor $w_{\min}$ ensures every sample retains a small directed component; and τ and λ control the sharpness and scaling of the decay.

A natural alternative is the standard reciprocal mapping

$$\hat{w}_i^{\mathrm{std}} := \frac{1}{\left(1 + \lambda \tilde{\nu}_i\right)^{1/\tau}}, \tag{15}$$

which is bounded in $(0, 1]$ but does *not* average to one. A unit-mean reciprocal mapping can be obtained by rescaling,

$$\hat{w}_i^{\mathrm{ada}} := w_{\min} + (1 - w_{\min}) \frac{\hat{w}_i^{\mathrm{std}}}{\frac{1}{N}\sum_{j=1}^{N} \hat{w}_j^{\mathrm{std}}}, \tag{16}$$

so the average sample (with $\tilde{\nu}_i \approx 1$) maps to a weight near one. Both alternatives require explicit lower bounds and, for the standard mapping, a separate rescaling step to recover the PPO scale. SRN enforces the mean-one property exactly by construction and keeps every weight in the finite interval $[w_{\min}, N)$ after flooring. A formal comparison of these mappings, including the boundedness argument, is given in Appendix B.

The exploration side of the objective uses the complementary clipped weight

$$h_i := \mathrm{clip}\big(w_{\max} - w_i,\ 0,\ w_{\max}\big), \tag{17}$$

where $w_{\max} \geq w_{\min}$ is an upper-bound hyperparameter. Samples whose directed weight is suppressed below $w_{\max}$ get a larger exploration share, and samples with $w_i \geq w_{\max}$ get none.

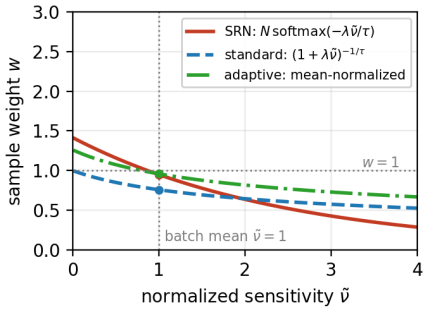


Figure 2: Three decreasing mappings from the batch-normalized sensitivity $\tilde{\nu}$ to the sample weight w on a reference batch. SRN averages exactly to one by construction; the adaptive mapping rescales to the same property; the standard reciprocal mapping does not.

Proposition 1. *Both weights are non-negative and bounded: for every sample i, $w_{\min} \le w_i < N$ and $0 \le h_i \le w_{\max}$, with $w_i + h_i = \max\{w_i, w_{\max}\}$. Consequently, samples with $w_i \le w_{\max}$ split their total weight evenly between the two loss terms ($w_i + h_i = w_{\max}$), and samples with $w_i > w_{\max}$ are dominated by the directed surrogate ($h_i = 0$).*

Proof. The softmax probabilities satisfy $0 < p_i < 1$, hence $0 < \hat{w}_i < N$. Flooring gives $w_{\min} \le w_i < N$. The claim for h_i follows directly from the definition of the clip operation and the bound on w_i. □

Remark (discrete state spaces). The gradient $\nabla_s V_\phi(s)$ assumes a continuous state. For the discrete tasks in our benchmark the observations are one-hot encoded, and the gradient is taken with respect to this embedding.

4.6 Stochastic Mixing Objective

The per-sample objective mixes the directed PPO surrogate with an entropy bonus:

$$J_i(\theta) := w_i\, \ell_i^{\mathrm{PPO}}(\theta) + h_i\, \beta_{\mathrm{ent}} \mathcal{H}_i(\theta), \tag{18}$$

where $\ell_i^{\mathrm{PPO}}(\theta)$ is the clipped PPO surrogate from Eq. (1), $\mathcal{H}_i(\theta)$ is the policy entropy at state s_i, w_i is the directed weight from Eq. (13), h_i is the clipped exploration weight from Eq. (17), and $\beta_{\mathrm{ent}} \ge 0$ is the entropy coefficient. The actor loss is the negative average over a minibatch of M samples:

$$\mathcal{L}^{\mathrm{actor}}(\theta) := -\frac{1}{M} \sum_{i=1}^{M} J_i(\theta). \tag{19}$$

When the critic is smooth at s_i, $\tilde{\nu}_i$ is small, w_i is large, and the update follows the PPO surrogate with little or no entropy. When the critic is sensitive, w_i is suppressed toward $w_{\min}$ and the update shifts toward exploration through h_i. Because the normalized weights average to one (Eq. (14)), the mapping redistributes the directed update across the batch rather than rescaling it. Since w_i and h_i are non-negative and bounded, $J_i(\theta)$ is a bounded non-negative combination of the two components and the PPO clip structure is preserved.

PPO performance is known to be sensitive to the entropy coefficient (Schulman et al., 2017; Engstrom et al., 2020). Here the sensitivity score $\tilde{\nu}_i$ adjusts the effective coefficient per sample: where the critic is smooth the directed surrogate dominates, and where it is steep the update leans more on exploration. Unlike soft actor-critic (Haarnoja et al., 2018), which uses a single global entropy temperature, GFIM-PPO varies the coefficient through the sample weights.

Remark (state-independent policy standard deviation). In our continuous-control implementation the Gaussian policy standard deviation is state-independent, so $\mathcal{H}_i(\theta)$ does not depend on the

Algorithm 1 GFIM-PPO update

Input: policy π_θ, critic V_ϕ, rollout buffer $\mathcal{D}$ of N samples, hyperparameters $\epsilon, \lambda, \tau, \beta_{\text{ent}}, w_{\min}, w_{\max}, \delta_{\text{stab}}, K$.
Compute advantage estimates $\{A_i\}$ and target values $\{V_i^{\text{target}}\}$ from $\mathcal{D}$ (e.g., GAE).
Compute state gradients $g_i \coloneqq \nabla_s V_\phi(s_i)$ and raw sensitivities ν_i by Eq. (11) for all $i \in \mathcal{D}$.
Compute normalized sensitivities $\tilde{\nu}_i$ by Eq. (12), SRN weights w_i by Eq. (13), and exploration weights h_i by Eq. (17), for all $i \in \mathcal{D}$.
for update epoch $= 1, \ldots, K$ **do**
  for each minibatch $\mathcal{B} \subseteq \mathcal{D}$ **do**
    Compute ratios $r_i(\theta)$ by Eq. (2), PPO surrogates $\ell_i^{\text{PPO}}(\theta)$ by Eq. (1), and entropies $\mathcal{H}_i(\theta)$ for all $i \in \mathcal{B}$.
    Form $J_i(\theta)$ by Eq. (18) with the fixed weights w_i, h_i.
    Update θ by ascending $\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} J_i(\theta)$.
    Update ϕ by minimizing the critic MSE in Eq. (3).
  end for
end for

state within an update. The mixing then modulates the *effective entropy coefficient* $\beta_{\text{ent}} \cdot \frac{1}{M} \sum_i h_i$ rather than per-state exploration. For discrete policies the entropy is state-dependent and the mixing acts at the state level directly.

4.7 Algorithm

Algorithm 1 summarizes the update. The sensitivity is the squared ℓ_2-norm of the critic state-gradient, which equals the FIM trace under the homoscedastic model. The weights are computed once per update over the full rollout buffer and held fixed across the K epochs of minibatch updates.

5 Experiments

We evaluate GFIM-PPO on a benchmark of 14 Gymnasium (Towers et al., 2024; Brockman et al., 2016) tasks in three categories: six classic-control tasks (80,000 environment steps), four small MuJoCo (Todorov et al., 2012) tasks (300,000 steps), and four large MuJoCo tasks (1,000,000 steps). The MuJoCo *small* and *large* labels reflect the training-step budget (300k vs. 1M) rather than the state dimension; for instance, `Hopper-v4` shares its observation dimension with several small tasks but is grouped with the large-budget locomotion tasks because it needs the longer horizon to converge. Every run uses 6 seeds and evaluates 40 episodes at each of 4 evenly spaced checkpoints. The ablation studies use a representative subset of five tasks (`Pendulum-v1`, `Hopper-v4`, `HalfCheetah-v4`, `BipedalWalker-v3`, `LunarLanderContinuous-v3`) covering all three categories.

