MODEL-FREE UMBRELLA REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

We study hard reinforcement learning problems that combine sparse reward, state traps, and the absence of a terminal state. After an agent enters a trap, the trajectories it collects provide little evidence about an escape route. Umbrella Reinforcement Learning (URL) tackles this setting with an ensemble over the state space. Its objective combines return with the entropy of the ensemble's joint state and action distribution. The entropy encourages broad state coverage, while the return term favors rewarding behavior. The original URL formulation is written in continuous time and requires analytical state dynamics and their divergence. We introduce Model-Free Umbrella Reinforcement Learning (MF URL), a discrete time instantiation trained from independent one step queries to a resettable simulator, without analytical dynamics or policy trajectories, at the cost of resets to arbitrary states and many simulator queries. MF URL adapts the DualDICE objective by using a known proposal density as the denominator of the learned density ratio, so that the ratio determines the discounted state occupancy density. A second saddle point objective trains a critic with separate value and advantage heads. On Multi Valley Mountain Car and StandUp, MF URL learns policies that reach and remain in the reward regions, with returns comparable to Model Based URL. Standard SAC and PPO remain well below both umbrella methods, even when PPO uses task specific reward shaping. The learned densities recover the main structure of the Monte Carlo reference state distributions under the learned policies. A diagnostic comparison with MF URL SAC, which uses a temporal difference critic, finds that the saddle point critic agrees more closely with Monte Carlo reference returns under the policy learned by each method.

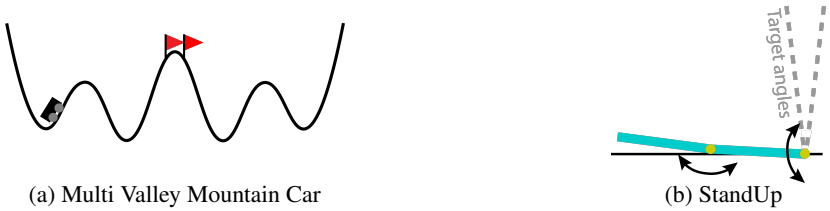
1 INTRODUCTION

Some reinforcement learning problems combine three difficulties: reward is available only after a successful sequence of actions, the dynamics contain traps that are hard to leave, and the objective is a region that must be maintained rather than a terminal state that must be reached (Nuzhin & Brilliantov, 2025). We call tasks with this combination hard reinforcement learning problems. A policy trapped far from the reward produces uninformative trajectories, while a policy that reaches the target must still learn how to remain there. The two tasks in Figure 1 have low dimensional state spaces and simple dynamics, yet PPO (Schulman et al., 2017), Random Network Distillation (Burda et al., 2019), and iLQR (Li & Todorov, 2004) make no learning progress on them in the experiments of Nuzhin & Brilliantov (2025).

The difficulty stems from the training distribution, not only from the policy update. When data are collected along trajectories and a policy repeatedly enters the same trap, its batches contain the same unrewarded states and offer little evidence about how to escape. Intrinsic bonuses (Pathak et al., 2017; Burda et al., 2019; Bellemare et al., 2016) densify the signal but are still evaluated where the policy already goes. Potential based shaping (Ng et al., 1999) helps when an informative potential is available, but constructing one requires detailed knowledge of the task.

Umbrella Reinforcement Learning (Umbrella RL) (Nuzhin & Brilliantov, 2025), named after umbrella sampling in statistical physics (Torrie & Valleau, 1977; Frenkel & Smit, 1996), changes the source of training data. It represents an ensemble of agents by a density $p(s, t)$ over the state space and maximizes the ensemble return together with its joint state and action entropy. Without reward, the entropy spreads the ensemble. Once rewarding states enter its support, the return competes with the entropy and the density can concentrate near the target. Training states are drawn from a proposal

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) Multi Valley Mountain Car (b) StandUp

Figure 1: Two hard problems. Multi Valley Mountain Car rewards remaining near the central peak, reachable only by building momentum across several valleys. StandUp rewards an upright arm with two links that must first fold and swing up, because the torques cannot lift it directly. Neither task has a terminal state. Adapted from Nuzhin & Brilliantov (2025, Fig. 1(c,d)) (CC BY 4.0).

distribution over the admissible state space rather than from policy trajectories. The method is derived in continuous time, where the density and value functions satisfy steady state partial differential equations whose coefficients include the state change rate vector field $v(s, a)$ and its divergence $\nabla_s \cdot v$. Both must be supplied analytically for each environment, even when a resettable simulator is available.

We develop Model-Free Umbrella Reinforcement Learning (MF URL), which keeps the ensemble objective and the training on proposal states but replaces the analytical dynamics with sampled one step transitions. Bellman and occupancy fixed point equations replace the differential equations. The quantity that remains unknown is the discounted state occupancy density $d^\pi(s)$, whose logarithm enters the shaped reward even at states the policy rarely visits. We estimate it with a saddle point objective adapted from DualDICE (Nachum et al., 2019), referenced to the known proposal density $m(s)$ so that the learned ratio $d^\pi(s)/m(s)$ yields an absolute density, and we learn the critic using the Fenchel dual form of the Bellman residual, as in SBEED (Dai et al., 2018), with a different parameterization and training distribution.

Contributions.

- We formulate Umbrella RL in discrete time for a resettable simulator. Bellman and occupancy fixed point equations replace the steady state partial differential equations of Model-Based URL, so the method does not require the state change rate vector field $v(s, a)$ or its divergence $\nabla_s \cdot v$, and training uses independently sampled one step simulator transitions.
- We adapt two saddle point estimators to this setting. Using the known proposal density $m(s)$ as the reference measure turns a DualDICE (Nachum et al., 2019) ratio into the absolute discounted state occupancy density. The critic combines the SBEED (Dai et al., 2018) saddle-point form of the mean squared Bellman residual with the value and advantage heads of the dueling architecture (Wang et al., 2016).
- We evaluate both components on Multi Valley Mountain Car and StandUp. MF URL reaches returns comparable to Model-Based URL using only simulator queries, while SAC, PPO, and A2C on the sparse reward remain near zero and shaped PPO stays well below both umbrella methods. Each learned component is verified against a Monte Carlo reference computed under the policy and density learned in the same run the density recovers the spatial structure of the occupancy, and the saddle point critic matches its reference more closely than the temporal difference critic matches its own.

2 Related Work

Exploration in hard reinforcement learning problems. Intrinsic motivation adds novelty rewards based on prediction error from a learned forward model (Pathak et al., 2017), error against a distilled random network (Burda et al., 2019), or visitation counts (Bellemare et al., 2016). Reward shaping instead changes the reward function itself (Ng et al., 1999). These methods modify the reward observed along policy trajectories, but they do not provide training information at independently sampled states. Other approaches change how experience is collected or reused, by returning to previously reached states (Ecoffet et al., 2021), pretraining without task reward (Laskin et al., 2021),

relabelling failed episodes with other goals (Andrychowicz et al., 2017), or decomposing control across time scales (Nachum et al., 2018). In each case, the training data still arrive as trajectories.

Objectives over the state distribution. Hazan et al. (2019) study objectives defined over state visitation frequencies and give an efficient algorithm based on a planning oracle, with sample and computational guarantees in the tabular setting. Bolland et al. (2024) define an intrinsic reward from the relative entropy of the discounted future state and action visitation measure, which they obtain as the fixed point of a contraction operator. Umbrella RL instead uses a state space ensemble inspired by umbrella sampling and trains it without policy trajectories. State distribution entropy also differs from policy entropy (Ziebart, 2010; Haarnoja et al., 2018a): policy entropy randomizes actions at visited states, whereas state distribution entropy acts on the visitation measure itself.

Estimating occupancy measures. The successor representation (Dayan, 1993) and successor features (Barreto et al., 2017) encode expected future occupancy conditioned on a starting state for value prediction and transfer. MF URL instead requires the marginal discounted occupancy density of the policy being trained. DualDICE (Nachum et al., 2019) and GenDICE (Zhang et al., 2020) estimate occupancy ratios for off policy evaluation, where a fixed target policy is evaluated from a logged dataset whose distribution is unknown. MF URL uses the DualDICE objective with the known proposal as reference, a policy that changes during training, and fresh one step transitions from a resettable simulator, so the learned ratio determines the absolute occupancy density rather than serving as an importance weight.

Duality in value estimation. SBEED (Dai et al., 2018) reformulates the smoothed Bellman optimality equation as a primal and dual problem using Nesterov smoothing and the Fenchel representation of the squared residual, learning a state value function and a policy jointly from behavior policy samples. Our critic uses the same representation for a different problem: it evaluates a fixed policy under the density shaped reward at proposal states, with a value head and a policy centered advantage head, and the policy is updated in a separate step.

3 Background

This section fixes notation and states the results the method builds on. Derivations are given in Appendix B.

3.1 Markov decision processes

We consider a Markov decision process $(\mathcal{S}, \mathcal{A}, P, r, \gamma, d_0)$ with transition kernel $P(s' \mid s, a)$, reward $r(s, a)$, per step discount $\gamma \in [0, 1)$, and initial state distribution d_0. The quality of a policy $\pi(a \mid s)$ is measured by its expected discounted return

$$J(\pi) = \mathbb{E}_\pi\Big[\sum_{t\geq 0} \gamma^t r(s_t, a_t)\Big], \qquad s_0 \sim d_0, \quad a_t \sim \pi(\cdot \mid s_t), \quad s_{t+1} \sim P(\cdot \mid s_t, a_t). \tag{1}$$

The state value function $V^\pi(s)$ and the action value function $Q^\pi(s, a)$ are the same expectation conditioned on $s_0 = s$ and on $(s_0, a_0) = (s, a)$, respectively, so $J(\pi) = \mathbb{E}_{s_0\sim d_0}[V^\pi(s_0)]$. They satisfy the Bellman equations

$$Q^\pi(s, a) = r(s, a) + \gamma\, \mathbb{E}_{s'\sim P(\cdot|s,a)}[V^\pi(s')], \qquad V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}[Q^\pi(s, a)], \tag{2}$$

and the advantage $A^\pi = Q^\pi - V^\pi$ measures how much an action improves on the policy average.

3.2 Discounted state occupancy

The discounted state occupancy density of π is

$$d^\pi(s) = (1 - \gamma) \sum_{t\geq 0} \gamma^t p_t^\pi(s), \tag{3}$$

where p_t^π is the density (or probability mass) of s_t. It converts sums over time into averages over states, $J(\pi) = (1 - \gamma)^{-1}\mathbb{E}_{s\sim d^\pi,\, a\sim\pi}[r(s, a)]$, and the policy gradient theorem (Sutton et al., 2000) reads

$$\nabla_\omega J(\pi_\omega) = \tfrac{1}{1-\gamma}\, \mathbb{E}_{s\sim d^{\pi_\omega},\, a\sim\pi_\omega(\cdot|s)}\big[\nabla_\omega \log \pi_\omega(a \mid s)\, A^{\pi_\omega}(s, a)\big]. \tag{4}$$

The density d^π is also the unique solution of the one step fixed point equation

$$d^\pi(s) = (1-\gamma)d_0(s) + \gamma \int d^\pi(s')\, \mathbb{E}_{a\sim\pi(\cdot|s')}[P(s \mid s', a)]\, \mathrm{d}s', \tag{5}$$

which constrains d^π at each state through its predecessor states.

