Long-Horizon Model-Based Offline Reinforcement Learning Without Conservatism

Tianwei Ni
Université de Montréal
Mila - Quebec AI Institute

Esther Derman
Université de Montréal
Mila - Quebec AI Institute

Vineet Jain
McGill University
Mila - Quebec AI Institute

Vincent Taboga
Université de Montréal
Mila - Quebec AI Institute

Siamak Ravanbakhsh
McGill University
Mila - Quebec AI Institute

Pierre-Luc Bacon
Université de Montréal
Mila - Quebec AI Institute

Abstract

Popular offline reinforcement learning (RL) methods rely on *conservatism*, either by penalizing out-of-dataset actions or by restricting rollout horizons. In this work, we question the universality of this principle and instead revisit a complementary one: a *Bayesian* perspective. Rather than enforcing conservatism, the Bayesian approach tackles epistemic uncertainty in offline data by modeling a posterior distribution over plausible world models and training a history-dependent agent to maximize expected rewards, enabling test-time generalization. We first illustrate, in a bandit setting, that Bayesianism excels on low-quality datasets where conservatism fails. We then scale this principle to realistic tasks, identifying key design choices, such as layer normalization in the world model and adaptive long-horizon planning, that mitigate compounding error and value overestimation. These yield our practical algorithm, NEUBAY, grounded in the NEUtral BAYesian principle. On D4RL and NeoRL benchmarks, NEUBAY generally matches or surpasses leading conservative algorithms, achieving new state-of-the-art on 7 datasets. Notably, it succeeds with rollout horizons of several hundred steps, challenging prevailing belief. Finally, we characterize datasets by quality and coverage, showing when NEUBAY is preferable to conservative methods. Together, we argue NEUBAY lays the foundation for a new direction in offline and model-based RL.

1 Introduction

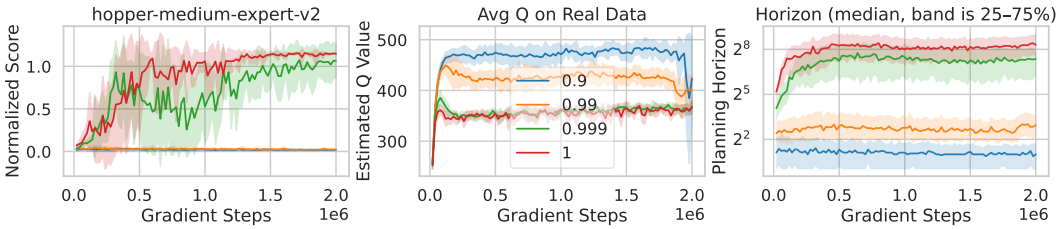

Figure 1: Our algorithm NEUBAY's result on a D4RL dataset. From left to right: normalized score on the real environment, estimated Q-value on the offline dataset, and **rollout horizon statistics** over 100 training rollouts (median with interquartile range). Here we vary the uncertainty quantile $\zeta \in \{0.9, 0.99, 0.999, 1.0\}$ for the rollout truncation threshold, **without using conservatism**.

Reinforcement learning (RL) often assumes direct interaction with the environment, which we refer to as online RL (Sutton and Barto, 2018). While successful in simulation, deploying it in real-world settings such as robotics or healthcare is limited by time-consuming and risky data collection (Dulac-

2nd Workshop on Aligning Reinforcement Learning Experimentalists and Theorists (ARLET 2025).

Arnold et al., 2021). A more practical alternative is offline RL (Lange et al., 2012), which learns from pre-collected datasets (e.g., from human demonstrations or prior agents) without further environment access (Levine et al., 2020; Fu et al., 2020; Gulcehre et al., 2020). This decoupling enables safe, scalable training and the potential to outperform the behavior policies that produced the data.

Most offline RL algorithms adopt a conservative principle by penalizing the policy and value function on out-of-dataset state-action pairs (Levine et al., 2020; Prudencio et al., 2023) and using short rollout horizons (Lu et al., 2021). In theory, these algorithms enforce robustness, either strict (Jin et al., 2021; Uehara and Sun, 2022) or soft (Zhang et al., 2024b), over the uncertainty set of possible MDPs consistent with the dataset. The trade-off is clear: conservatism reduces value overestimation and unsafe extrapolation, but can also suppress average-case performance and limit generalization, since policies are discouraged from exploring potentially high-reward but underrepresented actions.

Bayesian RL optimizes average-case performance under epistemic uncertainty (Duff, 2002). Its application to offline RL was pioneered by Ghosh et al. (2022), who formalized the problem as an *epistemic POMDP*, where partial observability arises from limited coverage. This formulation enables test-time generalization through history-dependent Bayes-optimal policies. In this work, we first revisit and extend this Bayesian principle through the lens of *data quality*. Using a two-armed bandit with skewed data, we show that conservative algorithms, with sufficient uncertainty penalty, are guaranteed to commit to the seen arm regardless of test-time conditions. In contrast, Bayesian algorithms can adaptively explore and commit to the better arm at test time, a clear advantage in low-quality datasets.

However, scaling the Bayesian principle to realistic tasks is challenging, as it requires solving an *approximate* epistemic POMDP. We identify *three key challenges*: (1) compounding error in world models, where inaccuracies grow rapidly with horizon (Lambert et al., 2022); (2) value overestimation, since Bayesian RL lacks explicit penalties for out-of-dataset actions and is vulnerable to extrapolation error (Fujimoto et al., 2019); and (3) training agents with long-term memory to enable test-time adaptation (Ni et al., 2023). Each aligns with a major research area: model-based RL, offline RL, and partially observable RL. This helps explain why prior Bayesian-inspired algorithms often *reintroduce* conservatism, through uncertainty penalties, short horizons, or reductions to model-free RL (Ghosh et al., 2022; Chen et al., 2021c; Jeong et al., 2023).