5.1 Setup

All experiments use the same in-house PPO implementation; the baseline are the same code with uniform sample weights $w_i \equiv 1$. GFIM-PPO uses the batch-normalized Fisher trace $\|\nabla_s V_\phi(s)\|^2$ as the per-sample sensitivity, the normalized weight mapping of Eq. (13) with $\lambda = 1$ and $\tau = 2.5$, and the clipped exploration weight of Eq. (17) with $w_{\min} = 0.01$ and $w_{\max} = 1$. We compare three methods: the PPO baseline, PPO-TD (a weighting baseline that scales the surrogate by the normalized temporal-difference error, in the spirit of TD-error prioritization (Schaul et al., 2015)), and the full GFIM-PPO. Further variants that disable or restrict single modules of GFIM-PPO are studied as ablations in the appendix.

All methods share every other hyperparameter: Adam (Kingma & Ba, 2015) with learning rate 10^{-3}, discount factor $\gamma = 0.99$, GAE parameter 0.9, clip coefficient $\epsilon = 0.1$, entropy coefficient 0.02, value-loss coefficient 0.5, and two hidden layers of width 128. A PPO update is triggered every 256 collected steps and runs 10 epochs over the whole rollout buffer with minibatches of size 256. Since code-level choices can influence PPO as much as algorithmic ones (Engstrom et al., 2020;

Table 1: Final performance (mean ± SEM over 6 seeds, 40 evaluation episodes at the final checkpoint) on 14 environments. **Bold** marks the best mean, underlined the runner-up; † marks environments where GFIM-PPO improves over PPO significantly after Holm correction ($p < 0.05$, Welch's t-test).

Environment	PPO	PPO-TD	GFIM-PPO
Classic control (80k steps)			
`Acrobot-v1`	$\mathbf{-107.2 \pm 5.5}$	-121.0 ± 4.9	-153.4 ± 69.3
`CliffWalking-v1`	-859.0 ± 142.0	$\mathbf{-758.3 \pm 184.3}$	-835.5 ± 164.5
`Pendulum-v1`	-389.9 ± 84.0	$\mathbf{-387.0 \pm 65.0}$	-658.6 ± 91.9
`MountainCarContinuous-v0`	-0.97 ± 0.15	$\mathbf{-0.89 \pm 0.14}$	-0.90 ± 0.16
`BipedalWalker-v3`	-99.15 ± 6.91	-110.0 ± 7.7	$\mathbf{-72.68 \pm 10.54}$
`LunarLanderContinuous-v3`	-70.85 ± 6.51	-97.86 ± 12.18	$\mathbf{-68.93 \pm 22.91}$
MuJoCo big (1M steps)			
`Hopper-v4`	404.5 ± 24.5	146.2 ± 21.3	$\mathbf{417.1 \pm 78.0}$
`Walker2d-v4`†	23.24 ± 6.76	3.63 ± 1.01	$\mathbf{476.8 \pm 65.6}$
`HalfCheetah-v4`†	681.9 ± 93.4	-176.3 ± 120.0	$\mathbf{2108.6 \pm 383.0}$
`Ant-v4`†	156.7 ± 18.3	41.66 ± 19.61	$\mathbf{1849.8 \pm 237.1}$
MuJoCo small (300k steps)			
`InvertedPendulum-v4`†	749.3 ± 35.0	681.2 ± 39.6	$\mathbf{911.1 \pm 30.5}$
`InvertedDoublePendulum-v4`	7214.5 ± 290.5	6842.9 ± 538.3	$\mathbf{8182.5 \pm 391.6}$
`Reacher-v4`†	-32.28 ± 1.51	-26.06 ± 0.81	$\mathbf{-21.25 \pm 0.48}$
`Swimmer-v4`	33.37 ± 1.05	32.30 ± 1.78	$\mathbf{34.64 \pm 1.04}$

Andrychowicz et al., 2021), all methods use the same codebase, update schedule, and hyperparameters, and differ only on the per-sample weighting.

Discrete observations are one-hot encoded, and the state gradient $\nabla_s V_\phi(s)$ is taken with respect to this embedding. On the continuous tasks the policy is a Gaussian whose standard deviation is a state-independent parameter.

5.2 GFIM-PPO vs Baselines

We evaluate all three methods on the 14-task benchmark. Because reward scales differ across tasks and no return normalization is applied, we compare methods by counting the tasks on which one improves over the other rather than by averaging returns. Table 1 reports the final-checkpoint performance of every method on every task, together with the within-task significance of GFIM-PPO over the PPO baseline. Detailed percentage improvements and the exact formula used to compute them are given in Appendix L.

Classic control. Among the six classic tasks, GFIM-PPO improves over PPO on four. The two exceptions are `Acrobot-v1` and `Pendulum-v1`; both gaps are non-significant ($p = 0.536$ and $p = 0.056$), and the Acrobot gap is driven by higher seed-level variance (SEM 69.3 vs. 5.5 for PPO). Two classic tasks are null results on which no method learns (`MountainCarContinuous-v0`, `CliffWalking-v1`).

MuJoCo. GFIM-PPO improves over PPO on all eight MuJoCo tasks. The improvements are pronounced on the four large MuJoCo tasks and also present on the four small MuJoCo tasks. Five MuJoCo tasks pass a Holm-corrected Welch t-test within each task ($n = 6$ seeds): `Walker2d-v4`, `HalfCheetah-v4`, `Ant-v4`, `InvertedPendulum-v4`, and `Reacher-v4`. With 6 seeds these tests have limited power (Agarwal et al., 2021), so the p-values should be read as exploratory.

The weighting baseline. Against PPO-TD, GFIM-PPO is ahead on 10 of 14 tasks, and PPO-TD significantly *degrades* performance on four tasks after the Holm correction, which indicates that per-sample weighting by itself is not enough; the structure of the weighting signal matters. GFIM-PPO is the best of the three methods on 10 of 14 tasks and among the top two on 12.

The smoothed training-return curves of the three methods are shown in Appendix N (Figure 6). GFIM-PPO leads at every checkpoint on seven tasks, so the final gaps are not endpoint artifacts. On `Walker2d-v4` the PPO baseline degrades late in training ($106.6 \rightarrow 23.2$ over the last two

checkpoints) while GFIM-PPO remains stable. The per-checkpoint evaluation curves are given in the appendix (Figure 7). We summarize the entropy, weight-mapping, and sensitivity-estimator ablations and the robustness, sensitivity, and overhead studies in Appendices E–K.

6 Conclusion

We introduced a state-level sensitivity measure derived from the Fisher information of a Gaussian critic model, and used it to modulate the trade-off between directed policy updates and exploratory entropy in actor-critic learning. Under a homoscedastic variance assumption the Fisher trace reduces to the squared norm of the critic state-gradient, which can be computed with a single additional backward pass and normalized batch-wise without knowing the return variance. The resulting algorithm assigns larger directed weights where the critic is smooth and larger entropy shares where it is sensitive, preserving the conservative structure of the underlying policy-gradient update while adapting the exploration pressure to each sample.

The method improves over the baseline on a range of continuous-control tasks and carries only modest computational overhead. More importantly, the design points toward a broader family of actor-critic algorithms in which critic sensitivity can be used to reweight policy gradients. Future work includes extending the idea to other actor-critic methods, introducing a state-dependent variance head to recover the heteroscedastic term of the Fisher trace without losing the modest computational cost, and combining the Fisher trace with model-based uncertainty estimates to handle partially observed or non-stationary environments.