3.3 Umbrella reinforcement learning

Umbrella RL (Nuzhin & Brilliantov, 2025) replaces the single agent by a continuous ensemble of agents with state density $p_t(s) = p(s,t)$ and maximizes the URL objective

$$J^{\mathrm{URL}}(\pi) = J_{\mathrm{ct}}(\pi) + \alpha\, H[p\pi], \tag{6}$$

where J_{ct} and H are, respectively, the continuous time return and the joint state and action entropy of the ensemble,

$$J_{\mathrm{ct}}(\pi) = \int_0^\infty \gamma_c^t \big\langle \mathbb{E}_{a\sim\pi}[r(s,a)] \big\rangle_{p_t} \mathrm{d}t, \qquad H[p\pi] = -\int_0^\infty \gamma_c^t \big\langle \mathbb{E}_{a\sim\pi}[\log(p_t(s)\pi(a \mid s))] \big\rangle_{p_t} \mathrm{d}t. \tag{7}$$

Here $\gamma_c \in (0,1)$ is the discount factor per unit time, $\alpha > 0$ weights the entropy, and $\langle\cdot\rangle_{p_t}$ averages over $s \sim p_t$. Each agent evolves as $\dot{s} = v(s,a)$, where v is the state change rate vector field, so the density obeys the continuity equation $\partial_t p_t + \nabla_s \cdot (p_t \bar{v}) = 0$ with $\bar{v} = \mathbb{E}_{a\sim\pi}[v]$ and $p_0 = d_0$. While the ensemble covers no rewarding state, $J_{\mathrm{ct}} = 0$ and maximizing H spreads it over reachable states. Once rewarding states receive mass, J_{ct} competes with H, and for small α the ensemble concentrates on the rewarding region. Unlike the policy entropy $H[\pi(\cdot \mid s)]$ of maximum entropy RL, the ensemble entropy acts on which states are visited: a policy can be maximally random at every visited state and still remain in one valley.

Model Based URL. Nuzhin & Brilliantov (2025) replace p_t in the entropy by the time averaged density $\bar{p}(s) = |\log\gamma_c| \int_0^\infty \gamma_c^t\, p(s,t)\, \mathrm{d}t$, the continuous time counterpart of equation 3. The entropy then acts as an additional reward term, and the policy gradient takes the form of equation 4,

$$\nabla_\omega J^{\mathrm{URL}} \propto \big\langle \mathbb{E}_{a\sim\pi_\omega}[\nabla_\omega \log \pi_\omega(a \mid s)\, A_u(s,a)] \big\rangle_{\bar{p}}, \tag{8}$$

with the modified advantage and effective reward

$$A_u(s,a) = r_u(s,a) + v(s,a)\cdot\nabla_s V_u(s) - |\log\gamma_c|\, V_u(s), \qquad r_u = r - \tilde{\alpha}\log\big(\bar{p}(s)\pi(a \mid s)\big), \tag{9}$$

where $\tilde{\alpha} = \alpha|\log\gamma_c|$ and V_u is the value function of r_u. The value function and the averaged density are the steady state solutions of

$$\bar{v}\cdot\nabla_s V_u + \bar{r}_u - |\log\gamma_c|\, V_u = 0, \qquad \nabla_s\cdot(\bar{p}\,\bar{v}) - \log\gamma_c\,(\bar{p} - d_0) = 0, \tag{10}$$

with $\bar{r}_u = \mathbb{E}_{a\sim\pi}[r_u]$. The first is the continuous time Bellman equation $\mathbb{E}_{a\sim\pi}[A_u] = 0$, and the second the counterpart of equation 5. Networks for π, V_u, and $\bar{p}$ are trained on states drawn independently from a proposal distribution $m(s)$, uniform in the original implementation, which also replaces $\bar{p}$ as the averaging distribution in equation 8. No trajectories are generated during training. The model enters through the directional derivative $v\cdot\nabla_s V_u$ and the divergence $\nabla_s\cdot(\bar{p}\,\bar{v}) = \bar{p}\,\mathbb{E}_{a\sim\pi}[\nabla_s\cdot v + v\cdot\nabla_s\log(\pi(a \mid s)\bar{p}(s))]$. Both v and $\nabla_s\cdot v$ must therefore be available as numerical functions of (s,a) for each environment. Simulator based tasks usually do not expose them, and deriving them requires additional analytical work for every new environment.

3.4 Soft Actor Critic

Soft Actor Critic (SAC) (Haarnoja et al., 2018a;b) adds the policy entropy $\alpha_{\mathrm{SAC}} H[\pi(\cdot \mid s_t)]$ to the reward (Ziebart, 2010) and learns Q^π by temporal difference regression on replay buffer batches. It maintains two Q networks, uses the minimum of their estimates in the target to reduce positive bias, and updates the target parameters by Polyak averaging (Polyak & Juditsky, 1992). Standard SAC is our trajectory based baseline. MF URL SAC trains on proposal states and optimizes the same density shaped, entropy regularized objective as MF URL, but learns the critic by temporal difference regression, with the action entropy in the soft target and actor loss (Appendix A.6). Unless stated otherwise, MF URL refers to the saddle point realization of Section 4.

3.5 DUAL FORMULATIONS OF ONE STEP FIXED POINT EQUATIONS

MF URL learns the action value function of a density shaped reward, defined by equation 2, and the occupancy density, defined by equation 5, from one step simulator queries at proposal states. Neither equation can be imposed directly with such samples: the squared Bellman residual contains a conditional expectation over successors, and equation 5 constrains $d^\pi(s)$ through the predecessors of s, which a forward query does not provide. The formulations below replace each equation by a saddle point objective over sampled states, their one step successors, and initial states. Sections 4.2 and 4.3 apply them.

Bellman residual. Let $\mu(s,a)$ be a sampling distribution, and write $\mathbb{E}_\mu$ for the expectation over $(s,a) \sim \mu$, $s' \sim P(\cdot \mid s,a)$, and $a' \sim \pi(\cdot \mid s')$. For a candidate Q, define the sampled Bellman residual $\delta_Q = r(s,a) + \gamma Q(s',a') - Q(s,a)$. Then Q^π minimizes the mean squared Bellman error

$$\min_Q \ \tfrac{1}{2}\, \mathbb{E}_{(s,a)\sim\mu}\big[\big(\mathbb{E}[\delta_Q \mid s,a]\big)^2\big]. \tag{11}$$

One successor per (s,a) gives an unbiased estimate of $\mathbb{E}[\delta_Q \mid s,a]$ but not of its square, since $\mathbb{E}[\delta_Q^2 \mid s,a] = (\mathbb{E}[\delta_Q \mid s,a])^2 + \mathrm{Var}(\delta_Q \mid s,a)$. An unbiased estimate would require two independent successors of the same (s,a), which is the double sampling problem (Baird, 1995). The Fenchel identity $\frac{1}{2}x^2 = \max_h(xh - \frac{1}{2}h^2)$, applied at each (s,a), gives the equivalent saddle point problem

$$\min_Q \max_h \ \mathbb{E}_\mu\big[\delta_Q\, h(s,a) - \tfrac{1}{2}h(s,a)^2\big], \tag{12}$$

whose inner maximizer is the witness $h^*(s,a) = \mathbb{E}[\delta_Q \mid s,a]$, the representation SBEED (Dai et al., 2018) uses for the smoothed Bellman optimality equation. Because h depends only on (s,a), the successor enters linearly, so minibatch averages of equation 12 and their gradients for fixed (Q,h) are unbiased with one successor per state and action pair (Dai et al., 2018); replacing the pointwise maximum by a maximum over functions follows from the interchangeability principle (Shapiro et al., 2009) (Appendix B.2).

Occupancy ratio. DualDICE (Nachum et al., 2019) recovers the occupancy through its ratio $w = d^\pi/\mu$ to a sampling density μ. In its state marginal form, states are drawn from μ and actions from π. If d^π is absolutely continuous with respect to μ and w is bounded, as in DualDICE, then w solves

$$\min_\nu \max_w \ \mathbb{E}_{s\sim\mu,\, a\sim\pi,\, s'\sim P}\big[w(s)\big(\nu(s) - \gamma\nu(s')\big) - \tfrac{1}{2}w(s)^2\big] - (1-\gamma)\,\mathbb{E}_{s_0\sim d_0}[\nu(s_0)], \tag{13}$$

with $w^* = d^\pi/\mu$ at the saddle point. Two duality steps give this objective (Appendix B.3): the ratio minimizes $\frac{1}{2}\mathbb{E}_\mu[w^2] - \mathbb{E}_{d^\pi}[w]$, and the substitution $w(s) = \nu(s) - \gamma\,\mathbb{E}[\nu(s') \mid s]$ with equation 5 and the Fenchel identity removes the need for samples from d^π, so minibatch averages are unbiased (Nachum et al., 2019). For unrestricted functions and exact expectations, every saddle point of equation 12 satisfies the Bellman equation and every saddle point of equation 13 has $w = d^\pi/\mu$ (Appendix B.4); convergence of alternating stochastic gradient updates with neural networks is not guaranteed (Nachum et al., 2019).

4 MODEL-FREE UMBRELLA REINFORCEMENT LEARNING

MF URL keeps the ensemble objective and the proposal state training of Model Based URL and removes its analytical requirements v and $\nabla_s \cdot v$. It works in discrete time with a generative model (Section 4.1), estimates the occupancy density with a saddle point estimator referenced to the known proposal density (Section 4.2), and learns the critic with a saddle point objective (Section 4.3). Algorithm 1 gives the full procedure.

4.1 SETTING AND DISCRETE TIME OBJECTIVE

We assume access to a generative model of the environment: for any admissible state and action pair (s,a), a simulator returns a reward $r(s,a,s')$ and a successor state $s' \sim P(\cdot \mid s,a)$ after one time step Δt. We call such a simulator resettable, because it can be queried at arbitrary states. Training

draws states independently from a known proposal density $m(s)$ and initial states from d_0, and we assume $d^\pi \ll m$ with a bounded ratio d^π/m. Successors are used only in the update that queried them and are never extended into trajectories.

Model Based URL maximizes the continuous time return of the effective reward r_u in equation 9. We use its discrete time counterpart. The reward r is a rate, as in equation 7, so one step accumulates $\Delta t\, r$, and the per step discount is $\gamma_d = \gamma_c^{\Delta t}$. Substituting $t = k\Delta t$, $\mathrm{d}t \to \Delta t$, and $\gamma_c^t \to \gamma_d^k$, and replacing $\bar{p}$ by the occupancy d^π of equation 3 with $\gamma = \gamma_d$, gives

$$J_{\mathrm{MFURL}}(\pi) = \mathbb{E}_\pi\Big[\textstyle\sum_{k\geq 0} \gamma_d^k\, \hat{r}_k\Big], \qquad \hat{r}(s,a,s') = \Delta t\,\big[r(s,a,s') - \lambda \log(d^\pi(s)\pi(a \mid s))\big]. \tag{14}$$

with $\hat{r}_k = \hat{r}(s_k, a_k, s_{k+1})$. The coefficient λ plays the role of $\tilde{\alpha}$ in equation 9 and weights both logarithms, so the shaping term is the joint state and action entropy of the discounted occupancy. The densities $\bar{p}$ and d^π agree up to a factor that tends to one as $\Delta t \to 0$ and only shifts $\hat{r}$ by a constant (Appendix C.1). Equation 14 is the return of Section 3.1 with per step reward $\hat{r}$ and discount γ_d, so Section 3 applies with $\gamma = \gamma_d$. Although $\hat{r}$ depends on π through d^π and $\log \pi$, the terms produced by differentiating these logarithms have zero expectation, so the policy gradient of J_{MFURL} is equation 4 with the advantage of $\hat{r}$ computed for fixed d^π (Appendix C.2).