We build a practical algorithm, NEUBAY, to address these challenges. First, we show that layer normalization (Ba et al., 2016), effective in model-free RL (Ball et al., 2023), also helps world models mitigate compounding error. Second, we observe that model-generated rewards are less biased than bootstrapped values, allowing Bayesian RL to amortize overestimation risk through long-horizon planning. To keep such rollouts reliable, we truncate them using epistemic uncertainty as a threshold (Frauenknecht et al., 2024), an *alternative* use of uncertainty beyond penalties. Finally, we leverage advances in recurrent RL (Morad et al., 2024; Luo et al., 2024a) to stabilize training and support long-term memory. Together, these components adapt insights from online RL to make Bayesian offline RL practical.

We evaluate NEUBAY on the D4RL (Fu et al., 2020) and NeoRL (Qin et al., 2022) benchmarks, covering 33 datasets. Overall, NEUBAY matches or surpasses leading conservative algorithms and outperforms other Bayesian-inspired methods, establishing new state-of-the-art results on 7 datasets. In realistic tasks, NEUBAY performs best on low-quality datasets and on medium-quality datasets with moderate coverage. Sensitivity and ablation studies confirm the importance of our design choices, with adaptive long-horizon planning, often reaching hundreds of steps, emerging as the key to success (e.g., Fig. 1). Notably, short horizons often fail entirely, challenging dominant belief in model-based RL. These results position NEUBAY as a new foundation for model-based and offline RL from a Bayesian perspective, with future advances in world modeling to push the limits further.

2 Background on Offline RL

In the standard offline RL setting, a static dataset $\mathcal{D}$ is collected by interacting with an MDP $\mathcal{M}^*$, which we refer to as the *true MDP*. Formally, $\mathcal{M}^* = (\mathcal{S}, \mathcal{A}, \gamma, T, f_{\text{term}}, m^*, \rho)$ where $\mathcal{S}$ and $\mathcal{A}$ are state and action spaces, $\gamma \in (0, 1)$ is the discount factor, $T \in \mathbb{N}$ is the maximum episode length, and $f_{\text{term}} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \{0, 1\}$ is the terminal function. These components are assumed to be **known** (Puterman, 2014; Yu et al., 2020). The *joint reward-transition function* is $m^* : \mathcal{S} \times \mathcal{A} \to \Delta([-r_{\max}, r_{\max}] \times \mathcal{S})$, consisting of reward function $R^* : \mathcal{S} \times \mathcal{A} \to \Delta([-r_{\max}, r_{\max}])$ (here, $r_{\max}$ is a positive constant) and dynamics $P^* : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, both **unknown** to the agent and learned in model-based methods. The initial state distribution $\rho \in \Delta(\mathcal{S})$ is also unknown, but we do not model it explicitly since initial states can be directly sampled from $\mathcal{D}$ (Janner et al., 2019).

The static dataset of trajectories[1] $\mathcal{D} = \{\tau^i\}_{i=1}^{\text{num\_traj}}$ is collected by an unknown (possibly) history-dependent behavior policy $\pi_\beta : \mathcal{H}_t \to \Delta(\mathcal{A})$, where $\mathcal{H}_t$ is the space of state-action-reward sequences up to timestep t. Define $h_t = (s_{0:t}, a_{0:t-1}, r_{1:t}) \in \mathcal{H}_t$ for $t \geq 1$ with the convention that $h_0 = s_0$. Each trajectory $\tau = (s_0, a_0, r_1, d_1, s_1, a_1, \dots)$ is generated by: $s_0 \sim \rho, a_t \sim \pi_\beta(h_t), (r_{t+1}, s_{t+1}) \sim m^*(s_t, a_t), d_{t+1} = f_{\text{term}}(s_t, a_t, s_{t+1})$. A trajectory ends either when $d_t = 1$ (termination) or when $t = T$ (truncation). We stress this distinction: *termination* implies absorbing states with zero future rewards, whereas *truncation* preserves continuation and thus allows bootstrapping.

The ideal objective in offline RL is to find a possibly history-dependent policy $\pi : \mathcal{H}_t \to \Delta(\mathcal{A})$ that maximizes the expected discounted return under the true MDP $\mathcal{M}^*$:

$$\max_\pi J(\pi, m^*) := \mathbb{E}\left[\sum\nolimits_{t=0}^{\infty} \gamma^t r_{t+1} \,\middle|\, s_0 \sim \rho, a_t \sim \pi(h_t), (r_{t+1}, s_{t+1}) \sim m^*(s_t, a_t)\right]. \tag{1}$$

The defining constraint of offline RL is that the agent cannot interact with m^*, making it intractable to direct optimize Eq. 1. This leads to the following discussion about epistemic uncertainty on m^*.