AI Use Statement

In this work, we used generative AI tools to aid and polish the writing and for retrieval and discovery of related work. Concretely, AI tools were used to improve the clarity and grammar of author-written text and to help locate potentially relevant references. We did not use generative AI tools to generate synthetic datasets, to prove mathematical claims, or to draft whole sections of the paper. All AI-assisted text and every AI-suggested reference were reviewed and verified by the authors against the original sources, all proofs and derivations were checked by hand, and the authors take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

Reproducibility Statement

We provide the details needed to reproduce our results in the main text and the appendix. All experiments use a single in-house PPO implementation and a shared set of hyperparameters, which are stated in Section 5.1. Each environment is run with 6 seeds and evaluated at four evenly spaced checkpoints with 40 episodes per checkpoint. Full proofs of the Fisher-trace and bounded-mixing results are given in Appendices A and D, the Gaussian critic model and the normalized weight mapping are detailed in Appendix B, and the complete design-space ablations (entropy term, weight mapping and cap, and sensitivity-estimator variants) are reported in Appendices F–H. The computational overhead of the method is quantified in Appendix K. An anonymized implementation is available at https://anonymous.4open.science/r/CAAC-F11F.

References

N. Abramson, D. J. Braverman, and G. S. Sebestyen. Pattern recognition and machine learning. *Publications of the American Statistical Association*, 103(4):886–887, 2006.

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

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

Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphael Marinier, Léonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, et al. What matters in on-policy reinforcement learning? a large-scale empirical study. In *ICLR 2021-ninth international conference on learning representations*, 2021.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pp. 449–458. Pmlr, 2017.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016. URL https://arxiv.org/abs/1606.01540.

Ju-Seung Byun and Andrew Perrault. Normality-guided distributional reinforcement learning for continuous control. *arXiv preprint arXiv:2208.13125*, 2022.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. Reasoning with exploration: An entropy perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30377–30385, 2026.

Wojciech Marian Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Swirszcz, and Razvan Pascanu. Sobolev training for neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 4281–4290, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Will Dabney, Mark Rowland, Marc Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

H Drucker and Y Le Cun. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):P.991–997, 1992.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on ppo and trpo [j]. *arXiv preprint arXiv:2005.12729*, 2020.

Ronald A Fisher. On the mathematical foundations of theoretical statistics. *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, 222(594-604):309–368, 1922.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870, 2018.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 448–456, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/ioffe15.html.

Sham M Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14, 2001.

Dongyoung Kim, Jinwoo Shin, Pieter Abbeel, and Younggyo Seo. Accelerating reinforcement learning with value-conditional state entropy exploration. *Advances in Neural Information Processing Systems*, 36:31811–31830, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015. URL https://mlanthology.org/iclr/2015/kingma2015iclr-adam/.

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

Mohammadreza Nakhaeinezhadfard, Aidan Scannell, and Joni Pajarinen. Entropy regularized task representation learning for offline meta-reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 19616–19623, 2025.

Michal Nauman and Marek Cygan. Decoupled policy actor-critic: Bridging pessimism and risk awareness in reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 19633–19641, 2025.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. *IEEE*, 2017.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 833–840, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

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

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

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

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

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2 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.

Richard S Sutton, A Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research*, 17(73):1–29, 2016.

Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. Deep reinforcement learning for robotics: A survey of real-world successes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 28694–28698, 2025. doi: 10.1609/aaai.v39i27.35095. URL https://doi.org/10.1609/aaai.v39i27.35095.

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

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

Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *Advances in neural information processing systems*, 30, 2017.

Yuhong Yang. Elements of information theory, 2008.

A Proof of Theorem 1

For $Y \mid s \sim \mathcal{N}(V_\phi(s), \sigma^2(s))$ with $\sigma^2(s) > 0$,

$$\log p(y \mid s) = -\frac{1}{2}\log(2\pi) - \frac{1}{2}\log\sigma^2(s) - \frac{\big(y - V_\phi(s)\big)^2}{2\sigma^2(s)}. \tag{20}$$

Differentiating with respect to s,

$$\nabla_s \log p(y \mid s) = \frac{y - V_\phi(s)}{\sigma^2(s)}\nabla_s V_\phi(s) + \left(\frac{\big(y - V_\phi(s)\big)^2}{2\sigma^4(s)} - \frac{1}{2\sigma^2(s)}\right)\nabla_s\sigma^2(s) \tag{21}$$

$$= \frac{y - V_\phi(s)}{\sigma^2(s)}\nabla_s V_\phi(s) + \frac{\big(y - V_\phi(s)\big)^2 - \sigma^2(s)}{2\sigma^4(s)}\nabla_s\sigma^2(s). \tag{22}$$

Hence

$$\begin{aligned}
\mathcal{I}(s) = \mathbb{E}\Big[&\Big(\frac{Y - V_\phi(s)}{\sigma^2(s)}\nabla_s V_\phi(s) + \frac{(Y - V_\phi(s))^2 - \sigma^2(s)}{2\sigma^4(s)}\nabla_s\sigma^2(s)\Big) \\
&\times \Big(\frac{Y - V_\phi(s)}{\sigma^2(s)}\nabla_s V_\phi(s) + \frac{(Y - V_\phi(s))^2 - \sigma^2(s)}{2\sigma^4(s)}\nabla_s\sigma^2(s)\Big)^\top\Big] \qquad (23)\\
= \mathbb{E}\Big[&\frac{(Y - V_\phi(s))^2}{\sigma^4(s)}\nabla_s V_\phi(s)\nabla_s V_\phi(s)^\top\Big] \\
&+ \mathbb{E}\Big[\frac{(Y - V_\phi(s))\big((Y - V_\phi(s))^2 - \sigma^2(s)\big)}{2\sigma^6(s)}\big(\nabla_s V_\phi(s)\nabla_s\sigma^2(s)^\top + \nabla_s\sigma^2(s)\nabla_s V_\phi(s)^\top\big)\Big] \\
&+ \mathbb{E}\Big[\frac{\big((Y - V_\phi(s))^2 - \sigma^2(s)\big)^2}{4\sigma^8(s)}\nabla_s\sigma^2(s)\nabla_s\sigma^2(s)^\top\Big] \qquad (24)\\
= &\frac{\mathbb{E}[(Y - V_\phi(s))^2]}{\sigma^4(s)}\nabla_s V_\phi(s)\nabla_s V_\phi(s)^\top \\
&+ \frac{\mathbb{E}[(Y - V_\phi(s))^3] - \sigma^2(s)\,\mathbb{E}[Y - V_\phi(s)]}{2\sigma^6(s)}\big(\nabla_s V_\phi(s)\nabla_s\sigma^2(s)^\top + \nabla_s\sigma^2(s)\nabla_s V_\phi(s)^\top\big) \\
&+ \frac{\mathbb{E}[\big((Y - V_\phi(s))^2 - \sigma^2(s)\big)^2]}{4\sigma^8(s)}\nabla_s\sigma^2(s)\nabla_s\sigma^2(s)^\top \qquad (25)\\
= &\frac{1}{\sigma^2(s)}\nabla_s V_\phi(s)\nabla_s V_\phi(s)^\top + \frac{1}{2\sigma^4(s)}\nabla_s\sigma^2(s)\nabla_s\sigma^2(s)^\top, \qquad (26)
\end{aligned}$$

where the last equality uses $Y - V_\phi(s) \mid s \sim \mathcal{N}(0, \sigma^2(s))$, whose odd-order central moments vanish and whose second and fourth central moments satisfy $\mathbb{E}[(Y - V_\phi(s))^2 \mid s] = \sigma^2(s)$ and $\mathbb{E}[\big((Y - V_\phi(s))^2 - \sigma^2(s)\big)^2 \mid s] = 2\sigma^4(s)$. Taking the trace and using $\operatorname{tr}(gg^\top) = \|g\|^2$ completes the proof:

$$\operatorname{tr}\mathcal{I}(s) = \frac{\|\nabla_s V_\phi(s)\|^2}{\sigma^2(s)} + \frac{\|\nabla_s\sigma^2(s)\|^2}{2\sigma^4(s)}. \tag{27}$$

B Weight Mapping Details

This appendix collects the weight-mapping formulas used in the method and ablation studies. Recall the batch-normalized sensitivity $\tilde{\nu}_i$ from Eq. (12).