4.2 Occupancy density estimation

The density enters $\hat{r}$ at every proposal state, including states the policy rarely visits, so it must be estimated on the support of m. We apply the occupancy saddle point 13 with $\mu = m$ and $\gamma = \gamma_d$. A positive valued network ζ_ξ (Softplus output) represents the ratio $w = d^\pi/m$, and a network ν_ψ represents the dual function. With the occupancy residual $\delta_\nu(s,s') = \nu_\psi(s) - \gamma_d\, \nu_\psi(s')$, the saddle point problem is

$$\min_\psi \max_\xi\ \mathbb{E}_{s\sim m,\, a\sim\pi_\omega,\, s'\sim P}\big[\zeta_\xi(s)\,\delta_\nu(s,s') - \tfrac{1}{2}\zeta_\xi(s)^2\big] - (1-\gamma_d)\,\mathbb{E}_{s_0\sim d_0}[\nu_\psi(s_0)]. \tag{15}$$

The ratio network ζ_ξ is the maximizing player and ν_ψ the minimizing player. At the saddle point $\zeta_\xi = d^\pi/m$, and the density used in $\hat{r}$ is

$$\log \widehat{d^\pi}(s) = \log \zeta_\xi(s) + \log m(s). \tag{16}$$

We solve equation 15 by alternating gradient steps on

$$\mathcal{L}_\zeta(\xi) = \mathbb{E}\big[\tfrac{1}{2}\zeta_\xi^2 - \mathrm{sg}(\delta_\nu)\,\zeta_\xi\big], \qquad \mathcal{L}_\nu(\psi) = \mathbb{E}\big[\mathrm{sg}(\zeta_\xi)\,\delta_\nu\big] - (1-\gamma_d)\,\mathbb{E}_{s_0\sim d_0}[\nu_\psi(s_0)], \tag{17}$$

where sg is the stop gradient and the expectations are those of equation 15. Minimizing $\mathcal{L}_\zeta$ ascends equation 15 in ξ, and minimizing $\mathcal{L}_\nu$ descends it in ψ. The expectation over $a \sim \pi_\omega$ is computed exactly by summing over the discrete actions, with the policy held fixed, and no samples from d^π are required. Unlike DualDICE in off policy evaluation, the reference density is known, so the ratio yields an absolute density. Actions come from the current policy, so the ratio depends on the state only. The policy changes during training, so each update uses fresh simulator queries.

4.3 Saddle point critic

The critic evaluates the current policy π_ω under $\hat{r}$, with $\widehat{d^\pi}$ in place of d^π. By Section 3.1 with $\gamma = \gamma_d$, its action value function satisfies (Appendix C.2)

$$Q(s,a) = \mathbb{E}_{s'\sim P(\cdot|s,a)}\big[\hat{r}(s,a,s') + \gamma_d V(s')\big], \qquad V(s) = \mathbb{E}_{a\sim\pi_\omega(\cdot|s)}[Q(s,a)]. \tag{18}$$

We represent Q with a value head V_θ and an advantage head A_θ, as in the dueling architecture (Wang et al., 2016), but center the advantage under the current policy instead of the uniform mean over actions:

$$Q_\theta(s,a) = V_\theta(s) + A_\theta(s,a) - \mathbb{E}_{b\sim\pi_\omega(\cdot|s)}[A_\theta(s,b)]. \tag{19}$$

Then $\mathbb{E}_{a\sim\pi_\omega}[Q_\theta(s,a)] = V_\theta(s)$, so the value head is the policy average in equation 18, and the sampled Bellman residual needs no successor action:

$$\delta_\theta(s,a,s') = \hat{r}(s,a,s') + \gamma_d V_\theta(s') - Q_\theta(s,a). \tag{20}$$

The parameterization does not restrict the function class: any Q has this form with $V = \mathbb{E}_{\pi_\omega}[Q]$ and $A = Q$, so the solution of the critic problem is unchanged (Appendix C.3). We apply the Bellman

Algorithm 1 MF URL training

Require: proposal density m, initial distribution d_0, generative model, reference distribution q, iterations K, update counts $K_\zeta, K_\nu, K_h, K_Q, K_\pi$

Initialize policy π_ω, density ratio ζ_ξ, density dual ν_ψ, critic (V_θ, A_θ), witness h_η
for $k = 1, \dots, K$ **do**
  Density estimation
  for K_ζ steps **do**
    Sample $s_i \sim m$, $a_i \sim \pi_\omega(\cdot \mid s_i)$, $s_i' \sim P(\cdot \mid s_i, a_i)$, then update ξ on $\mathcal{L}_\zeta$ ▷ Eq. 17
  for K_ν steps **do**
    Sample a fresh batch (s_i, a_i, s_i') and $s_{0,j} \sim d_0$, then update ψ on $\mathcal{L}_\nu$ ▷ Eq. 17
  Critic estimation
  for K_h steps **do**
    Sample $s_i \sim m$, $a_i \sim q(\cdot \mid s_i)$, query (r_i, s_i'), and compute $\delta_\theta(s_i, a_i, s_i')$ ▷ Eq. 20
    Update η on $\mathcal{L}_h$ ▷ Eq. 22
  for K_Q steps **do**
    Sample a fresh batch, compute δ_θ, and update θ on $\mathcal{L}_Q$ ▷ Eq. 22
  Policy improvement
  for K_π steps **do**
    Sample $s_i \sim m$ (no simulator queries) and update ω on $\mathcal{L}_\pi$ ▷ Eq. 23
return π_ω, $\widehat{d^\pi} = m\,\zeta_\xi$, Q_θ

saddle point 12 with states $s \sim m$ and current actions from a fixed reference distribution $q(a \mid s)$ whose support covers that of π_ω. We take q uniform and sum over all actions. A witness network $h_\eta(s, a)$ represents the conditional Bellman residual:

$$\min_\theta \max_\eta \; \mathbb{E}_{s\sim m,\, a\sim q,\, s'\sim P}\big[\delta_\theta(s, a, s')\, h_\eta(s, a) - \tfrac{1}{2} h_\eta(s, a)^2\big]. \tag{21}$$

The critic parameters θ, shared by V_θ and A_θ, minimize equation 21, and the witness parameters η maximize it. For an expressive enough critic, the saddle point has $h_\eta = \mathbb{E}[\delta_\theta \mid s, a] = 0$, so Q_θ solves equation 18. The two players are updated with

$$\mathcal{L}_h(\eta) = \mathbb{E}\big[\tfrac{1}{2} h_\eta^2 - \mathrm{sg}(\delta_\theta)\, h_\eta\big], \qquad \mathcal{L}_Q(\theta) = \mathbb{E}\big[\delta_\theta\, \mathrm{sg}(h_\eta)\big], \tag{22}$$

with the expectations of equation 21. The successor value $V_\theta(s')$ uses the online critic without a target network, so the gradient of $\mathcal{L}_Q$ passes through both $Q_\theta(s, a)$ and $V_\theta(s')$. The policy probabilities in equation 19, the density term in $\hat{r}$, and the witness are held fixed. On the deterministic benchmarks of Section 5, with action expectations computed exactly, $\mathrm{Var}(\delta_\theta \mid s, a) = 0$, so equation 21 has the same solution as direct residual minimization. We keep the saddle form because it remains unbiased under stochastic transitions without a second simulator query. The policy is a multilayer perceptron, and V_θ, A_θ, h_η, ζ_ξ, and ν_ψ are SIRENs (Sitzmann et al., 2020) (Appendix A.6).

4.4 Policy update

The actor uses a separate proposal batch and makes no simulator queries. Let π^- denote the current policy detached from the computation graph, and define the rate scaled centered advantage

$$\bar{A}_\theta(s, a) = \tfrac{1}{\Delta t}\big[A_\theta(s, a) - \mathbb{E}_{b\sim\pi^-}[A_\theta(s, b)]\big], \qquad \mathcal{L}_\pi(\omega) = -\mathbb{E}_{s\sim m,\, a\sim\pi_\omega}\big[\mathrm{sg}(\bar{A}_\theta(s, a))\big]. \tag{23}$$

The factor $1/\Delta t$ expresses the per step advantage as a rate, like A_u in equation 9. The action expectation is a sum over the discrete actions, and gradients enter only through the policy probabilities, so the update is a surrogate for equation 4 with states drawn from m instead of d^π, the substitution also made by Model Based URL, without the ratio d^π/m. The action entropy already enters through $\hat{r}$, so $\mathcal{L}_\pi$ has no separate entropy term.

5 Experiments

5.1 Experimental setup

The experiments evaluate three aspects of MF URL: its control performance relative to Model Based URL, the accuracy of the learned density under a changing policy, and the accuracy of the saddle

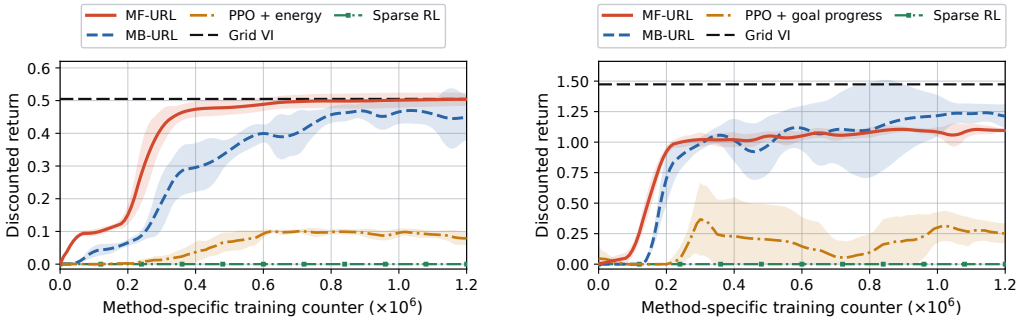
point critic relative to a temporal difference critic. An ablation then isolates the effect of the state density term on the number of updates needed to learn.

Tasks and compared methods. We use Multi Valley Mountain Car and StandUp (Figure 1). Appendix A.1 defines both tasks. They have deterministic dynamics and discrete actions (two and four) and use a fixed uniform proposal density over the valid state domain (Appendix A.2). Model Based URL is the closest reference: it trains on proposal states with the same ensemble objective but requires the state change rate vector field and its divergence in closed form. The trajectory based baselines are SAC, PPO, and A2C with the original sparse reward, and PPO with a task specific shaped reward (Appendix A.7). Grid VI is a planning reference that uses the known dynamics, and MF URL SAC appears only in the critic comparison (Appendix A.3).