Empirical model and epistemic uncertainty. From the agent's view, knowledge of m^* is well-defined only on the state-action support of the dataset $\mathcal{D}$: $\text{supp}_{\mathcal{S}\times\mathcal{A}}(\mathcal{D}) := \{(s, a) \mid (s, a, r, s') \in \mathcal{D}\}$. Let $\mathfrak{M}_{\text{in}}$ denote a model class whose domain is restricted to $\text{supp}_{\mathcal{S}\times\mathcal{A}}(\mathcal{D})$. The *empirical model* (Fujimoto et al., 2019) is then obtained by maximum likelihood estimation (MLE):

$$m_\mathcal{D} = \text{argmax}_{m\in\mathfrak{M}_{\text{in}}} \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}[\log m(r, s' \mid s, a)]. \tag{2}$$

Thus, $m_\mathcal{D}$ is uniquely determined in-support by empirical frequencies, but remains *undefined* for $(s, a) \notin \text{supp}_{\mathcal{S}\times\mathcal{A}}(\mathcal{D})$, giving rise to substantial epistemic uncertainty (Gal, 2016). A common way to formalize this uncertainty is through an *uncertainty set* $\mathfrak{M}_\mathcal{D}$, the set of plausible models on $\mathcal{S} \times \mathcal{A}$ that agree with $\mathcal{D}$ on $\text{supp}_{\mathcal{S}\times\mathcal{A}}(\mathcal{D})$ (i.e, close to $m_\mathcal{D}$). Offline policy learning then incorporates $\mathfrak{M}_\mathcal{D}$ into optimization, typically via two paradigms: *conservatism* (or *pessimism*), which optimizes against worst-case models in $\mathfrak{M}_\mathcal{D}$, or *Bayesianism*, which leverages a posterior distribution over $\mathfrak{M}_\mathcal{D}$.

Conservative principle: robust MDPs. Conservative RL methods commonly optimize return under the *worst* model of the uncertainty set $\mathfrak{M}_\mathcal{D}$ (Uehara and Sun, 2022):

$$\max_\pi J(\pi; \mathfrak{M}_\mathcal{D}) := \max_\pi \min_{m\in\mathfrak{M}_\mathcal{D}} J(\pi, m). \tag{3}$$

This robust MDP formulation (Wiesemann et al., 2013) covers both model-free (Fujimoto et al., 2019) and model-based algorithms (Rigter et al., 2022; Yu et al., 2020), differing only in the choice of uncertainty set $\mathfrak{M}_\mathcal{D}$ and the degree of robustness (Zhang et al., 2024b). We provide formal connections between conservatism and robustness in Sec. B.

Bayesian principle: epistemic POMDPs. Bayesianism instead treats the true model m^* as a random variable (Ghavamzadeh et al., 2015), maintaining a posterior distribution $\mathbb{P}_\mathcal{D}(m)$ over plausible models. Bayesian offline RL (Ghosh et al., 2022; Uehara and Sun, 2022; Fellows et al., 2025) then optimizes the *expected* return under this posterior, known as *ambiguity-neutrality* (Ellsberg, 1961):

$$\max_\pi J(\pi; \mathbb{P}_\mathcal{D}) := \max_\pi \mathbb{E}_{m\sim\mathbb{P}_\mathcal{D}}[J(\pi, m)]. \tag{4}$$

The posterior naturally induces an uncertainty set via its support. If $\mathfrak{M}_\mathcal{D} = \text{supp}(\mathbb{P}_\mathcal{D})$, the Bayesian objective (Eq. 4) is less pessimistic than the conservative one (Eq. 3), since for all policies π, $J(\pi; \mathbb{P}_\mathcal{D}) \geq J(\pi; \mathfrak{M}_\mathcal{D})$. By averaging over $\mathbb{P}_\mathcal{D}$, the Bayesian approach places more weight on likely models and less on unlikely ones, effectively yielding a *soft-robust* optimization (Derman et al., 2018). Eq. 4 is equivalent to solving a Bayes-adaptive MDP (BAMDP) (Duff, 2002), also known as an epistemic POMDP; see Ghosh et al. (2022) and Sec. C for a connection with POMDPs.

3 Illustrative Example: When Is Bayesianism Better?

The bandit dataset. To illustrate *when* the Bayesian principle is preferred over conservatism, we construct a skewed bandit dataset. We consider a sequential two-armed bandit with $\mathcal{A} = \{0, 1\}$. Each arm $a \in \mathcal{A}$ yields a Bernoulli reward $r \sim R^*(a) = \mathcal{B}(p_a^*)$, so the true MDP m^* is specified by reward parameters $p^* \in [0, 1] \times [0, 1]$. We fix $p_0^* = 0.5$ in our setup. The optimal policy π^* in m^* is deterministic and memoryless, always choosing $\text{argmax}_{a\in\mathcal{A}} p_a^*$. To highlight out-of-support challenges, we construct a dataset that only covers arm 0. Specifically, the dataset $\mathcal{D} = \{(a_{0:T-1}^i, r_{1:T}^i)\}_i$ is collected by a deterministic behavior policy $\pi_\beta(a) = \mathbb{1}(a = 0)$. Thus, for each $t < T = 100$, $a_t = 0$, $r_{t+1} \sim \mathcal{B}(p_{a_t}^*)$, and $\mathcal{D}$ contains no data on $a = 1$.

[1]While offline RL datasets are often expressed as transition tuples, the trajectory format is available in common benchmarks (Fu et al., 2020) and required by sequence-based algorithms (Chen et al., 2021a).

This skewed dataset $\mathcal{D}$ leaves the true reward parameter p_1^* for arm 1 *completely* unobserved, inducing substantial epistemic uncertainty on p_1^*. Theoretically, under an uninformative prior for Bernoulli rewards, this uncertainty corresponds to the entropy of $\mathrm{Unif}[0, 1]$. In practice, we approximate the posterior by fitting a reward-model ensemble using Gaussian outputs (Chua et al., 2018). As shown in Fig. 2, after pretraining on $\mathcal{D}$, the ensemble predictions concentrate around the observed arm 0 but sharply disagree on the unobserved arm 1, with estimated uncertainty about $10\times$ larger than on arm 0. Next, we test the Bayesian method from Eq. 4, where planning begins at $t = 0$ and truncates at $t = T$. Dynamics are not modeled due to the bandit structure, and there is no concern about compounding error.