Softmax-renormalized mapping (SRN). The SRN mapping of Eq. (13) is

$$z_i = -\frac{\lambda\tilde{\nu}_i}{\tau}, \qquad \hat{w}_i = N\,\frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}, \qquad w_i = \max\{\hat{w}_i, w_{\min}\}. \tag{28}$$

Because the softmax probabilities $p_i = e^{z_i}/\sum_j e^{z_j}$ satisfy $0 < p_i < 1$ and $\sum_i p_i = 1$, the unfloored weights obey $0 < \hat{w}_i < N$ and average to one. After flooring, $w_{\min} \le w_i < N$ and the mean is at least one.

Standard reciprocal mapping. The standard reciprocal mapping is

$$w_i^{\mathrm{std}} = \frac{1}{\left(1 + \lambda \tilde{\nu}_i\right)^{1/\tau}}, \tag{29}$$

clipped to $[w_{\min}, 1]$. Each weight is bounded, but the batch average is generally smaller than one, so the directed update is shrunk unless an additional rescaling is applied.

Adaptive mean-normalized mapping. The adaptive mapping first computes the standard reciprocal weights and then rescales them to unit mean:

$$\hat{w}_i^{\mathrm{ada}} = w_{\min} + (1 - w_{\min}) \, \frac{w_i^{\mathrm{std}}}{\frac{1}{N}\sum_{j=1}^{N} w_j^{\mathrm{std}}}. \tag{30}$$

This enforces the same mean-one property as SRN numerically, but the weights are no longer bounded above by a constant independent of the batch; in practice they remain finite because the batch is finite.

SRN is therefore the default: it enforces mean-one exactly, requires only a lower bound, and keeps every weight in the finite interval $[w_{\min}, N)$.

C Theoretical Properties

Since w_i and h_i are non-negative and bounded, $J_i(\theta)$ lies within the interval spanned by its two components, scaled by $w_i + h_i = \max\{w_i, w_{\max}\}$. For samples with $w_i \le w_{\max}$ the weights sum to $w_{\max}$; when $w_{\max} = 1$ this is a convex mixture of the directed PPO surrogate and the entropy bonus, so no sample contributes more than the larger of its two components. For samples with $w_i > w_{\max}$ the objective reduces to the directed surrogate scaled by w_i, and the entropy share is zero. In either case

$$|J_i(\theta)| \le \max\{w_i, w_{\max}\} \, \max\left\{|\ell_i^{\mathrm{PPO}}|, |\beta_{\mathrm{ent}} \mathcal{H}_i|\right\}, \tag{31}$$

which has two practical consequences. First, the modification does not amplify any sample beyond the PPO surrogate or the entropy term; the per-sample update magnitude is controlled by the same trust-region-like quantities that standard PPO clips. Second, the entropy contribution is capped by $w_{\max}$, so the exploration term cannot overwhelm the directed term even when the critic is very sensitive. A formal statement and proof are given in Appendix D.

D Proof of Theorem 31

Fix a sample i and write $D \coloneqq D_i(\theta)$, $R \coloneqq R_i(\theta)$, $m \coloneqq \min\{D, R\}$, $M \coloneqq \max\{D, R\}$. By Proposition 1, $w_i \ge 0$ and $h_i \ge 0$, so

$$J_i(\theta) = w_i D + h_i R \;\ge\; w_i m + h_i m = (w_i + h_i)\, m, \tag{32}$$

and analogously $J_i(\theta) \le (w_i + h_i)\, M$. Proposition 1 also gives $w_i + h_i = \max\{w_i, w_{\max}\}$, which yields the first claim. For the second claim,

$$\begin{aligned} |J_i(\theta)| &\le w_i |D| + h_i |R| \\ &\le (w_i + h_i) \max\{|D|, |R|\} \\ &= \max\{w_i, w_{\max}\} \, \max\{|D|, |R|\}, \end{aligned} \tag{33}$$

which completes the proof.

E Ablation Overview

We summarize the ablations here; full tables and figures are in Appendices F–K. Each variant disables or restricts exactly one module of GFIM-PPO. All ablations use the five representative tasks and the final checkpoint, with 6 seeds per configuration.

The entropy term is environment-dependent by design. Our motivation is that the value of an entropy bonus depends on the environment: where the value signal is unstable or keeps shifting during training, sustained exploration helps the policy track it; where the value landscape is already smooth and stationary, extra entropy is unnecessary. The ablations support this reading rather than a blanket verdict for or against entropy. Removing the uniform entropy bonus from PPO helps on three of the five tasks and hurts on two, and a fixed entropy share inside GFIM (GFIM-const-ent) beats the no-entropy variant on only two of the five, so a static choice either way is a per-environment gamble. The sensitivity-modulated share is what makes entropy reliable: the full method beats both its no-entropy and its fixed-entropy counterparts on four of the five tasks, the directed weighting alone (GFIM-noent vs. PPO-noent) and the weighted entropy alone (GFIM-onlyent vs. PPO) each also help on four of the five, and no single component wins everywhere; `Pendulum-v1` prefers no entropy at all, which is exactly why the method modulates the entropy share per sample instead of fixing it per task. Full results are in Appendix F.

The combination. The full method is rarely the per-task optimum on the ablation tasks (GFIM-onlyent is ahead on 4 of 5), but it is the most consistent across the benchmark: over all 14 tasks GFIM-PPO is the best method on 10 and among the top two on 12, whereas each ablated variant trades one regime for another. We therefore keep the combination: it is the only configuration that never falls far behind on any task family.

Weight mapping. We compare SRN against a standard reciprocal mapping and an adaptive (mean-normalized reciprocal) mapping (Table 3). SRN is ahead of the standard mapping on 4 of 5 tasks; the adaptive mapping is stronger on `Pendulum-v1` and the two large MuJoCo tasks, while SRN is stronger on the two classic continuous tasks, mostly within one standard error. We keep SRN because its weights average to one by construction; the adaptive mapping enforces the same property numerically and is a viable alternative. See Appendix G.

Other design choices. The default cap $w_{\max} = 1$ is the best tested value on 3 of 5 tasks, and raising it sharply degrades `HalfCheetah-v4` (Table 4; Appendix G). The four sensitivity estimators perform similarly after batch normalization, with the default g^2 and g^3 each best on 2 of 5 tasks and most gaps within one standard error (Table 6; Appendix H). Under training-time observation noise, GFIM-PPO degrades less than PPO on `Hopper-v4` at every noise level and retains higher absolute returns on `HalfCheetah-v4` and `BipedalWalker-v3`, while the baseline degrades less on `Pendulum-v1` (Table 7; Section I); the benefit is again task-dependent. The default $(\tau, \lambda) = (2.5, 1)$ is not the per-task optimum; the best configuration varies by environment, but it stays within noise of the best on most tasks (Table 8; Appendix J). The overhead stays within 1% of the baseline throughput on 3 of 5 profiled tasks, with the largest reduction on `BipedalWalker-v3` (-13.8% steps per second; Table 9; Appendix K).

F Entropy Ablation

This section collects the entropy ablation studies summarized in the ablation overview.

Motivation

Each variant below disables or restricts exactly one module of GFIM-PPO. All ablations use the five representative tasks and the final checkpoint, with 6 seeds per configuration. The value of an entropy bonus is environment-dependent. In tasks where the value signal is unstable, noisy, or keeps shifting as training proceeds, so the policy must keep exploring to track it, a well-placed entropy share is useful; in near-stationary tasks whose value landscape is already smooth, extra entropy is unnecessary and can dilute the directed update. GFIM-PPO is therefore designed not to prescribe entropy globally, but to let the per-sample sensitivity decide where exploration is allocated.