Training data, budget, and protocol. MF URL queries the simulator at proposal states, Model Based URL evaluates the analytical dynamics at proposal states, and the other baselines learn from their own trajectories. Each method has a budget of 1.2×10^6 steps of its native training counter: an outer iteration for the umbrella methods and an environment transition for the trajectory based baselines, so Figure 2 does not compare sample efficiency (Appendices A.4 and A.8). Training uses independently sampled one step simulator transitions. Complete trajectories are used only for policy evaluation and Monte Carlo reference calculations. The return curves, density comparison, and critic comparison use ten training seeds per method, and the state density ablation uses five seeds per condition. Every method is evaluated with deterministic actions from the same 50 initial states for $T = 1000$ steps, using the original task reward and the discounted return $G = \sum_{k<T} \gamma_d^k \, \Delta t \, r_k$ (Appendix A.5).

5.2 Results

Control performance. On Multi-Valley Mountain Car, the sparse-reward SAC, PPO, and A2C runs (Sparse RL in Figure 2) remain near zero. Energy shaping improves PPO, but its return stays well below both umbrella methods (Figure 2a). MF URL finishes close to the Grid VI reference, with a narrower band across seeds than Model-Based URL. On StandUp, SAC makes no progress, and goal-progress shaping gives PPO a modest improvement. Both umbrella methods reach a much higher return, finish at comparable levels, and remain below the Grid VI reference (Figure 2b). On these two tasks, MF URL attains control performance comparable to Model-Based URL using only simulator queries.

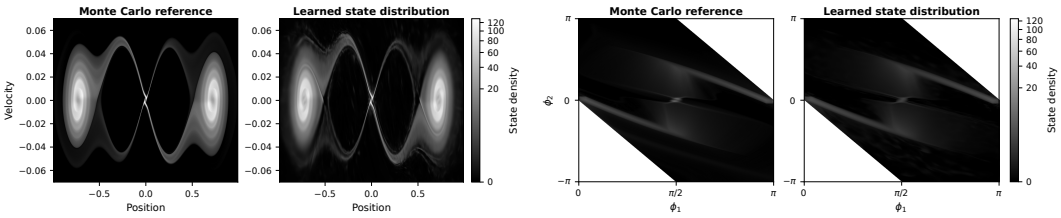


(a) Multi Valley Mountain Car

(b) StandUp

Figure 2: Discounted evaluation return over ten training seeds bands show one sample standard deviation. The horizontal axis counts outer iterations for MF URL and Model-Based URL, but environment transitions for the trajectory-based baselines.

Density estimation under a changing policy. The policy changes during training, so the density estimator tracks a moving target. At the final checkpoint, we compare each seed's learned density $\widehat{d}^{\pi} = m \, \zeta_{\xi}$ with a Monte Carlo reference from the same policy (Appendix A.9). The learned densities reproduce the main spatial structure of the references (Figure 3). In Multi Valley Mountain Car, the estimate retains the outer valley modes and the paths connecting them. In StandUp, the average map recovers the two branches of the valid state space and their junction.



(a) Multi Valley Mountain Car (b) StandUp

Figure 3: Discounted state distributions at the end of training: Monte Carlo reference (left) and learned density (right) under the same policy, averaged over ten separately normalized seed maps. White regions in StandUp lie outside the valid state space.

Critic accuracy under learned policies. For each seed, we freeze the policy, density estimator, and critic at the end of training and compute a Monte Carlo reference of the density shaped value under that policy and density estimate. Across ten seeds on both tasks, the saddle point critic matches its Monte Carlo reference more closely than the temporal difference critic of MF URL SAC matches its own (normalized RMSE, i.e., RMSE divided by the standard deviation of the reference Appendix A.10), and the temporal difference critic also varies more across StandUp seeds. The realizations also differ in architecture and current-action averaging (Appendix A.10); each critic is verified against the value function it was trained to estimate, under its own policy and density estimate.

Effect of the state density term. Removing $-\lambda \log \widehat{d}^{\pi}(s)$ from $\hat{r}$ while keeping the action entropy term removes the state distribution entropy but not the policy entropy. On Multi Valley Mountain Car, the full objective reaches high return in about half as many outer iterations as this ablation (Appendix A.11, Figure 8). Both conditions perform the same critic and policy updates per iteration, so the state density term reduces the number of these updates needed to learn. The ablation skips the density updates, so the saving is in updates, not in simulator queries.

6 Conclusion

We introduced MF URL, a discrete time instantiation of Umbrella RL that replaces the steady state conditions of the model based realization by two saddle point problems solved from one step transitions at independently sampled proposal states. Using the known proposal as the reference measure turns a DualDICE ratio into the absolute discounted occupancy density, the quantity the ensemble entropy requires and the one that cannot be read off a single transition. The critic combines the Fenchel dual form of the Bellman residual used by SBEED with a dueling value advantage head centered under the current policy, which supplies the successor term without a successor action sample. On Multi Valley Mountain Car and StandUp, MF URL reaches the return of the model-based realization without its analytical dynamics; the learned density recovers the occupancy structure of its Monte Carlo reference, and the saddle-point critic matches its reference more closely than the temporal-difference realization matches its own. Removing the state density term roughly doubles the number of outer iterations needed to reach high return on Multi Valley Mountain Car. Hard problems that previously required the analytical state rate and its divergence can therefore be solved from sampled transitions alone.

These results come at the cost of an access requirement that trajectory based methods typically do not have. MF URL requires a simulator resettable to arbitrary states, and every update draws a fresh proposal batch rather than reusing collected experience, so the comparison with trajectory based methods asks whether a task can be solved at all rather than how efficiently samples are used. Action expectations are computed exactly over small discrete action sets, and both benchmarks are low-dimensional; larger or continuous action spaces would require sampling these expectations. Nothing in the formulation assumes deterministic transitions, so stochastic environments are the direct next step.

AI USE STATEMENT

In this work, we used generative AI tools to assist with literature search and discovery, improve the clarity and readability of parts of the manuscript, and help write and debug parts of the training, evaluation, and visualization code.

Generative AI was also used to provide feedback on some experimental choices and to discuss possible interpretations of the experimental results. The authors independently reviewed these suggestions, carried out the experiments, analyzed the results, and determined the conclusions reported in the paper.

The research idea, hypotheses, methodology, and mathematical formulation were developed by the authors. Generative AI was not used to generate synthetic datasets, clean or reformat datasets, or formulate or prove mathematical claims.

All text and code produced with AI assistance were reviewed by the authors, and the experimental code was checked for its intended behavior. The authors take responsibility for the final content of the paper.

REPRODUCIBILITY STATEMENT

The implemented losses, gradient paths, and update order are stated in Section 4 and Algorithm 1. Appendix A records the environment definitions, proposal distributions, architectures, optimization settings, evaluation protocol, and simulator query accounting used in the reported runs, and Appendices B and C give the derivations. Performance curves aggregate independent training seeds as described in Section 5.1. Rollout trajectories are used only for evaluation and Monte Carlo references.

Code and checkpoints: Anonymous repository.

REFERENCES

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Leemon C. Baird. Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 30–37, 1995.

André Barreto, Will Dabney, Rémi Munos, Jonathan J. Hunt, Tom Schaul, Hado van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Rémi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Adrien Bolland, Gaspard Lambrechts, and Damien Ernst. Off-policy maximum entropy RL with future state and action visitation measures. *arXiv preprint arXiv:2412.06655*, 2024. URL https://arxiv.org/abs/2412.06655.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1810.12894.

Bo Dai, Albert Shaw, Lihong Li, Lin Xiao, Niao He, Zhen Liu, Jianshu Chen, and Le Song. SBEED: Convergent reinforcement learning with nonlinear function approximation. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1125–1134. PMLR, 2018. URL https://proceedings.mlr.press/v80/dai18c.html.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993. doi: 10.1162/neco.1993.5.4.613.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. First return, then explore. *Nature*, 590:580–586, 2021. doi: 10.1038/s41586-020-03157-9.

Daan Frenkel and Berend Smit. *Understanding Molecular Simulation: From Algorithms to Applications*. Academic Press, 1996.

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. PMLR, 2018a. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b. URL https://arxiv.org/abs/1812.05905.

Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2681–2691. PMLR, 2019. URL https://proceedings.mlr.press/v97/hazan19a.html.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. URLB: Unsupervised reinforcement learning benchmark. *NeurIPS Deep Reinforcement Learning Workshop*, 2021. URL https://openreview.net/forum?id=OLAYpF9TQtQ.

Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *Proceedings of the First International Conference on Informatics in Control, Automation and Robotics*, volume 1, pp. 222–229. SciTePress, 2004. doi: 10.5220/0001143902220229.

Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. DualDICE: Behavior-agnostic estimation of discounted stationary distribution corrections. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1906.04733.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*, pp. 278–287. Morgan Kaufmann, 1999.

Egor E. Nuzhin and Nikolai V. Brilliantov. Umbrella reinforcement learning—computationally efficient tool for hard non-linear problems. *Communications in Nonlinear Science and Numerical Simulation*, 143:108583, 2025. doi: 10.1016/j.cnsns.2024.108583. URL https://doi.org/10.1016/j.cnsns.2024.108583.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2778–2787. PMLR, 2017. URL https://proceedings.mlr.press/v70/pathak17a.html.

Boris T. Polyak and Anatoli B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory*. Society for Industrial and Applied Mathematics, 2009.

Maurice Sion. On general minimax theorems. *Pacific Journal of Mathematics*, 8(1):171–176, 1958.

Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12, 2000.

Glenn M. Torrie and John P. Valleau. Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling. *Journal of Computational Physics*, 23(2):187–199, 1977. doi: 10.1016/0021-9991(77)90121-8.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1995–2003. PMLR, 2016. URL https://proceedings.mlr.press/v48/wangf16.html.

Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. GenDICE: Generalized offline estimation of stationary values. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HkxlcnVFwB.

Brian D. Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University, 2010.

A Additional experimental details

A.1 Environment definitions

Table 1 summarizes the two tasks. Their complete definitions are given in Appendices A.1.1 and A.1.2.

Table 1: Summary of the experimental environments.

Environment	State and control	Positive reward condition
Multi Valley Mountain Car	State: position and velocity Control: left or right acceleration	The car is inside the narrow central target band
StandUp	State: two joint angles Control: four torque pairs	Both links are inside the upright target region

A.1.1 Multi Valley Mountain Car

The state is $s = (x, \dot{x})$, with $x \in [-0.99, 0.99]$ and $\dot{x} \in [-0.07, 0.07]$. The terrain profile is

$$y(x) = \tfrac{1}{10}\left(\cos(2\pi x) + 2\cos(4\pi x) - \log(1 - x^2)\right). \tag{24}$$

The two actions accelerate the car to the left or right. With $a \in \{0, 1\}$, $\Delta t = 0.1$, agent force $F = 0.001$, and gravity $g = 0.0025$, one simulator step is

$$\dot{x}_{t+1} = \mathrm{clip}(\dot{x}_t + [(2a-1)F - g y'(x_t)]\Delta t)\,, \qquad x_{t+1} = \mathrm{clip}(x_t + \dot{x}_{t+1}\Delta t)\,, \tag{25}$$

where each component is clipped to its state bound. The sign of x_0 is sampled uniformly, $|x_0| \sim \mathrm{Unif}[0.67, 0.77]$, and $\dot{x}_0 \sim \mathrm{Unif}[-0.01, 0.01]$. The reward is one for $x_{t+1} \in [-0.05, 0.05]$ and zero otherwise.