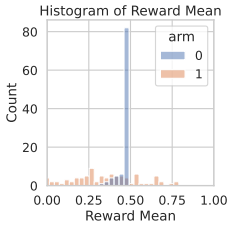

Figure 2: Histogram of estimated reward means p_0, p_1 across ensemble members.

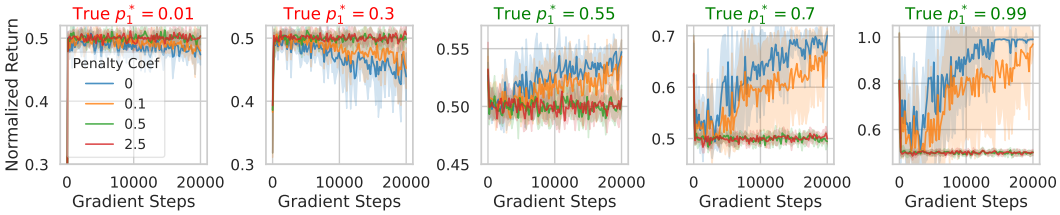

Figure 3: Average return (normalized by T) on **test-time bandits** with $p_1^* \in \{0.01, 0.3, 0.55, 0.7, 0.99\}$. Since the observed arm has $p_0^* = 0.5$, cases with $p_1^* < 0.5$ are *worse* and those with $p_1^* > 0.5$ are *better*.

Uncertainty penalty hurts generalization, but Bayesian agents adapt to worse *and* better cases. We study conservatism by adding an uncertainty penalty to the Bayesian objective. Such a design is common in empirical algorithms (Yu et al., 2020), and can be viewed as risk-sensitive Bayesian RL (Rigter et al., 2021) or a soft robust MDP (Zhang et al., 2024b) (see Sec. B). Since the estimated uncertainty on arm 1 is much larger than arm 0, a sufficiently large penalty coefficient λ forces the resulting algorithm to always select arm 0, regardless of the test-time environment.

In contrast, the Bayesian agent ($\lambda = 0$) trains on a posterior $\mathbb{P}_{\mathcal{D}}(p_1)$ that spans both worse and better values relative to p_0^* (see Fig. 2). It adopts a history-dependent policy that explores the unseen arm at test time before committing, similarly to Bayesian meta-RL (Zintgraf et al., 2020). Thus, as shown in Fig. 3, the Bayesian agent's return is slightly worse than conservatism when $p_1^* < p_0^* = 0.5$, because it briefly explores arm 1. When $p_1^* > 0.5$, Bayesianism is a clear win, as it identifies and exploits the better unseen action, whereas heavy conservatism remains stuck on arm 0. We can thus deduce our main insight: *Bayesianism excels with **low-quality datasets** lack of optimal actions, while remaining competitive on **high-quality datasets** by test-time adaptation with a cost.*

4 NEUBAY: A Practical Bayesian-Principled Algorithm

The Bayesian objective (Eq. 4) is conceptually simple and, as seen in the bandit example, clearly outperforms conservatism on low-quality data. Extending it to MDPs raises **three challenges**: compounding model error, value overestimation, and instability in training history-dependent agents. We address these with ensembles for posterior approximation (Sec. 4.1), adaptive long-horizon planning (Sec. 4.2), and stabilized recurrent training (Sec. 4.3). Together these yield our practical algorithm, NEUBAY (Sec. 4.4), whose design choices we analyze in Sec. 5.2.

4.1 Controlling Compounding Errors in Epistemic POMDP Modeling

A prerequisite for optimizing Eq. 4 is to approximate the epistemic POMDP $\mathbb{P}_{\mathcal{D}}$ with a learned posterior $\hat{\mathbb{P}}_{\mathcal{D}}$. Following popular model-based RL methods (Chua et al., 2018; Yu et al., 2020), we model the posterior with deep ensembles (Lakshminarayanan et al., 2017), a simple method that balances predictive accuracy with uncertainty quantification (Ovadia et al., 2019; Gustafsson et al., 2020). Our deep ensemble is a set of neural networks $\mathbf{m}_{\boldsymbol{\theta}} = \{m_{\theta^n}\}_{n=1}^N$ where each model outputs a Gaussian distribution over next state s' and reward r: $m_{\theta^n}(s, a) = \mathcal{N}(\mu_{\theta^n}(s, a), \sigma_{\theta^n}(s, a))$, with parameters $\boldsymbol{\theta} = \{\theta^n\}_{n=1}^N$ independently initialized at random. Each m_{θ^n} is trained on the same dataset using the maximum likelihood (MLE) loss and evaluated by mean squared error (MSE) on a shared validation set (Yu et al., 2020). Deep ensembles provide a practical approximation to Bayesian neural networks, as they capture diverse modes of the Bayesian posterior (Fort et al., 2019; Wilson and Izmailov, 2020). The induced posterior is $\hat{\mathbb{P}}_{\mathcal{D}}(m) = \frac{1}{N}\sum_{n=1}^N \mathbb{1}(m = m_{\theta^n})$ forms a

uniform distribution over ensemble members. Epistemic uncertainty can be estimated as the standard deviation of the *mean* predictions (Lakshminarayanan et al., 2017) of the ensemble members: $U_{\boldsymbol{\theta}}(s,a) = \mathrm{std}(\{\mu_{\theta^n}(s,a)\}_{n=1}^{N})$. This simple design enables direct comparison with popular methods, although multi-step prediction (Lin et al., 2025) and richer uncertainty quantification (Qiao et al., 2025) may further improve performance.

Design choice 1: larger ensemble size N for posterior fidelity. While deep ensembles approximate Bayesian neural networks, their epistemic fidelity relies on member diversity. With horizons extending hundreds of steps, compounding errors make faithful posteriors essential, rendering small ensembles (e.g., $N = 5$ in MBPO (Janner et al., 2019)) inadequate.