Entropy and Its Modulation

The sensitivity-modulated entropy share is a central module of the method, and Table 2 isolates it along three axes at once: whether an entropy term is present at all (PPO vs. PPO-noent, and GFIM

Table 2: Ablation of the entropy term on five representative environments: removing the entropy bonus from PPO (PPO-noent) and from GFIM (GFIM-noent), applying the SRN weights to the surrogate with a uniform entropy term (GFIM-const-ent), and applying the Fisher-derived weights only to the entropy term while keeping the surrogate uniformly weighted (GFIM-onlyent). Mean $\pm$ SEM over 6 seeds, 40 evaluation episodes at the final checkpoint. **Bold** marks the best mean; underlined marks the runner-up.

Environment	PPO	PPO-noent	GFIM-noent	GFIM-const-ent	GFIM	GFIM-onlyent
`Pendulum-v1`	$\mathbf{-389.9 \pm 84.0}$	-709.0 ± 166.9	-566.7 ± 122.5	-504.5 ± 84.9	-658.6 ± 91.9	$\underline{-484.1 \pm 100.5}$
`Hopper-v4`	404.5 ± 24.5	$\underline{431.0 \pm 65.5}$	239.3 ± 17.1	360.4 ± 74.9	417.1 ± 78.0	$\mathbf{473.4 \pm 53.0}$
`HalfCheetah-v4`	681.9 ± 93.4	1747.0 ± 456.3	1886.5 ± 350.6	686.4 ± 125.0	$\underline{2108.6 \pm 383.0}$	$\mathbf{2632.2 \pm 507.4}$
`BipedalWalker-v3`	-99.15 ± 6.91	-87.52 ± 8.03	$\underline{-84.42 \pm 12.92}$	-109.0 ± 7.5	$\mathbf{-72.68 \pm 10.54}$	-96.36 ± 10.43
`LunarLanderContinuous-v3`	-70.85 ± 6.51	-78.27 ± 9.05	-73.60 ± 13.24	-96.79 ± 11.05	$\underline{-68.93 \pm 22.91}$	$\mathbf{-60.44 \pm 11.47}$

vs. GFIM-noent), whether the entropy share is fixed or sensitivity-modulated (GFIM-const-ent vs. GFIM), and what the directed weighting contributes on its own when entropy is removed entirely (GFIM-noent vs. PPO-noent).

The results support the design reading rather than a blanket verdict. Removing the uniform entropy bonus from PPO helps on three of the five tasks and hurts on two, so a fixed entropy bonus is a per-environment gamble. Inside GFIM the picture is clearer: the full method beats GFIM-noent on four of the five tasks, so once the directed weighting is in place, the modulated entropy share helps on most tasks. Between the two static alternatives there is no winner: GFIM-const-ent (fixed entropy share) beats GFIM-noent on only two of the five tasks, so we cannot say that adding a fixed entropy term is better than dropping entropy, nor the reverse. The modulated share, however, beats the fixed one on four of the five tasks (GFIM vs. GFIM-const-ent), which is direct evidence that *where* the entropy is placed matters more than whether it is present. The directed weighting also carries value on its own; GFIM-noent beats PPO-noent on four of the five tasks, but the gains brought by the modulated entropy share alone are at least as large: GFIM-onlyent, which applies the Fisher weights only to the entropy term, is ahead of the full method on four of the five tasks. Taken together, the entropy term should not be discarded, because it genuinely helps in several environments, but its benefit is reliable only when its placement is modulated by the sensitivity signal.

Exploration-side weighting alone. GFIM-onlyent computes the Fisher weights exactly as the full method but applies them only to the entropy term, leaving the surrogate uniformly weighted. It improves over the PPO baseline on four of the five tasks (trailing only on `Pendulum-v1`), which shows that the weighted entropy share alone is already useful in most of the representative tasks. On the other four tasks it is also ahead of the full method, with `BipedalWalker-v3` as the exception; over the whole 14-task benchmark, however, the full method is the most consistent (best on 10 tasks, top-two on 12), so we keep the combination rather than the exploration-only variant.

G Weight Mapping and Cap Ablation

Weight Mapping

We compare SRN against a standard reciprocal mapping and an adaptive (mean-normalized reciprocal) mapping (Table 3). SRN is ahead of the standard mapping on 4 of 5 tasks, so the softmax renormalization itself is a sound default. The adaptive mapping is stronger on `Pendulum-v1` and the two large MuJoCo tasks (`Hopper-v4`, `HalfCheetah-v4`), while SRN is stronger on the two classic continuous tasks; most differences lie within one standard error. We keep SRN because its weights average to one by construction; the adaptive mapping enforces the same property numerically and is a viable alternative.

Exploration-Weight Cap

Table 4 varies $w_{\max}$. The default $w_{\max} = 1$ is the best tested value on 3 of 5 tasks, and raising the cap sharply degrades `HalfCheetah-v4` ($2108.6 \to 1449.3 \to 891.3$ for $w_{\max} = 1.0/1.5/2.0$); `Hopper-v4` prefers 1.5 and `Pendulum-v1` prefers 2.0. We keep $w_{\max} = 1$ as a robust default.

Table 3: Comparison of gradient mappings on five representative environments: the SRN mapping used by GFIM versus standard and adaptive mappings; PPO is shown for reference. Mean ± SEM over 6 seeds, 40 evaluation episodes at the final checkpoint. **Bold** marks the best mean; underlined marks the runner-up.

Environment	PPO	GFIM (SRN)	GFIM-map-standard	GFIM-map-adaptive
`Pendulum-v1`	$\mathbf{-389.9 \pm 84.0}$	-658.6 ± 91.9	-411.7 ± 63.6	-451.0 ± 102.9
`Hopper-v4`	404.5 ± 24.5	417.1 ± 78.0	384.6 ± 50.3	$\mathbf{436.4 \pm 56.4}$
`HalfCheetah-v4`	681.9 ± 93.4	2108.6 ± 383.0	2011.9 ± 444.4	$\mathbf{2460.5 \pm 435.9}$
`BipedalWalker-v3`	-99.15 ± 6.91	$\mathbf{-72.68 \pm 10.54}$	-102.8 ± 11.4	-117.3 ± 10.5
`LunarLanderContinuous-v3`	-70.85 ± 6.51	$\mathbf{-68.93 \pm 22.91}$	-96.28 ± 20.55	-84.21 ± 14.17

Table 4: Sensitivity to the weight cap $w_{\max}$ on five representative environments; $w_{\max} = 1.0$ is the default GFIM configuration. Mean ± SEM over 6 seeds, 40 evaluation episodes at the final checkpoint. **Bold** marks the best mean; underlined marks the runner-up.

Environment	$w_{\max} = 1.0$ (GFIM)	$w_{\max} = 1.5$	$w_{\max} = 2.0$
`Pendulum-v1`	-658.6 ± 91.9	-542.2 ± 135.7	$\mathbf{-492.4 \pm 112.5}$
`Hopper-v4`	417.1 ± 78.0	$\mathbf{488.2 \pm 52.0}$	309.9 ± 38.1
`HalfCheetah-v4`	$\mathbf{2108.6 \pm 383.0}$	1449.3 ± 134.9	891.3 ± 167.6
`BipedalWalker-v3`	$\mathbf{-72.68 \pm 10.54}$	-82.08 ± 14.79	-116.4 ± 1.6
`LunarLanderContinuous-v3`	$\mathbf{-68.93 \pm 22.91}$	-85.43 ± 15.85	-81.19 ± 12.50

Table 5: Sensitivity variants used in this paper, with $g = \|\nabla_s V_\phi(s)\|$.