A.1.2 StandUp

The state is $s = (\varphi_1, \varphi_2)$, where φ_1 is the angle between the ground and the first link and φ_2 is the angle between the two links. The four actions are the torque pairs

$$\mathcal{A} = \{[-\tau, -\tau], [-\tau, \tau], [\tau, -\tau], [\tau, \tau]\}. \tag{26}$$

The system is overdamped, with first order dynamics

$$\dot{\varphi}_1 = \tau_1 - \tau_2 - g\cos\varphi_1, \tag{27}$$

$$\dot{\varphi}_2 = \tau_2 - g\cos(\varphi_1 + \varphi_2). \tag{28}$$

We use $\Delta t = 0.1$, $g = 0.025$, torque magnitude $\tau = 2g = 0.05$, and reward half width $\kappa = \pi/24$. For action $a \in \{0, 1, 2, 3\}$,

$$\tau_1(a) = (2\lfloor a/2 \rfloor - 1)\,\tau, \qquad \tau_2(a) = (2(a \bmod 2) - 1)\,\tau. \tag{29}$$

The rates are integrated by one explicit Euler step. The valid state region is

$$0 < \varphi_1 \le \pi, \qquad -\pi < \varphi_2 \le \pi, \qquad 0 < \varphi_1 + \tfrac{1}{2}\varphi_2 \le \pi. \tag{30}$$

The first angle is clipped to $[0, \pi]$. With

$$\ell(\varphi_1) = \max(-2\varphi_1, -\pi), \qquad u(\varphi_1) = \min(2(\pi - \varphi_1), \pi), \tag{31}$$

the second angle is clipped to $[\ell(\varphi_1), u(\varphi_1)]$ using the clipped first angle. Boundary states produced by clipping are retained by the simulator. Reward is one when the successor satisfies $|\varphi_1' - \pi/2| \le \kappa$ and $|\varphi_2'| \le \kappa$, and zero otherwise.

The initial distribution is an equal mixture of two uniform boxes. The right box is $\varphi_1 \in [0, \kappa)$, $\varphi_2 \in [0, \kappa)$. The left box is $\varphi_1 \in (\pi - \kappa, \pi]$, $\varphi_2 \in (-\kappa, 0]$.

A.2 Proposal distributions

The proposal distribution is fixed throughout training and covers the full valid state domain. For Multi Valley Mountain Car, m is uniform on $[-0.99, 0.99] \times [-0.07, 0.07]$, a rectangle of area 0.2772. For StandUp, rejection sampling gives a uniform proposal on

$$0 < \varphi_1 \le \pi, \qquad -\pi < \varphi_2 \le \pi, \qquad 0 < \varphi_1 + \tfrac{1}{2}\varphi_2 \le \pi. \tag{32}$$

The domain has analytic area $3\pi^2/2$. For the ratio to density conversion, the implementation uses a numerical estimate of this area stored with the checkpoint.

A.3 Compared methods

Table 2: Properties of the learned methods. "Trajectories" refers to sequential data collected during training. All methods use trajectories for evaluation. The PPO row covers both the original reward and task specific shaped reward variants.

Method	Training trajectories	Analytical dynamics	Proposal states	Learned density	Critic
Standard SAC	yes	no	no	no	temporal difference
PPO	yes	no	no	no	temporal difference
A2C	yes	no	no	no	temporal difference
Model Based URL	no	yes	yes	yes (PDE residual)	PDE residual
MF URL	no	no	yes	yes (saddle point)	saddle point
MF URL SAC	no	no	yes	yes (saddle point)	temporal difference

The return curves for MF URL, Model Based URL, SAC, and PPO with shaped rewards use ten training seeds per task. The critic diagnostic compares ten runs of each MF URL critic per task. Each method has a nominal budget of $1{,}200{,}000$ in its native training counter. For trajectory methods this counter is an environment transition. For the umbrella methods it is an outer update. Standard SAC uses a ReLU policy with two hidden layers and twin-Q networks of width 256, a replay capacity

of $1{,}200{,}000$, a batch of $10{,}000$, one update per transition after $10{,}000$ initial random transitions, learning rate 10^{-5}, entropy coefficient 0.01, and target coefficient 0.005. A2C uses the default Stable Baselines3 MLP policy, 16 environments, rollout length 5, RMSprop learning rate 7×10^{-4}, and no entropy regularization. The PPO experiments are specified separately in Appendix A.7.

Model Based URL uses the implementation of Nuzhin & Brilliantov (2025), two hidden layers of width 128, a batch of $10{,}000$, and entropy coefficient 0.01. For value, density, and policy respectively, the Adam learning rates are $(10^{-5}, 10^{-5}, 10^{-5})$ on Multi Valley Mountain Car and $(10^{-6}, 10^{-7}, 10^{-6})$ on StandUp. The corresponding weight decays are $(10^{-4}, 5 \times 10^{-4}, 5 \times 10^{-6})$ and $(10^{-5}, 5 \times 10^{-4}, 5 \times 10^{-5})$.

A.4 Optimization and update schedule

All MF URL networks are optimized with Adam with moment parameters $(0.9,\ 0.999)$ and numerical constant 10^{-8}, and zero weight decay. Table 3 records the settings used in every reported MF URL run.

Table 3: Optimization settings for MF URL. Each update uses a fresh proposal batch.

Component	Batch size	Updates	Learning rate	Grad.-norm clip
Density ratio ζ_ξ	10,000	5	10^{-5}	1
Density dual ν_ψ	10,000	1	10^{-5}	1
Residual witness h_η	4,096	5	10^{-5}	5
Value V_θ	4,096	1	10^{-5}	5
Advantage A_θ	4,096	1	10^{-5}	5
Policy π_ω	4,096	1	3×10^{-6}	1

The coefficient $\lambda = 0.01$ weights both logarithmic terms of $\hat{r}$ in equation 14, and the actor adds no separate entropy term. Before the logarithm in equation 16, ζ_ξ is clamped below at 10^{-12}. The quantity $\log \pi_\omega$ is computed with a log softmax. The continuous time discount base is $\gamma_c = 0.95$. With $\Delta t = 0.1$, the one step factor is $\gamma_d = \gamma_c^{\Delta t} = 0.9948838031$. Each seed is trained for $1{,}200{,}000$ outer iterations. A density diagnostic without gradients is also run once per outer iteration on a fresh batch covering all actions.

MF URL uses the online successor value $V_\theta(s')$ and no target network. It also uses no Huber or smooth L_1 loss and no residual clipping.

A.5 Training and evaluation protocol

Each MF URL update draws a fresh batch of states from the proposal distribution. For its density and critic updates, every available action is evaluated and the resulting action expectations are computed exactly. Successors are used only in the update that generated them. They are not added to the proposal or extended into trajectories. The actor update draws proposal states without querying the simulator.

Policy performance is the discounted return

$$G = \sum_{t=0}^{T-1} \gamma_d^t \Delta t\, r_t, \tag{33}$$

where $\gamma_d = \gamma_c^{\Delta t}$. The factor Δt converts the reward rate into reward accumulated during one simulator interval, so the return is measured on the simulated time scale rather than per simulator step.

Evaluation is separate from training and contributes no training data. MF URL and Model Based URL are evaluated every $50{,}000$ outer iterations, SAC and A2C every $10{,}000$ environment transitions, and PPO after each rollout of $8 \times 2048 = 16{,}384$ transitions. Evaluations use deterministic greedy actions. Model Based URL uses the same $\Delta t = 0.1$ and $\gamma_c = 0.95$ as MF URL. Each checkpoint uses the same 50 seeded initial states. Episodes begin from $s_0 \sim d_0$ and have a horizon of $T = 1000$

simulator steps in both environments. All methods, including those trained with reward shaping, are evaluated using the original task reward.

The return curves use $N_{\text{seed}} = 10$ independent training seeds per method. For seed i at a recorded checkpoint k, the evaluation mean is

$$R_i(k) = \frac{1}{50} \sum_{j=1}^{50} G_{i,j}(k), \tag{34}$$

Because methods report checkpoints at different counters, we align each seed curve to a common 10,000-step grid and apply a five-point moving average for display, giving the curve $\widetilde{R}_i(k)$. The plotted mean is

$$\bar{R}(k) = \frac{1}{N_{\text{seed}}} \sum_{i=1}^{N_{\text{seed}}} \widetilde{R}_i(k). \tag{35}$$

The shaded band is one sample standard deviation across the smoothed seed curves.

Grid value iteration reference. Grid VI is a known dynamics planning reference used only in the return plots. We discretize the state space and apply value iteration using the exact deterministic transition f. With $r_i^a = r(s_i, a, f(s_i, a))$, the update is

$$V_{k+1}(s_i) = \max_a \left[\Delta t \, r_i^a + \gamma_d \, \mathcal{I}[V_k]\big(f(s_i, a)\big) \right], \tag{36}$$

where $\mathcal{I}$ is bilinear interpolation at an off grid successor. Multi Valley Mountain Car uses a 401×401 grid and StandUp uses an 801×801 grid. We set $\Delta t = 0.1$, $\gamma_d = 0.95^{0.1}$, and stop when the maximum value change is below 2×10^{-6}. For StandUp, nominal grid points outside the valid region are projected to its boundary before transitions and values are evaluated. The greedy grid policy is evaluated from the same 50 initial states, with the same 1,000-step horizon, discount, and original reward as the learned policies. The horizontal oracle line has no uncertainty band because it is one converged planning solution. It uses the environment equations directly and is not a model-free baseline. This planning reference is separate from the Monte Carlo references used in the density and critic diagnostics.

A.6 Network architectures

All MF URL networks receive normalized two dimensional states. Multi Valley Mountain Car uses $(x/0.99, \dot{x}/0.07)$, and StandUp uses $(2\varphi_1/\pi - 1, \varphi_2/\pi)$. The policy is a multilayer perceptron with two hidden layers of width 128 and $\tanh$ activations, followed by a softmax over the discrete actions.

The value, raw advantage, and residual witness networks are SIRENs (Sitzmann et al., 2020) with width 256 and four sine layers. Their first layer and hidden layer frequencies are 30 and 4, respectively. The value head is scalar. The advantage and witness heads contain one output per action. The density dual ν_ψ and ratio ζ_ξ are width-256 SIRENs with six sine layers and frequencies 5 and 1. Both have scalar heads, with a Softplus applied only to ζ_ξ. StandUp has four policy actions and Multi Valley Mountain Car has two. SIREN parameters use first layer and frequency aware hidden layer initialization. The policy uses the default PyTorch linear layer initialization.

MF URL SAC realization. MF URL SAC uses a ReLU policy with two hidden layers and twin action value networks of width 256. Its density ratio and dual networks have width 512, five sine layers, and frequencies 30 and 4. Each outer iteration draws a proposal batch of $10{,}000$, samples one current action uniformly per state, and takes one temporal difference critic update and one policy update. It optimizes the same density shaped objective with action entropy as MF URL, using the same λ and time scaling. The density term enters the reward, and the action entropy enters through the soft target and actor loss rather than through the reward of the sampled current action. With target networks $\bar{Q}_j$, the target and actor loss are

$$y = \Delta t[r - \lambda \log \widehat{d}^{\pi}(s)] + \gamma_d \sum_{a'} \pi(a' \mid s') \big[\min_j \bar{Q}_j(s', a') - \lambda \Delta t \log \pi(a' \mid s') \big], \tag{37}$$

$$\mathcal{L}_\pi^{\text{SAC}} = \mathbb{E}_{s \sim m} \sum_a \pi(a \mid s) \big[\lambda \Delta t \log \pi(a \mid s) - \min_j Q_j(s, a) \big]. \tag{38}$$

The ratio ζ_ξ is clamped below at 10^{-8} before the logarithm. The density update uses a separate batch of $10{,}000$ with five ratio updates and one dual update. All learning rates are 10^{-5}. The critic uses a smooth L_1 loss and a Polyak averaged target network with coefficient 0.005. The Multi Valley Mountain Car density networks use weight decays 10^{-4} and 5×10^{-4} for the dual and ratio networks. The StandUp density networks use no weight decay.