Design choice 2: layer normalization in the world model to control compounding errors. A fundamental problem with world models is that small prediction errors compound during multi-step planning (Talvitie, 2014). To mitigate this, we apply *layer normalization* (LN) (Ba et al., 2016) to each world model m_{θ^n}. Similar to its role in reducing value overestimation (Ball et al., 2023), LN mitigates compounding model error. To address this, we structure the world model as a delta predictor and apply *layer normalization* (LN) (Ba et al., 2016) to its hidden features. Concretely, the model predicts the next state as $\mathbb{E}[\hat{s}'] = s + \mathbf{W}^\top \mathrm{ReLU}(\mathrm{LN}(\psi(s,a))$, with features $\psi(s,a) \in \mathbb{R}^k$ and output weights $\mathbf{W} \in \mathbb{R}^{k \times |\mathcal{S}|}$. For LN without affine parameters, under ℓ_2 norm:

$$\|\mathbb{E}[\hat{s}'] - s\| \leq \|\mathbf{W}\| \|\mathrm{ReLU}(\mathrm{LN}(\psi(s,a))\| \leq \|\mathbf{W}\| \|\mathrm{LN}(\psi(s,a))\| = \sqrt{k}\|\mathbf{W}\|, \tag{5}$$

using that fact that $\|\mathrm{LN}(x)\| = \sqrt{k}$ for any $x \in \mathbb{R}^k$. Applying the triangle inequality over an H-step imagined trajectory yields a linear compounding bound, $\|\mathbb{E}[\hat{s}_H] - s_0\| \leq H\sqrt{k}\|\mathbf{W}\|$. Therefore, by controlling $\|\mathbb{E}[\hat{s}_H]\|$, we can upper bound the compounding error: $\|\mathbb{E}[\hat{s}_H] - s_H\| \leq \|\mathbb{E}[\hat{s}_H]\| + \|s_H\|$.

4.2 Why and How Do we Perform Long-Horizon Planning?

Where to start planning? From the Bayesian objective (Eq. 4), it is natural to initiate planning rollouts by sampling initial states $s_0 \sim \rho_{\mathcal{D}}$, where $\rho_{\mathcal{D}}$ is the empirical initial-state distribution. But this underrepresents states appearing *later* in real trajectories, as compounding errors make them harder to reach (Lambert et al., 2022; Lin et al., 2025). While layer normalization substantially mitigates this issue, the errors at later time steps can still grow exponentially (e.g., Fig. 5). To maintain stability, we sample starting states $s_t \sim \mathcal{D}$ from any timestep t, following MBPO-style branched rollouts (Janner et al., 2019). Since the environment appears as an epistemic POMDP to the *agent*, we also provide the corresponding history $h_t = (s_{0:t}, a_{0:t-1}, r_{1:t}) \in \mathcal{D}$, which we refer to as the *initial history*.

Why do we need long-horizon planning? Ideally, under a correct posterior, the Bayesian principle calls for *full-horizon* planning. In practice, however, compounding errors make full-horizon rollouts unreliable. Even so, *long-horizon* rollouts remain valuable when they are informative for augmenting training data (Young et al., 2023). Beyond augmentation, we identify a new role for Bayesian offline RL: long-horizon rollouts also help mitigate **value overestimation**.

To illustrate, starting from a real history $h_t \in \mathcal{D}$, we sample $m_\theta \sim \mathbf{m}_{\boldsymbol{\theta}}$ and generate a trajectory of length H where $\hat{a}_{t+j} = \pi(\hat{h}_{t+j})$ comes from a deterministic policy, $(\hat{r}_{t+j+1}, \hat{s}_{t+j+1}) \sim m_\theta(\hat{s}_{t+j}, \hat{a}_{t+j})$, and $\hat{h}_t = h_t$. Applying one-step Bellman backups on the Bayesian value function *along this rollout*, i.e., $Q^{\text{Bayes}}(\hat{h}_{t+j}, \hat{a}_{t+j}) \leftarrow \hat{r}_{t+j+1} + \gamma Q^{\text{Bayes}}(\hat{h}_{t+j+1}, \pi(\hat{h}_{t+j+1}))$, $0 \leq j < H$, we obtain an approximate H-step backup on the real history:

$$Q^{\text{Bayes}}(h_t, \hat{a}_t) \leftarrow \sum_{j=0}^{H-1} \gamma^j \underbrace{\hat{r}_{t+j+1}}_{\text{lower bias}} + \gamma^H \underbrace{Q^{\text{Bayes}}(\hat{h}_{t+H}, \pi(\hat{h}_{t+H}))}_{\text{higher bias}}, \tag{6}$$

which highlights the bias trade-off: imagined rewards can be low-bias if the model generalizes, while the bootstrapped term – more susceptible to overestimation (Kumar et al., 2019; Sims et al., 2024) – is exponentially discounted with H. Prior works mitigate overestimation through uncertainty penalties on imagined rewards, enabling short-horizon rollouts. However, such short horizons suffer from severe overestimation with true models, known as the *edge-of-reach* problem (Sims et al., 2024). In contrast, NEUBAY can, by design, absorb the overestimation risk across long horizons without requiring hand-crafted penalties.

How should we truncate rollouts? Given that Bayesian RL favors long-horizon planning, a key question is not simply *when* to truncate, but *where*, since model errors depend on specific (s,a) pairs. A natural criterion is the model's uncertainty estimate $U_{\boldsymbol{\theta}}(s,a)$, which correlates with prediction

error. Prior work has pursued this *conservatively*, combining uncertainty threshold with short horizon caps (Frauenknecht et al., 2024; Pan et al., 2020; Zhang et al., 2023). In contrast, we remove restriction on the maximum horizon.