Variant	Sensitivity $\nu(s)$
standard (Fisher trace)	g^2
sub-linear	$g^{2.5}$
linear	g^3
logarithmic	$g^2 \ln(1+g)$

H Sensitivity Estimator Variants

Sensitivity Estimator Variants

Design of Sensitivity Variants

The Fisher trace reduces to g^2, the squared norm of the critic state-gradient. This raises a natural question: does the exact growth rate of the sensitivity with respect to g matter? We compare three alternatives that grow faster than g^2 in the high-gradient regime ($g^{2.5}$, g^3, and $g^2 \ln(1+g)$), alongside the default g^2 (Table 5). All four share the same pipeline: the sensitivity is batch-normalized and mapped to a weight by the SRN mapping, so the only difference is how aggressively the directed update is suppressed where the critic gradient is large. Because most samples in a typical batch have moderate gradient norms, batch normalization keeps the resulting weight curves close over the bulk of the range; the variants separate only in the sparsely populated high-gradient tail. The four variants are evaluated empirically in Appendix H.

Table 6 evaluates the four sensitivity estimators of Table 5 on the representative tasks. As with the weight mappings, we treat these as a study of common modulations rather than a search for a best estimator. No variant dominates: the default Fisher trace g^2 and the g^3 variant are each best on 2 of 5 tasks and the $g^2 \ln(1+g)$ variant on 1, but most pairwise differences lie within one standard error of the mean. This is consistent with the batch-normalization argument from the method section: after normalization the weight curves differ mainly in the extreme high-gradient tail, which is sparsely populated in a typical batch. We therefore keep the plain Fisher trace as the model-derived default and treat the variants as task-dependent refinements.

Table 6: Uncertainty-estimator variants on five representative environments; GFIM uses $u = g^2$ by default. Mean $\pm$ SEM over 6 seeds, 40 evaluation episodes at the final checkpoint. **Bold** marks the best mean; underlined marks the runner-up.

Environment	GFIM (g^2)	$g^{2.5}$	g^3	$g^2 \ln(1+g)$
`Pendulum-v1`	-658.6 ± 91.9	-642.2 ± 94.5	$\mathbf{-404.6 \pm 86.1}$	-660.8 ± 70.6
`Hopper-v4`	417.1 ± 78.0	334.2 ± 53.6	406.7 ± 52.9	$\mathbf{459.0 \pm 52.1}$
`HalfCheetah-v4`	2108.6 ± 383.0	2196.2 ± 400.3	$\mathbf{2484.0 \pm 503.4}$	2388.1 ± 390.0
`BipedalWalker-v3`	$\mathbf{-72.68 \pm 10.54}$	-91.55 ± 5.62	-109.1 ± 9.9	-94.71 ± 9.28
`LunarLanderContinuous-v3`	$\mathbf{-68.93 \pm 22.91}$	-81.71 ± 15.05	-79.98 ± 21.02	-73.37 ± 14.08

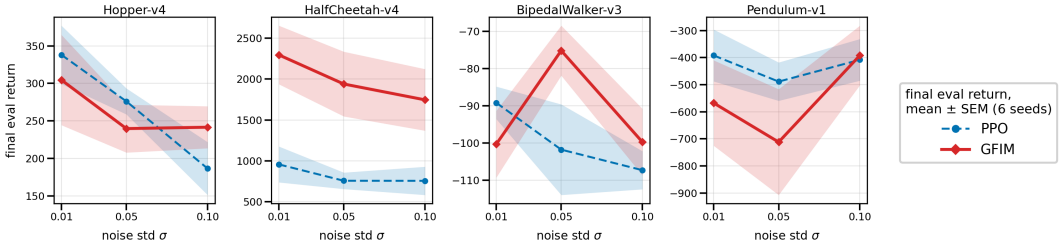


Figure 3: Final evaluation return under training-time observation noise of standard deviation σ (mean $\pm$ SEM over 6 seeds; shaded regions show the standard error of the mean) on four representative tasks. GFIM-PPO degrades less on `Hopper-v4` and holds higher absolute returns on `HalfCheetah-v4` and (at $\sigma \geq 0.05$) `BipedalWalker-v3`, while the baseline degrades less on `Pendulum-v1`.

I Robustness to Observation Noise

Training-time noise. We train both GFIM-PPO and the PPO baseline with zero-mean Gaussian observation noise of standard deviation $\sigma \in \{0.01, 0.05, 0.1\}$ on four of the representative tasks and measure the final performance. Noise is injected at every environment step during both training and evaluation. Table 7 and Figure 3 report the results; because the reference returns cross zero on the classic tasks, relative retention ratios are ill-conditioned there and we compare absolute returns, using retention only on the two large MuJoCo tasks whose returns are positive.

The outcome is task-dependent rather than uniformly in favor of either method. On `Hopper-v4` GFIM-PPO degrades less at every noise level (retention $0.89/0.71/0.63$ against $0.85/0.70/0.48$; at $\sigma = 0.1$ the absolute return is 241.6 vs. 186.5). On `HalfCheetah-v4` both methods tolerate the small noise level; their $\sigma = 0.01$ returns are at or above the noise-free ones, and GFIM-PPO keeps a large absolute margin throughout (1745.6 vs. 755.0 at $\sigma = 0.1$), even though its relative retention is lower than the baseline's, whose noise-free run is weak to begin with. On `BipedalWalker-v3` GFIM-PPO holds higher absolute returns at $\sigma = 0.05$ and $\sigma = 0.10$ (-75.2 vs. -101.7 and -99.7 vs. -107.3), while on `Pendulum-v1` the baseline degrades less in absolute terms.

This mixed picture is consistent with the sensitivity-based motivation: down-weighting high-sensitivity states limits the influence of perturbed observations where the critic landscape is steep, but it does not turn every task in GFIM-PPO's favor under noise.

J Hyperparameter Sensitivity

Hyperparameter Sensitivity

The weight mapping introduces two hyperparameters, the temperature τ and the scale λ. Before tuning anything we ask a design question: are these *algorithm-level* constants, for which one setting works everywhere, or *environment-sensitive* knobs whose best value changes with the task? The answer decides whether GFIM-PPO needs per-task tuning, which the method is explicitly designed to avoid. We therefore run a one-factor-at-a-time sweep of $\tau \in \{1.5, 2, 3\}$ and $\lambda \in \{0.5, 2, 4\}$ around the default $(\tau, \lambda) = (2.5, 1)$ on all five representative tasks (6 seeds each).

Table 7: Final performance under observation noise injected during training on four representative environments (mean ± SEM, 6 seeds), with retention relative to the noise-free run in parentheses. Retention ratios are ill-conditioned on tasks whose reference returns cross zero (`Pendulum-v1`, `BipedalWalker-v3`) and are therefore omitted.