A.7 PPO EXPERIMENTS

We train PPO with either the original task reward or a task specific shaped reward. Both variants use the default Stable Baselines3 MLP policy, eight parallel environments, rollout length 2,048, batch size 256, and GAE parameter 0.95. On Multi Valley Mountain Car, PPO uses ten epochs per update, clipping parameter 0.2, learning rate 3×10^{-4}, and entropy coefficient 0.05. On StandUp, it uses five epochs, clipping parameter 0.1, no entropy bonus, and a learning rate decreasing from 3×10^{-4} to 3×10^{-5}.

The shaped runs keep these settings and change only the training reward. Let $r(s, s')$ denote the original task reward. For Multi Valley Mountain Car, define $E(s) = \frac{1}{2}\dot{x}^2 + y(x)$. The energy shaped reward is

$$r_{\mathrm{PPO}}^{\mathrm{MVMC}}(s, s') = 5r(s, s') + 3 \max\left(0, \frac{E(s') - E(s)}{\Delta t}\right). \tag{39}$$

For StandUp, define $e(s) = ((\varphi_1 - \pi/2)/(\pi/2))^2 + (\varphi_2/\pi)^2$. The goal progress reward is

$$r_{\mathrm{PPO}}^{\mathrm{StandUp}}(s, s') = 5r(s, s') + 3\frac{e(s) - e(s')}{\Delta t}. \tag{40}$$

All PPO policies are evaluated with the original task reward.

A.8 SIMULATOR QUERY ACCOUNTING

One MF URL iteration calls the simulator 13 times: six calls for the density update, one further call for a density check that uses no gradients, and six calls for the critic update. The seven density calls use the density batch of 10,000 states, and the six critic calls use the critic batch of 4,096 states; every call queries a successor for each of the $|\mathcal{A}|$ actions at every state in its batch. One iteration therefore queries the simulator

$$\left[7 \times 10{,}000 \times |\mathcal{A}|\right]_{\mathrm{density}} + \left[6 \times 4{,}096 \times |\mathcal{A}|\right]_{\mathrm{critic}}$$

times: 378,304 on StandUp ($|\mathcal{A}| = 4$) and 189,152 on Multi Valley Mountain Car ($|\mathcal{A}| = 2$).

Model-Based URL instead evaluates the analytical state rate and its divergence once per iteration, on one batch of 10,000 states, and needs no successor queries. Because this is a different kind of computation from a simulator query, we report the two counts separately rather than treat the horizontal axis of Figure 2 as a direct sample-efficiency comparison. These counts exclude evaluation.

A.9 DENSITY DIAGNOSTIC

Figure 3 uses the final checkpoint of all ten training seeds per environment. For each seed, the Monte Carlo reference uses $50{,}000$ stochastic rollouts initialized from d_0 with horizon $T = 1000$; at step k, each state contributes weight γ_d^k to a 1001×1001 state histogram. The learned map evaluates $m(s)\zeta_\xi(s)$ on the same grid. For Figure 3 and for the correlation and total-variation numbers below, we normalize the reference and the learned map separately to unit mass, so those comparisons test occupancy shape and not absolute calibration.

As a direct calibration check, we also integrate each seed's learned map before this normalization. Its mass is 0.934 ± 0.007 on Multi-Valley Mountain Car and 0.863 ± 0.131 on StandUp (mean $\pm$ sample standard deviation over ten seeds; the target mass is 1). This check only reads off the density already learned during training; it changes the diagnostic, not the training procedure.

For each seed, we compute the Pearson correlation between the two normalized maps p and $\hat{p}$ and their total variation distance $\mathrm{TV}(p, \hat{p}) = \frac{1}{2}\sum_i |p_i - \hat{p}_i|$. Table 4 reports the mean and sample standard deviation across seeds. The maps shown in Figure 3 average the ten normalized maps.

Table 4: Agreement between learned and Monte Carlo reference occupancy maps at the end of training (mean ± sample standard deviation over ten seeds).

Environment	Pearson correlation ↑	Total variation ↓
Multi-Valley Mountain Car	0.987 ± 0.003	0.071 ± 0.006
StandUp	0.743 ± 0.324	0.125 ± 0.119

Simulator clipping can concentrate probability mass on the state boundary. A continuous density network approximates this distribution at finite resolution; the exact ratio identity in Section 3.5 assumes absolute continuity with respect to the proposal measure.

A.10 Critic diagnostic

MF URL SAC and MF URL train on the same proposal states, the same density-shaped, entropy-regularized objective with the same coefficient λ, the same discount, and the same number of outer iterations, and both are evaluated with the protocol above. They differ in two respects. First, the critic objective: MF URL SAC learns twin action-value networks by Polyak-averaged temporal-difference regression, while MF URL learns the saddle-point critic of Section 4.3, with correspondingly different network sizes (Appendix A.6). Second, how the current action is averaged: MF URL SAC samples one action from the uniform reference distribution per proposal state, while MF URL sums over all actions; both target the same expectation, but the sum has no sampling variance and costs more simulator queries. At the end of training, the policy, density estimator, and critic of each run are frozen, and a separate Monte Carlo reference is generated for every seed from its own frozen policy and density estimate, since the critic objective also shapes the policy that is learned.

For every valid state on a fixed grid, the Monte Carlo reference averages N_{MC} stochastic rollouts of horizon $T = 1000$:

$$\widehat{V}_{\mathrm{MC}}(s) = \frac{1}{N_{\mathrm{MC}}} \sum_{j=1}^{N_{\mathrm{MC}}} \sum_{k=0}^{T-1} \gamma_d^k \, \Delta t \Big[r_k^{(j)} - \lambda \log \widehat{d}^{\pi}\big(s_k^{(j)}\big) + \lambda \, H\big[\pi(\cdot \mid s_k^{(j)})\big] \Big], \tag{41}$$

where $\gamma_d = \gamma_c^{\Delta t}$ and $\widehat{d}^{\pi}(s) = m(s)\zeta_\xi(s)$. Because the proposal is uniform in both tasks, $\widehat{d}^{\pi}(s) = \zeta_\xi(s)/|\Omega|$ on the valid state domain Ω, using the stored area estimate of Appendix A.2. Actions are sampled from the frozen policy, and $\log \widehat{d}^{\pi}$ is evaluated by bilinear interpolation of a field precomputed on a 2401×2401 grid. We compute the categorical entropy exactly,

$$H[\pi(\cdot \mid s)] = -\sum_a \pi(a \mid s) \log \pi(a \mid s), \tag{42}$$

rather than sampling $-\log \pi(a \mid s)$, which removes the sampling noise this term would otherwise add at each visited state. The ten-seed summaries use $N_{\mathrm{MC}} = 16$, from two independent groups of eight rollouts; the value maps below use $N_{\mathrm{MC}} = 64$, from two groups of 32. The reference is truncated at T without a bootstrap term and so approximates the infinite-horizon value the critic targets.

MF URL SAC has no separate state-value head, so its prediction comes from the policy and the twin action-value networks,

$$V_{\mathrm{SAC}}(s) = \sum_a \pi(a \mid s) \Big[\min_{i \in \{1,2\}} Q_i(s,a) - \lambda \, \Delta t \log \pi(a \mid s) \Big]. \tag{43}$$

The MF URL critic uses the centered parameterization

$$Q_\theta(s,a) = V_\theta(s) + A_\theta(s,a) - \sum_b \pi(b \mid s) A_\theta(s,b),$$

whose policy average is $V_\theta(s)$, so the value head is used directly.

We report RMSE, signed bias, and Pearson correlation between each critic's prediction and its Monte Carlo reference. Bias is the critic prediction minus the reference, and normalized RMSE is RMSE

Table 5: Normalized RMSE of each critic against its Monte Carlo reference under the policy learned in the same run (mean $\pm$ sample standard deviation over ten seeds). Lower is better.

Environment	MF URL	MF URL SAC
Multi-Valley Mountain Car	0.404 ± 0.030	1.603 ± 0.103
StandUp	0.131 ± 0.042	1.212 ± 0.908

divided by the reference's standard deviation. Table 5 summarizes normalized RMSE across ten seeds.

The ten-seed summaries use 1,189 states on a 41×29 grid for Multi-Valley Mountain Car and 1,261 valid points on a 41×41 grid for StandUp. The value maps below use a denser 121×85 grid for Multi-Valley Mountain Car and 11,347 valid points from a 123×123 grid for StandUp, and each shows the seed whose RMSE is closest to the ten-seed median for that method; it is not the best seed. Ties are resolved by the smaller seed identifier. Each map shows one run, not an average across seeds.

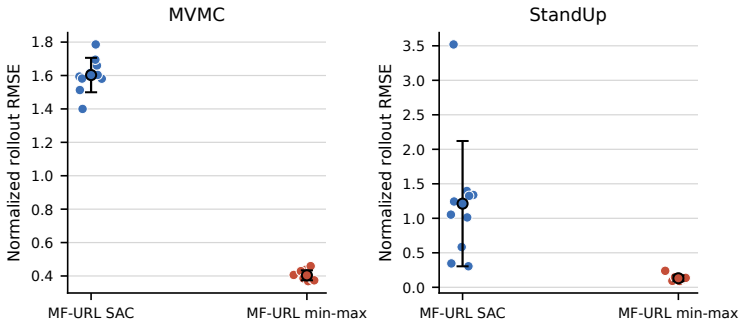


Figure 4: Critic error under each method's learned policy and density estimate. Normalized RMSE is RMSE divided by the standard deviation of the corresponding Monte Carlo reference. Each colored point is one training seed; black markers show the mean across ten seeds and whiskers show one sample standard deviation. Each reference averages $N_{\mathrm{MC}} = 16$ rollouts per state with horizon $T = 1000$.