Epistemic uncertainty is highly *non-uniform* even within $\mathcal{D}$: frequently visited (s, a) pairs yield low uncertainty, while rarely seen ones result in high uncertainty. This reflects that epistemic uncertainty concerns the true model m^*, *not* the empirical model $m_{\mathcal{D}}$, which can be formalized via standard concentration bounds in the frequentist sense.[2] As shown in Fig. 4, most empirical CDFs are long-tailed, with dataset-dependent skewness reflecting the underlying visitation distribution. This motivates our use of data-dependent truncation thresholds, described next.

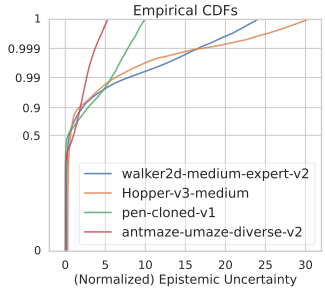

Figure 4: **Empirical CDFs** of epistemic uncertainty $U_{\boldsymbol{\theta}}$ over $(s, a) \in \text{supp}_{\mathcal{S}\times\mathcal{A}}(\mathcal{D})$, with logit-scaled y-axis. Uncertainties are normalized by the dataset mean, so 1 is the average value.

Design choice 3: uncertainty threshold $\mathcal{U}(\zeta)$ for rollout truncation. Let $\zeta \in [0, 1]$, we define

$$\mathcal{U}(\zeta) = \mathcal{U}_{\boldsymbol{\theta},\mathcal{D}}(\zeta) := F_Y^{-1}(\zeta), \quad Y := U_{\boldsymbol{\theta}}(s, a), \quad (s, a) \sim \mathcal{D},$$

where F_Y^{-1} is the quantile function of Y. Quantile-based thresholds adapt naturally to different datasets and uncertainty scales. When $\zeta = 1.0$, the threshold equals the maximum uncertainty in $\mathcal{D}$, encouraging the longest possible rollouts. We avoid thresholds beyond the dataset maximum, since this implies uncertainty beyond all real data, a proxy for out-of-distribution.

4.3 Stable Recurrent Off-Policy RL Training

The last core component of our algorithm is RL training. We follow the MBPO paradigm (Janner et al., 2019), using model-free RL on a mixture of real and model-generated data. Since the real data is off-policy and $\hat{\mathbb{P}}_{\mathcal{D}}$ induces a POMDP for the agent, we adopt recurrent off-policy RL (Ni et al., 2022) rather than on-policy RL (Fellows et al., 2025). We use a recurrent actor $\pi_\nu(a_t \mid h_t)$ and a recurrent critic $Q_\omega(h_t, a_t)$, each with a separate RNN encoder ($\nu_\phi(h_t)$ and $\omega_\phi(h_t)$, respectively) for stability (Ni et al., 2022). Long rollout horizons and the need for test-time generalization require memory spanning entire episodes (up to 1000 steps in our tasks), which exceeds the capacity of *vanilla* RNNs (Ni et al., 2023). To address this, we extend memoroid (Morad et al., 2024) to actor-critic architectures using linear recurrent units (LRUs) (Orvieto et al., 2023) as the RNN encoders, enabling both long-term memory and parallel optimization. While a memoroid-based architecture is powerful, it was originally designed for online POMDPs. Bridging the gap to epistemic POMDPs in offline RL introduces new challenges, motivating several design choices that we detail below.

Design choice 4: balancing real and imagined data with $\kappa \in (0, 1)$. Following MBPO, we introduce a mixing ratio κ to weight real versus imagined transitions in the off-policy RL loss (see Sec. D). Prior work shows this ratio is sensitive in both offline and online RL (Lai et al., 2021). Intuitively, when real data is of higher quality, a larger κ is preferable (Zhang et al., 2023).

Design choice 5: small context encoder learning rate η_ϕ for training stability. The learning rate is a critical hyperparameter in deep RL, and recurrent off-policy RL is especially sensitive to it. In online POMDPs, RESeL (Luo et al., 2024a) shows that recurrent encoders require much smaller (around $30\times$) learning rates than those of actor or critic heads, since history representations can diverge exponentially with history length even under tiny parameter changes.

4.4 Overall Algorithm

We now present the full method in Algorithm 1, integrating all design choices with the core rollout subroutine (Algorithm 2). NEUBAY remains conceptually simple, using single-step models and no uncertainty penalty, while carefully addressing the challenges of Bayesian offline RL. We highlight the lines where NEUBAY differs from prior work. For completeness, Sec. D provides details on RL loss and stopping criteria, although they are not needed to follow the experiments below.

[2] With probability $1 - \delta$, for $(s, a) \in \text{supp}_{\mathcal{S}\times\mathcal{A}}(\mathcal{D})$, $\text{TV}(m_{\mathcal{D}}(s, a), m^*(s, a)) \leq c_{m^*,\delta}/\sqrt{n_{\mathcal{D}}(s, a)}$, where $n_{\mathcal{D}}(s, a)$ is the visitation count in $\mathcal{D}$ and $c_{m^*,\delta}$ is a constant (Kumar et al., 2020, Section D.3).

Algorithm 1 NEUBAY: Full training loop

Require: Offline dataset $\mathcal{D}$, Online buffer $\mathcal{B} \leftarrow \emptyset$. World ensemble: $\mathbf{m_\theta}$, Recurrent actor: $\pi_\nu(a_t \mid h_t)$, Recurrent critic: $Q_\omega(h_t, a_t)$.