Environment	Method	$\sigma = 0.01$	$\sigma = 0.05$	$\sigma = 0.10$
`Pendulum-v1`	GFIM	-567.7 ± 157.6	-711.9 ± 195.2	-392.0 ± 109.9
	PPO	-392.1 ± 95.7	-488.7 ± 70.7	-408.3 ± 76.9
`Hopper-v4`	GFIM	304.6 ± 60.1 (0.89)	239.7 ± 31.9 (0.71)	241.6 ± 28.0 (0.63)
	PPO	338.1 ± 38.9 (0.85)	276.1 ± 17.3 (0.70)	186.5 ± 35.2 (0.48)
`HalfCheetah-v4`	GFIM	2295.6 ± 358.1 (1.20)	1939.4 ± 395.6 (0.92)	1745.6 ± 376.8 (0.83)
	PPO	956.1 ± 219.4 (1.40)	755.7 ± 100.5 (1.11)	755.0 ± 172.8 (1.11)
`BipedalWalker-v3`	GFIM	-100.3 ± 8.9	-75.19 ± 6.70	-99.73 ± 8.92
	PPO	-89.18 ± 4.35	-101.7 ± 12.2	-107.3 ± 5.1

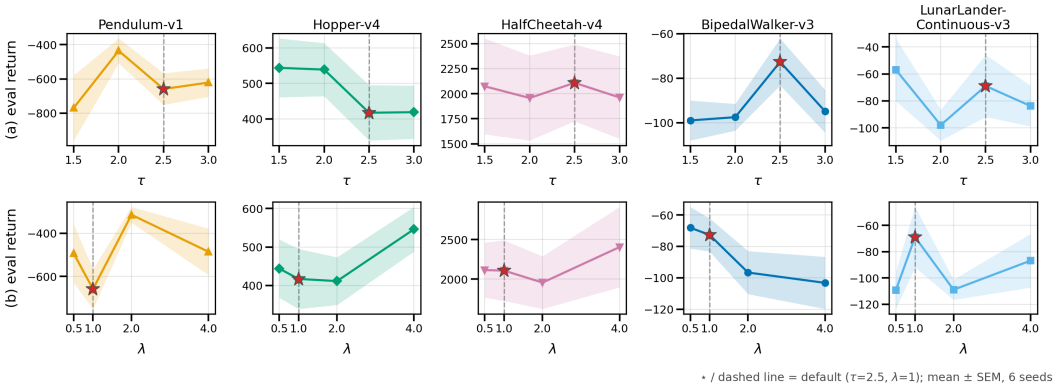


Figure 4: Hyperparameter sensitivity on the five representative tasks (mean final return over 6 seeds; error bars show the standard error of the mean). Top row: temperature τ sweep; bottom row: scaling λ sweep. The default $(\tau, \lambda) = (2.5, 1)$ is marked by a star and a dashed line. The best configuration varies by environment.

Figure 4 and Table 8 report the results. The best configuration is different on every task: λ=2 on `Pendulum-v1` (-314.0 vs. -658.6 for the default), λ=4 on `Hopper-v4` (546.3 vs. 417.1) and `HalfCheetah-v4` (2405.9 vs. 2108.6), λ=0.5 on `BipedalWalker-v3` (-68.1 vs. -72.7), and τ=1.5 on `LunarLanderContinuous-v3` (-56.9 vs. -68.9). The hyperparameters are therefore environment-sensitive rather than algorithm-level: they interact with the task, not just with the algorithm. The same pattern appears from the other direction. A single off-default setting already changes the comparison with the baseline from task to task. At τ=2, for example, GFIM-PPO clearly improves over PPO on `Hopper-v4` (538.9 vs. 404.5), `HalfCheetah-v4` (1957.1 vs. 681.9), and `BipedalWalker-v3` (-97.5 vs. -99.2), yet still trails it on `Pendulum-v1` and `LunarLanderContinuous-v3`. At the same time, the default is not fragile: it lies within about one standard error of the best configuration on `BipedalWalker-v3`, `HalfCheetah-v4`, and `LunarLanderContinuous-v3`, and is clearly beaten only on `Pendulum-v1`. We keep the default untuned, accepting the Pendulum gap as the cost of a single configuration.

The two parameters are inversely coupled in the mapping. The SRN logit is $z_i = -\lambda\tilde{\nu}_i/\tau$, so lowering τ and raising λ both steepen the decay of w as a function of $\tilde{\nu}_i$; equivalently, τ controls the same sensitivity threshold as $1/\lambda$. This explains why the per-task optimum can be reached from either knob. On `Pendulum-v1` both τ=2 and λ=2 recover most of the gap against the default; on `Hopper-v4` τ=1.5 and λ=4 reach nearly the same return. The parameters are therefore not independent degrees of freedom, but two parameterizations of a single sensitivity-threshold control. Figure 5 makes the relationship explicit: lower τ and higher λ produce the same steep weight curve, and the default sits in the middle of the workable range on both axes.

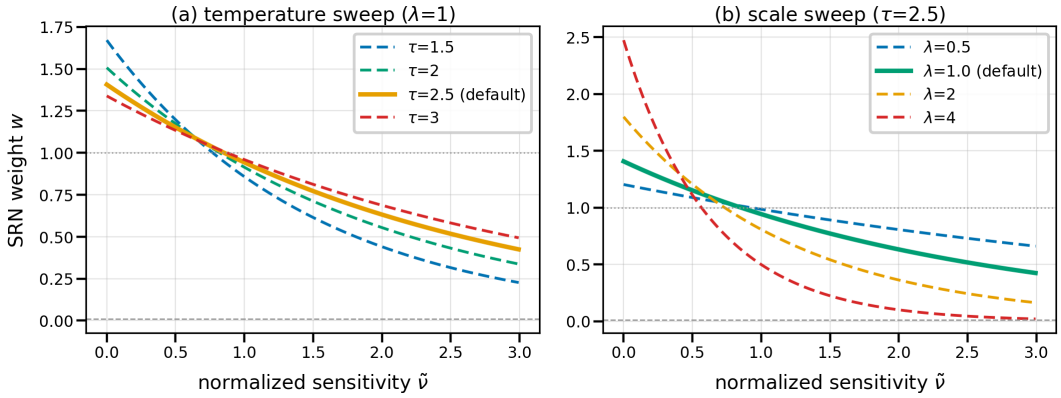


Figure 5: SRN weight w as a function of the normalized sensitivity $\tilde{\nu}$ for (a) several temperatures τ at $\lambda = 1$ and (b) several scales λ at $\tau = 2.5$, computed on a synthetic batch with $\nu = g^2$. The default (τ=2.5, λ=1) is drawn with a thicker line. Lower τ and higher λ steepen the decay in the same way, which is why the two hyperparameters partly substitute for each other in Table 8.

Table 8: Sensitivity of GFIM to the temperature τ (top) and the entropy-weight scale λ (bottom) on five representative environments (mean $\pm$ SEM, 6 seeds, final checkpoint). **Bold** marks the best mean within each sweep, underlined the runner-up.

Environment	Default (τ=2.5)	τ=1.5	τ=2	τ=3
`Pendulum-v1`	-658.6 ± 91.9	-768.5 ± 189.8	$\mathbf{-436.0 \pm 71.8}$	-621.3 ± 82.3
`Hopper-v4`	417.1 ± 78.0	$\mathbf{543.9 \pm 82.9}$	538.9 ± 74.7	418.8 ± 75.0
`HalfCheetah-v4`	$\mathbf{2108.6 \pm 383.0}$	2073.6 ± 478.0	1957.1 ± 423.6	1962.5 ± 411.2
`BipedalWalker-v3`	$\mathbf{-72.68 \pm 10.54}$	-98.96 ± 8.98	-97.51 ± 6.01	-94.85 ± 9.58
`LunarLanderContinuous-v3`	-68.93 ± 22.91	$\mathbf{-56.90 \pm 23.18}$	-97.91 ± 11.35	-83.77 ± 15.10

Environment	Default (λ=1)	λ=0.5	λ=2	λ=4
`Pendulum-v1`	-658.6 ± 91.9	-493.3 ± 135.3	$\mathbf{-314.0 \pm 36.4}$	-485.4 ± 107.3
`Hopper-v4`	417.1 ± 78.0	444.0 ± 75.6	411.4 ± 62.0	$\mathbf{546.3 \pm 58.0}$
`HalfCheetah-v4`	2108.6 ± 383.0	2114.5 ± 348.3	1956.3 ± 332.8	$\mathbf{2405.9 \pm 507.8}$
`BipedalWalker-v3`	-72.68 ± 10.54	$\mathbf{-68.11 \pm 13.23}$	-96.67 ± 13.70	-103.2 ± 16.6
`LunarLanderContinuous-v3`	$\mathbf{-68.93 \pm 22.91}$	-109.3 ± 14.1	-108.9 ± 7.7	-86.81 ± 20.34

K Computational Overhead

Computational Overhead

The Fisher-trace weights add one extra backward pass through the critic per update. Table 9 compares the throughput and per-update wall time of GFIM-PPO with the PPO baseline on the representative tasks. On three of the five tasks the throughput is within 1% of the baseline; the largest reduction is on `BipedalWalker-v3` (-13.8% steps per second, $+10.7\%$ wall time per update), and the average across the five tasks is -3.2% throughput and $+3.1\%$ wall time. The overhead is therefore modest, though not uniform across tasks.