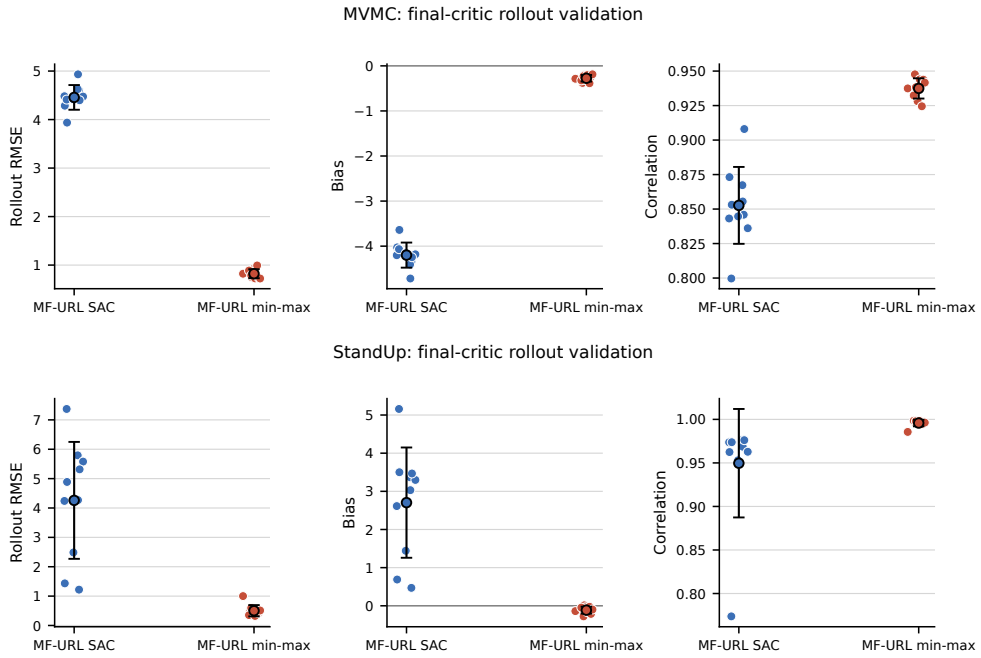


Figure 5: Critic accuracy across ten seeds on Multi-Valley Mountain Car (top) and StandUp (bottom). Each colored point is one seed; black markers show the mean and whiskers show one sample standard deviation. Bias is the critic prediction minus the Monte Carlo reference.

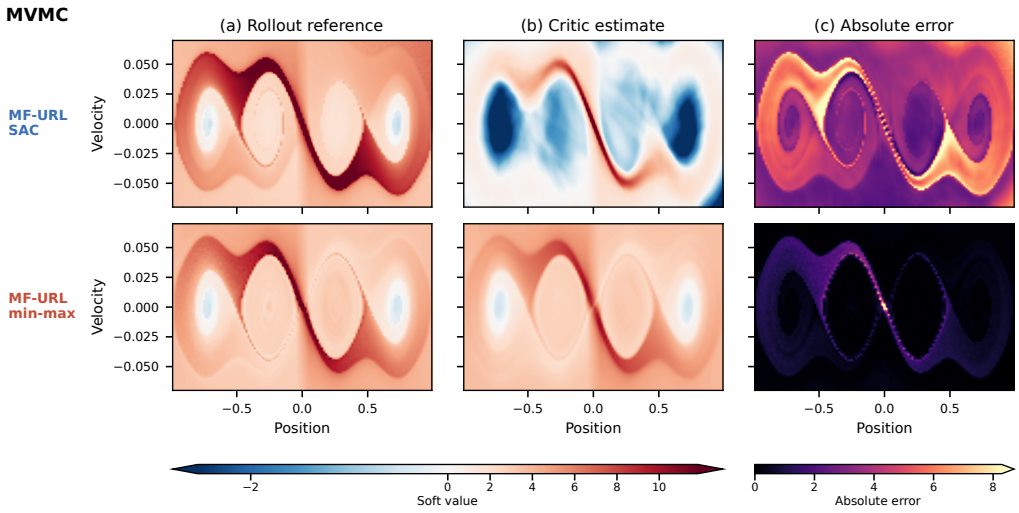


Figure 6: Critic predictions and Monte Carlo references on Multi-Valley Mountain Car. Rows show MF URL SAC and MF URL; columns show the Monte Carlo reference, critic prediction, and pointwise absolute error, for the seed closest to the ten-seed median RMSE of each method. Reference and critic panels share one value scale, and the error panels share a separate scale. Each reference uses $N_{\mathrm{MC}} = 64$ rollouts per state and horizon $T = 1000$.

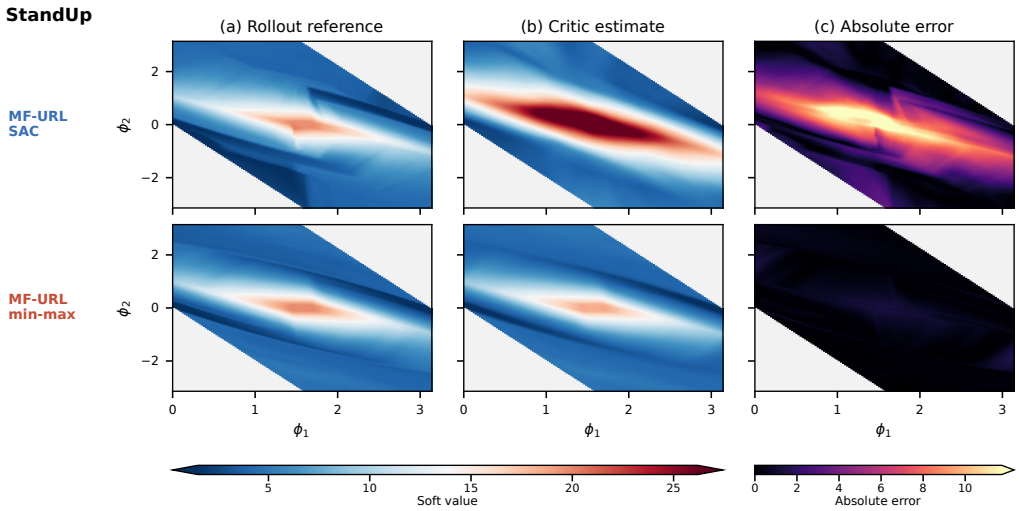


Figure 7: Critic predictions and Monte Carlo references on StandUp under the protocol of Figure 6. Reference and critic panels share one value scale, and the error panels share a separate scale. Blank regions lie outside the valid state domain.

A.11 State density ablation

The ablation removes $-\lambda \log \widehat{d}^{\pi}(s)$ from $\hat{r}$ and keeps $-\lambda \log \pi(a \mid s)$ with $\lambda = 0.01$. The density estimator is then unused, so density updates and the density diagnostic are skipped. The critic, witness, and actor use the batches, learning rates, update counts, and entropy coefficient of Appendix A.4. Both conditions therefore have the same outer iteration budget and the same critic and policy updates per iteration. The ablated condition makes 49,152 simulator queries per outer iteration instead of 189,152, so the comparison is in outer iterations, not simulator queries or elapsed time. Each condition uses five independent training seeds on Multi Valley Mountain Car. Evaluation uses 50 deterministic episodes with horizon $T = 1000$ and the original sparse reward. Curves in Figure 8 use a moving average over five evaluations. Table 6 reports the number of outer iterations needed to reach two return thresholds. For each seed, we record the first evaluation at which its return reaches the threshold and then average over seeds.

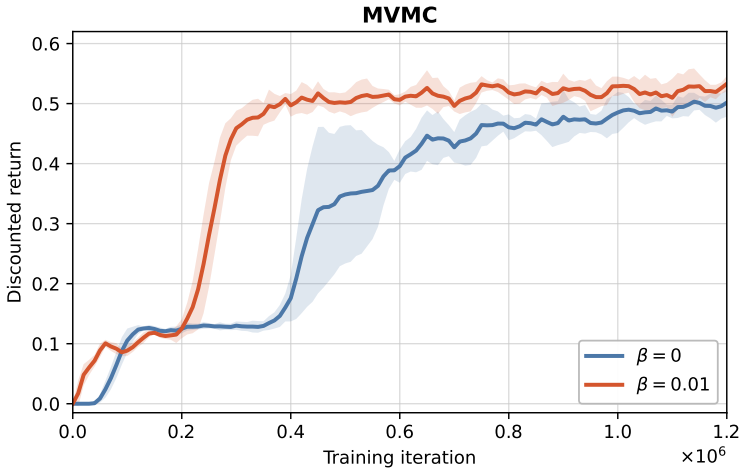


Figure 8: Effect of the state density term on Multi Valley Mountain Car. The ablation removes $-\lambda \log \widehat{d}^{\pi}(s)$ from the shaped reward and keeps the action entropy term. The plot uses β for the state density coefficient: $\beta = 0.01$ is the full method and $\beta = 0$ is the ablation. Lines show the mean over five training seeds, and shaded bands show one sample standard deviation, with smoothing over five evaluations. Evaluation uses 50 deterministic episodes of 1,000 steps with the original sparse reward.

Table 6: Outer iterations at first attainment of each return threshold, averaged over five seeds, for the state density ablation on Multi Valley Mountain Car.

Method	Outer iterations to $G \geq 0.40$	Outer iterations to $G \geq 0.48$
MF URL	254,000	284,000
MF URL without state density	456,000	658,000

B Background derivations

B.1 Occupancy identities

For any integrable f, equation 3 gives $\mathbb{E}_{s\sim d^{\pi}}[f(s)] = (1-\gamma)\sum_{t\geq 0}\gamma^{t}\,\mathbb{E}[f(s_t)]$. With $f(s) = \mathbb{E}_{a\sim\pi}[r(s,a)]$, this yields $J(\pi) = (1-\gamma)^{-1}\mathbb{E}_{s\sim d^{\pi},\,a\sim\pi}[r(s,a)]$. The state densities satisfy $p^{\pi}_{t+1}(s) = \int p^{\pi}_{t}(s')\,\mathbb{E}_{a\sim\pi(\cdot\mid s')}[P(s \mid s', a)]\,\mathrm{d}s'$. Multiplying by $(1-\gamma)\gamma^{t+1}$, summing over $t \geq 0$, and adding $(1-\gamma)d_0$ gives equation 5. Its right hand side is a γ-contraction in total variation, so the solution is unique.

For a bounded test function ν, integrating equation 5 against ν gives $\gamma\,\mathbb{E}_{s\sim d^\pi,\,a\sim\pi,\,s'\sim P}[\nu(s')] = \mathbb{E}_{d^\pi}[\nu] - (1-\gamma)\mathbb{E}_{d_0}[\nu]$, and therefore

$$\mathbb{E}_{s\sim d^\pi,\,a\sim\pi,\,s'\sim P}\big[\nu(s) - \gamma\nu(s')\big] = (1-\gamma)\,\mathbb{E}_{s_0\sim d_0}[\nu(s_0)]. \tag{44}$$

B.2 Fenchel representation and sample averages

For $x \in \mathbb{R}$, $\frac{1}{2}x^2 = \max_{h\in\mathbb{R}}(xh - \frac{1}{2}h^2)$, with maximizer $h = x$. Applying this with $x = \mathbb{E}[\delta_Q \mid s,a]$ at each (s,a) writes equation 11 as $\mathbb{E}_{(s,a)\sim\mu}\big[\max_h \big(\mathbb{E}[\delta_Q \mid s,a]\,h - \frac{1}{2}h^2\big)\big]$. The pointwise maximizer $h^*(s,a) = \mathbb{E}[\delta_Q \mid s,a]$ is measurable and square integrable under μ, so by the interchangeability principle (Shapiro et al., 2009) the expectation of the pointwise maximum equals the maximum of the expectation over functions $h \in L^2(\mu)$. Because $h(s,a)$ is fixed given (s,a), the tower property gives $\mathbb{E}_{(s,a)\sim\mu}\big[\mathbb{E}[\delta_Q \mid s,a]\,h(s,a)\big] = \mathbb{E}_\mu[\delta_Q\,h(s,a)]$, which yields equation 12.