Pretrain $\mathbf{m_\theta}$ on $\mathcal{D}$ until convergence
while gradient steps $\leq$ max gradient steps **do**
  for $k = 1$ to K rollouts **do** ▷ Parallelized in practice
    Sample $m_\theta \sim \mathbf{m_\theta}$
    Sample initial history $h_t \sim \mathcal{D}$
    Append ROLLOUT(h_t, m_θ, π_ν) to $\mathcal{B}$
  end for
  for $g = 1$ to G gradient steps **do**
    Optimize recurrent off-policy RL loss $L(Q_\omega, \pi_\nu; \tau, \kappa)$ with $\tau \sim \mathcal{B}$
  end for
end while

Algorithm 2 NEUBAY: Rollout function

Require: Terminal function f_{term}, Max episode length T, Uncertainty threshold $\mathcal{U}(\zeta)$, ~~Rollout horizon H~~

function ROLLOUT(h_t, m_θ, π_ν)
  Set $\hat{h}_t = h_t$ and done to False
  while done is False **do**
    $\hat{a}_t \sim \pi_\nu(\hat{h}_t), (\hat{r}_{t+1}, \hat{s}_{t+1}) \sim m_\theta(\hat{s}_t, \hat{a}_t)$
    $\hat{d}_{t+1} = f_{\text{term}}(\hat{s}_t, \hat{a}_t, \hat{s}_{t+1})$
    $\text{trunc} = (U_\theta(\hat{s}_t, \hat{a}_t) > \mathcal{U}(\zeta))$
    $\text{done} = \hat{d}_{t+1} \vee \text{trunc} \vee (t+1 \geq T)$
    $\hat{h}_{t+1} = (\hat{h}_t, \hat{a}_t, \hat{r}_{t+1}, \hat{s}_{t+1}), t \leftarrow t+1$
  end while
  return Trajectory τ (initial history h_t plus imagined rollout)
end function

5 Experiments

Benchmarks. We evaluate on two widely used offline RL suites for continuous control: D4RL (Fu et al., 2020) and NeoRL (Qin et al., 2022). From **D4RL**, we use the **locomotion** benchmark (**12** datasets) spanning three MuJoCo environments: halfcheetah (hc), hopper (hp), and walker2d (wk). Each one provides four data regimes: random, medium-replay, medium, and medium-expert. From **NeoRL**, we use its **locomotion** benchmark (**9** datasets) built on the same environments, but with data collected from more deterministic policies, resulting in narrower coverage and different data regimes: Low, Medium, High. We also include the **Adroit** benchmark (**6** datasets) from D4RL, where a 28-DoF robotic arm manipulates objects to solve tasks (pen, door, hammer) with data collected from human demonstrations (human) and behavior cloning (cloned). The difficulty of Adroit lies in its high-dimensional control, small data size, and low-data quality. Lastly, we perform experiments on **AntMaze** (**6** datasets), where an 8-DoF ant robot must reach a goal position in a maze. Tasks span maze layouts of different sizes (umaze, medium, large) and start distributions (play, diverse), and are challenging for model-based RL due to navigation under sparse rewards. See more details in Sec. E.

Baselines. We evaluate against a broad set of offline RL baselines based on each benchmark. We group them into three categories, with a focus on model-based and Bayesian-inspired methods:

- **Conservative model-free RL:** behavior cloning (BC), CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2022), EDAC (An et al., 2021), ReBRAC (Tarasov et al., 2023a).
- **Conservative model-based RL:** MOPO (Yu et al., 2020), COMBO (Yu et al., 2021), RAMBO (Rigter et al., 2022), ARMOR (Bhardwaj et al., 2023), MOBILE (Sun et al., 2023), LEQ (Park and Lee, 2025), MoMo (Srinivasan and Knottenbelt, 2024), ADMPO (Lin et al., 2025), SUMO (Qiao et al., 2025), VIPO (Chen et al., 2025), ScorePen (Liu et al., 2025).
- **Bayesian-inspired methods:** APE-V (Ghosh et al., 2022), MAPLE (Chen et al., 2021c), CBOP (Jeong et al., 2023), MoDAP (Choi et al., 2024).

Algorithm setup. In our main experiments (Sec. 5.1), we adopt the following **default** design choices: ensemble size $N = 100$, layer normalization in the world models, uncertainty threshold $\mathcal{U}(\zeta)$ of $\zeta = 1.0$. The learning rates of the actor and critic MLP heads are fixed to 1×10^{-4}, while their RNN encoders share a tied learning rate η_ϕ swept over $[3 \times 10^{-7}, 1 \times 10^{-4}]$, the exact range being benchmark-dependent. Similarly, the real data ratio κ is swept within $[0.05, 0.95]$, also depending on the benchmark. We report the best results for each dataset. Implementation details are provided in Sec. F. In Sec. 5.2, we conduct sensitivity and ablation studies on design choices.

5.1 Benchmarking Results

We present results for D4RL locomotion in Tab. 1, NeoRL locomotion in Tab. 9, D4RL Adroit in Tab. 2, and D4RL AntMaze in Tab. 10. Overall, NEUBAY achieves competitive performance relative to prior conservative algorithms. The average normalized scores are 80.1 vs. 83.6 for the best baseline (VIPO) in D4RL locomotion, 64.7 vs. 73.3 for the best baseline (VIPO) in NeoRL locomotion, 21.1 vs. 28.1 for the best model-based baseline (MoMo) in D4RL Adroit, and 28.8 vs. 64.9 for the best model-based baseline (LEQ) in D4RL AntMaze. NEUBAY establishes **new state-of-the-art** (SOTA) mean performance on 7 of the 33 datasets, with 3 gains statistically significant. In terms of magnitude, it advances walker-random-v2 (27.9 $\rightarrow$ 34.1), walker-medium-v2 (95.5 $\rightarrow$ 106.4), pen-cloned-v1 (74.1 $\rightarrow$

Table 1: Comparison of offline RL methods on the **D4RL locomotion** benchmark. We report mean normalized scores for all baselines, with ±std for competitive baselines. The **best mean score** is bolded, and marked methods are statistically similar under a t-test. Our results use 6 seeds, each evaluated at the final step with 20 episodes.