L Percentage Improvement Definition

Because reward scales vary across tasks, the main text reports directional improvements and significance rather than normalized percentages. Where percentages are reported in the appendix, they are computed as

$$\text{improvement} = \frac{\bar{R}^{\text{GFIM}} - \bar{R}^{\text{PPO}}}{|\bar{R}^{\text{PPO}}|} \times 100\%, \tag{34}$$

Table 9: Computational overhead on five representative environments (mean over 6 seeds). sps = environment steps per second; wall = average update wall time in seconds; extra backward = average number of additional backward passes per update.

Environment	PPO sps	GFIM sps	PPO wall (s)	GFIM wall (s)	extra backward
`Pendulum-v1`	675.7	683.1	0.1521	0.1531	200.0
`Hopper-v4`	634.4	633.8	0.1535	0.1511	3190.5
`HalfCheetah-v4`	446.9	445.6	0.2702	0.2759	1000.0
`BipedalWalker-v3`	390.5	336.8	0.3855	0.4267	67.3
`LunarLanderContinuous-v3`	402.3	390.4	0.2452	0.2532	109.8

where $\bar{R}^{\mathrm{GFIM}}$ and $\bar{R}^{\mathrm{PPO}}$ are the mean final returns over the 6 seeds for GFIM-PPO and the PPO baseline, respectively. Negative returns are handled by the absolute value in the denominator; on tasks whose returns cross zero across seeds, percentage values are unstable and are therefore not reported.

M Discussion and Limitations

GFIM-PPO treats the critic as a local probe of gradient trustworthiness: where the value landscape is steep, the directed update is down-weighted and exploration takes over. This design principle is not specific to PPO—any actor-critic method with a differentiable critic can, in principle, replace uniform sample weighting with the same Fisher-trace signal. The current instantiation is built on PPO for its simplicity and wide adoption; we plan to evaluate the method on other actor-critic algorithms in future work.

The derivation assumes a homoscedastic Gaussian return model, which is convenient but strong. In practice, standard PPO minimizes mean squared error without enforcing constant variance, so the assumption serves to simplify the Fisher trace rather than to assert a property of the environment. A natural extension is to combine the current approach with methods that explicitly model or constrain return variance; under such a model the heteroscedastic term in Theorem 1 would become part of the sensitivity signal and could carry additional information beyond the mean-gradient term used here.

The ablations suggest no single component dominates across tasks; the full combination is the most robust configuration precisely because it does not rely on any one environment property. Broader limitations include the focus on on-policy clipped updates, continuous control, and moderate training budgets. Whether the signal generalizes to off-policy or model-based settings, longer horizons, or state-dependent exploration policies remains open.

N Evaluation Learning Curves

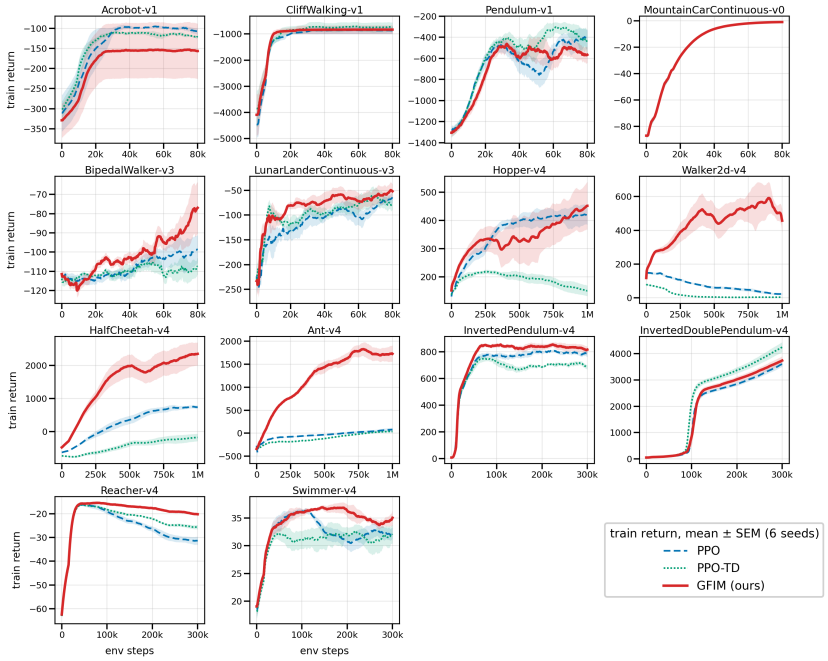


Figure 6: Smoothed training return over environment steps for all 14 tasks (mean over 6 seeds; shaded regions show the standard error of the mean). GFIM-PPO (red) learns faster or reaches higher returns on most MuJoCo tasks, and avoids the late-training degradation of the baseline on `Walker2d-v4`.

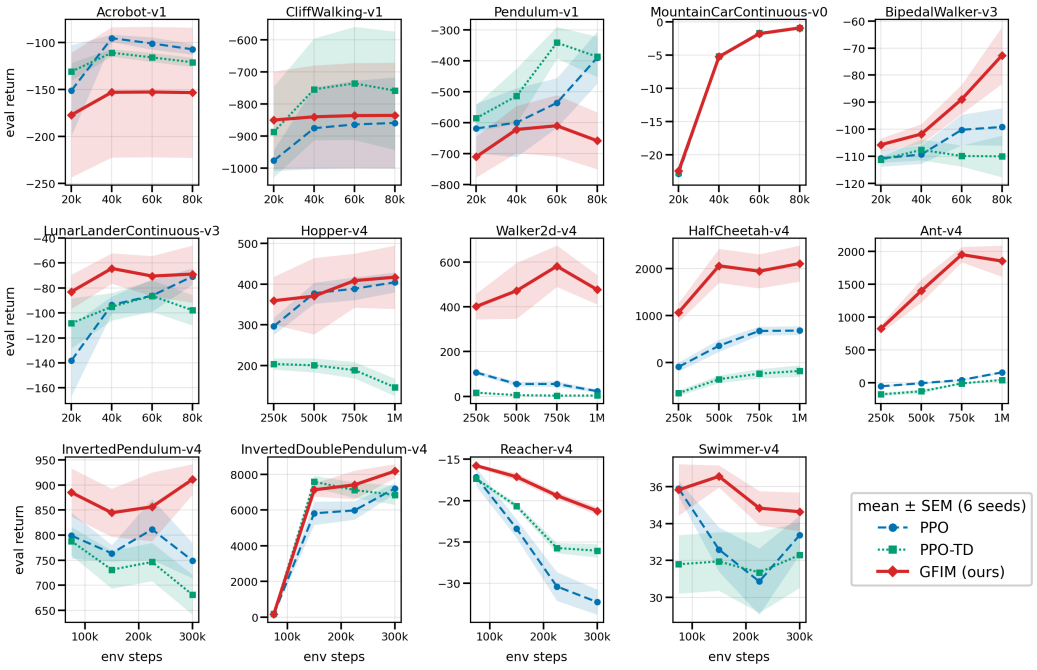


Figure 7: Evaluation return over training (mean over 6 seeds; shaded regions show the standard error of the mean) for all 14 tasks at the four evaluation checkpoints. The first two rows are the classic tasks; the third row contains the small and large MuJoCo tasks. GFIM-PPO (red) leads at every checkpoint on seven tasks and improves late on the large MuJoCo tasks where the PPO baseline stagnates or degrades.