For fixed (Q,h), the integrand $\delta_Q h(s,a) - \frac{1}{2}h(s,a)^2$ depends on a single draw (s,a,s',a'). For independent tuples $\{(s_i,a_i,s_i',a_i')\}_{i=1}^B$, the average $\frac{1}{B}\sum_i\big[\delta_{Q,i}\,h(s_i,a_i) - \frac{1}{2}h(s_i,a_i)^2\big]$ is an unbiased estimate of the objective and, under standard conditions that allow differentiation under the expectation, its gradient with respect to the parameters of Q or h is an unbiased estimate of the corresponding partial gradient. The direct estimate $\frac{1}{B}\sum_i \frac{1}{2}\delta_{Q,i}^2$ instead converges to equation 11 plus $\frac{1}{2}\mathbb{E}_{(s,a)\sim\mu}[\mathrm{Var}(\delta_Q \mid s,a)]$.

If Q^π belongs to the function class, $(Q^\pi, 0)$ is a saddle point of equation 12: equation 2 gives $\mathbb{E}[\delta_{Q^\pi} \mid s,a] = 0$, so $h^* = 0$ and the objective attains its minimum value 0. For a compact convex class of Q and $h \in L^2(\mu)$, the objective is affine in Q and strictly concave in h, and Sion's minimax theorem (Sion, 1958) allows the order of $\min$ and $\max$ to be exchanged. These conditions describe idealized function classes and do not hold for neural networks.

B.3 State marginal DualDICE objective

For each s, the scalar problem $\min_w \frac{1}{2}\mu(s)w^2 - d^\pi(s)w$ is minimized by $w = d^\pi(s)/\mu(s)$. Hence

$$\begin{aligned} w^* &= \arg\min_w\ \tfrac{1}{2}\,\mathbb{E}_{s\sim\mu}[w(s)^2] - \mathbb{E}_{s\sim d^\pi}[w(s)], \\ w^* &= d^\pi/\mu, \end{aligned} \tag{45}$$

under the assumptions of Section 3.5 $d^\pi \ll \mu$ and a bounded ratio $w^* = d^\pi/\mu$, as in Nachum et al. (2019). For the logarithmic density reward of equation 14, we additionally assume $d^\pi(s) > 0$ on the support of m, so that $\log d^\pi$ is finite at every proposal state the Softplus parameterization of ζ_ξ (Section 4.2) keeps the learned estimate positive regardless. This objective is the negative Fenchel dual form of the Pearson χ^2 divergence (Nachum et al., 2019), but its second term requires samples from d^π.

Define ν through $w(s) = \nu(s) - \gamma\,\mathbb{E}_{a\sim\pi,\,s'\sim P}[\nu(s') \mid s]$. Since $\gamma < 1$, $\nu = \sum_{k\geq 0}\gamma^k (P^\pi)^k w$ is well defined and bounded for bounded w, where P^π is the one step state transition operator under π. Equation 44 gives $\mathbb{E}_{d^\pi}[w] = (1-\gamma)\mathbb{E}_{d_0}[\nu]$, so equation 45 becomes

$$\min_\nu\ \tfrac{1}{2}\,\mathbb{E}_{s\sim\mu}\big[(\nu(s) - \gamma\,\mathbb{E}[\nu(s') \mid s])^2\big] - (1-\gamma)\,\mathbb{E}_{s_0\sim d_0}[\nu(s_0)], \tag{46}$$

whose optimal residual $\nu^*(s) - \gamma\,\mathbb{E}[\nu^*(s') \mid s]$ equals $d^\pi(s)/\mu(s)$. Applying the Fenchel identity and the interchangeability principle as in Appendix B.2, with dual variable $w(s)$, gives equation 13. The inner maximizer is $w = \nu - \gamma\,\mathbb{E}[\nu(s') \mid s]$, so $w^* = d^\pi/\mu$ at the saddle point, and the unbiasedness argument of Appendix B.2 applies unchanged.

B.4 Saddle points of the two objectives

We consider unrestricted function classes, exact expectations, bounded test functions, and $h, w \in L^2(\mu)$. For a finite signed measure σ on state and action pairs, σK denotes the measure of (s',a') when $(s,a) \sim \sigma$, $s' \sim P(\cdot \mid s,a)$, and $a' \sim \pi(\cdot \mid s')$; for a finite signed measure ρ on states, ρP^π denotes the analogous measure of s'. Neither operator increases total variation.

Critic. Let (Q^*, h^*) be a saddle point of equation 12. The objective is linear in Q, so Q^* minimizes it over unrestricted Q only if its linear term vanishes: for every bounded ϕ,

$$\mathbb{E}_\mu\big[h^*(s,a)\big(\gamma\phi(s',a') - \phi(s,a)\big)\big] = 0. \tag{47}$$

The measure $\sigma = h^*\mu$ is finite because $h^* \in L^2(\mu)$, and this condition reads $\sigma = \gamma\,\sigma K$. Taking total variation gives $\|\sigma\| \le \gamma\|\sigma\|$, so $\sigma = 0$ and $h^* = 0$ almost everywhere under μ. The inner maximization gives $h^*(s,a) = \mathbb{E}[\delta_{Q^*} \mid s, a]$, so $\mathbb{E}[\delta_{Q^*} \mid s, a] = 0$, which is the Bellman equation 2 at almost every (s,a) under μ.

Occupancy ratio. Let (ν^*, w^*) be a saddle point of equation 13. The objective is linear in ν, so for every bounded ϕ,

$$\mathbb{E}_{s\sim\mu,\, a\sim\pi,\, s'\sim P}\big[w^*(s)\big(\phi(s) - \gamma\phi(s')\big)\big] = (1-\gamma)\,\mathbb{E}_{s_0\sim d_0}[\phi(s_0)]. \tag{48}$$

With the finite measure $\rho = w^*\mu$, this reads $\rho = (1-\gamma)d_0 + \gamma\,\rho P^\pi$, which is equation 5. Its right hand side is a γ contraction in total variation (Appendix B.1), so $\rho = d^\pi$ and $w^* = d^\pi/\mu$ almost everywhere under μ.

These statements describe the saddle points of the objectives. They do not show that alternating stochastic gradient updates on neural network parameters reach such a point.

C Derivations for MF URL

C.1 Discretization of the URL objective

Model Based URL maximizes $\int_0^\infty \gamma_c^t \langle \mathbb{E}_{a\sim\pi}[r_u(s,a)]\rangle_{p_t} \mathrm{d}t$ with $r_u = r - \tilde{\alpha}\log(\bar{p}(s)\pi(a \mid s))$ (Section 3.3). Let the simulator advance the state by Δt per step, and let p_k^π be the state distribution after k steps. Substituting $t = k\Delta t$, $\mathrm{d}t \to \Delta t$, $\gamma_c^{k\Delta t} = \gamma_d^k$, and $p_{k\Delta t} \to p_k^\pi$ gives the Riemann sum $\sum_{k\ge0}\gamma_d^k\,\Delta t\,\mathbb{E}[r_u(s_k,a_k)]$. The same substitution in the time averaged density gives

$$\bar{p}_{\Delta t}(s) = |\log\gamma_c|\,\Delta t\sum_{k\ge0}\gamma_d^k p_k^\pi(s) = c_{\Delta t}\,d^\pi(s), \qquad c_{\Delta t} = \tfrac{|\log\gamma_c|\,\Delta t}{1-\gamma_d}, \tag{49}$$

by equation 3 with $\gamma = \gamma_d$. Since $1-\gamma_d = 1 - e^{-|\log\gamma_c|\Delta t} = |\log\gamma_c|\Delta t + O(\Delta t^2)$, $c_{\Delta t} \to 1$ as $\Delta t \to 0$. For $\gamma_c = 0.95$ and $\Delta t = 0.1$, $c_{\Delta t} \approx 1.0026$. Replacing $\bar{p}$ by d^π shifts the per step reward by the constant $\lambda\,\Delta t\log c_{\Delta t}$, which changes J_{MFURL} by a policy independent constant. With $\lambda = \tilde{\alpha}$, this gives equation 14.

C.2 Bellman equation and policy gradient of the discrete objective

For fixed π and d^π, equation 14 is the discounted return of the per step reward $\hat{r}$. Define $Q(s,a) = \mathbb{E}\big[\sum_{k\ge0}\gamma_d^k\hat{r}(s_k,a_k,s_{k+1}) \mid s_0 = s, a_0 = a\big]$ and $V(s) = \mathbb{E}_{a\sim\pi}[Q(s,a)]$. Separating the $k=0$ term and conditioning on $s_1 = s'$ gives

$$\begin{aligned} Q(s,a) &= \mathbb{E}_{s'\sim P(\cdot|s,a)}\Big[\hat{r}(s,a,s') + \gamma_d\,\mathbb{E}\big[\textstyle\sum_{k\ge1}\gamma_d^{k-1}\hat{r}(s_k,a_k,s_{k+1}) \mid s_1 = s'\big]\Big] \\ &= \mathbb{E}_{s'}\big[\hat{r}(s,a,s') + \gamma_d V(s')\big], \end{aligned} \tag{50}$$

which is equation 18.

The reward depends on the policy parameters ω through d^π and $\log\pi_\omega$. Write $f(s,a) = \Delta t\big(\mathbb{E}_{s'}[r(s,a,s')] - \lambda\log d^\pi(s) - \lambda\log\pi_\omega(a \mid s)\big)$, so that $J_{\mathrm{MFURL}} = (1-\gamma_d)^{-1}\mathbb{E}_{s\sim d^\pi,\,a\sim\pi_\omega}[f(s,a)]$ by Appendix B.1.Assuming differentiation under the integral is allowed, this gives the policy gradient equation 4 for f held fixed, plus the term $(1-\gamma_d)^{-1}\mathbb{E}_{d^\pi,\pi_\omega}[\nabla_\omega f]$. This term vanishes, because

$$\begin{aligned} \mathbb{E}_{s\sim d^\pi}[\nabla_\omega \log d^\pi(s)] &= \nabla_\omega \textstyle\int d^\pi(s)\,\mathrm{d}s = 0, \\ \mathbb{E}_{a\sim\pi_\omega}[\nabla_\omega \log\pi_\omega(a \mid s)] &= \nabla_\omega \textstyle\sum_a \pi_\omega(a \mid s) = 0. \end{aligned} \tag{51}$$

Hence $\nabla_\omega J_{\mathrm{MFURL}}$ equals equation 4 with $\gamma = \gamma_d$ and the advantage of $\hat{r}$ computed for fixed d^π, which is the quantity the critic estimates. Nuzhin & Brilliantov (2025, Appendix A.2) give the same argument for the continuous time objective.

C.3 POLICY CENTERED VALUE AND ADVANTAGE PARAMETERIZATION

Let Q be any function on $\mathcal{S} \times \mathcal{A}$. Setting $V(s) = \mathbb{E}_{a\sim\pi}[Q(s,a)]$ and $A = Q$ in equation 19 gives $V + A - \mathbb{E}_\pi[A] = Q$, so the parameterization represents every action value function and the solution of equation 21 is the same as for an unconstrained Q. Conversely, every represented Q_θ satisfies $\mathbb{E}_\pi[Q_\theta] = V_\theta$. The value head is therefore identified as the policy average of Q_θ, the successor term $\mathbb{E}_{a'\sim\pi}[Q_\theta(s', a')]$ equals $V_\theta(s')$, and adding any function of s to A_θ leaves the centered advantage, and hence Q_θ, unchanged.