	Model-free		Conservative model-based									Bayesian-inspired				Ours
Dataset	CQL	EDAC	COMBO	RAMBO	MOPO	LEQ	MOBILE	MoMo	ADMPO	SUMO	VIPO	APE-V	MAPLE	MoDAP	CBOP	NEUBAY
hc-random	31.3	28.4	38.8	40.0	38.5	30.8±3.3	39.3±3.0	39.6±3.7	**45.4**±2.8	34.9±2.1	42.5±0.2	29.9	41.5	36.5±1.8	32.8±0.4	37.0±3.3
hp-random	5.3	25.3	17.9	21.6	31.7	32.4±0.3	31.9±0.6	18.3±2.8	32.7±0.2	30.8±0.9	**33.4**±1.9	31.3	10.7	8.9±1.1	31.4±0.0	24.5±28.5
wk-random	5.4	16.6	7.0	11.5	7.4	21.5±0.1	17.9±6.6	26.8±3.3	22.2±0.2	27.9±2.0	20.0±0.1	15.5	22.1	23.1±1.6	17.8±0.4	**34.1**±6.8
hc-med-rep	45.3	61.3	55.1	**77.6**	72.1	65.5±1.1	71.7±1.2	72.9±1.8	67.6±3.4	76.2±1.3	77.2±0.4	64.6	69.5	67.3±3.4	66.4±0.3	72.1±2.4
hp-med-rep	86.3	101.0	89.5	92.8	103.5	103.9±1.3	103.9±1.0	104.0±1.8	104.4±0.4	109.9±1.4	109.6±0.9	98.5	85.0	94.2±4.8	104.3±0.4	**110.6**±0.7
wk-med-rep	76.8	87.1	56.0	86.9	85.6	98.7±6.0	89.9±1.5	90.4±7.7	95.6±2.1	78.2±1.5	98.4±0.3	82.9	75.4	88.4±4.2	92.7±0.9	**99.3**±19.3
hc-medium	46.9	65.9	54.2	68.9	73.0	71.7±4.4	74.6±1.2	77.1±0.9	72.2±0.6	**84.3**±2.4	80.0±0.4	69.1	48.5	77.3±1.1	74.3±0.2	78.6±1.6
hp-medium	61.9	101.6	97.2	96.6	62.8	103.4±0.3	106.6±0.6	**110.8**±2.3	107.4±0.6	104.8±2.1	107.7±1.0	—	44.1	106.6±1.9	102.6±0.1	54.2±7.2
wk-medium	79.5	92.5	81.9	85.0	84.1	74.9±26.9	87.7±1.1	95.0±1.4	93.2±1.1	94.1±2.5	93.1±1.8	90.3	81.3	81.1±6.5	95.5±0.4	**106.4**±23.0
hc-med-exp	95.0	106.3	90.0	93.7	90.8	102.8±0.4	108.2±2.5	107.9±1.9	103.7±0.2	106.6±2.4	**110.0**±0.4	101.4	55.4	103.4±4.3	105.4±1.6	109.5±8.7
hp-med-exp	96.9	110.7	111.1	83.3	81.6	109.4±1.8	112.6±0.2	109.1±0.4	112.7±0.3	107.8±0.7	113.2±0.1	105.7	95.3	94.5±7.8	111.6±0.2	**114.8**±0.5
wk-med-exp	109.1	114.7	103.3	68.3	112.9	108.2±1.3	115.2±0.7	118.4±0.9	114.9±0.3	**122.8**±0.4	117.7±1.0	110.0	107.0	112.2±2.8	117.2±0.5	120.7±1.3
AVG	61.6	76.0	66.8	68.9	70.3	76.9	80.0	80.9	81.0	81.5	**83.6**	—	61.3	74.5	79.3	80.1

Table 2: Comparison of offline RL methods on the **D4RL Adroit** benchmark. We report mean normalized scores for all baselines, with ±std for competitive baselines. The **best mean score** is bolded, and marked methods are statistically similar under a t-test. Our results use 6 seeds, each evaluated at the final step with 20 episodes. In addition, we include the average dataset performance, denoted as $\pi_{\mathcal{D}}$, to provide a reference for **data quality**.

		Model-free				Conservative model-based					Ours
Dataset	$\pi_{\mathcal{D}}$	BC	EDAC	IQL	ReBRAC	MOPO	MOBILE	ARMOR	MoMo	VIPO	NEUBAY
pen-human	88.7	71.0±6.3	52.1±8.6	78.5±8.2	**103.2**±8.5	10.7	30.1±14.6	72.8±13.9	74.9	52.6±7.7	20.8±13.2
pen-cloned	68.7	51.9±15.2	68.2±7.3	83.4±8.2	**102.8**±7.8	54.6	69.0±9.3	51.4±15.5	74.1	71.1±9.5	91.3±21.2
door-human	7.7	2.3±4.0	10.7±6.8	3.3±1.8	-0.1±0.0	-0.2	-0.2±0.1	6.3±6.0	**11.3**	2.0±0.3	0.0±0.0
door-cloned	3.2	-0.1±0.0	9.6±8.3	3.1±1.8	0.1±0.1	15.3	**24.0**±22.8	-0.1±0.0	5.8	—	0.0±0.0
hammer-human	1.5	**3.0**±3.4	0.8±0.4	1.8±0.8	0.2±0.2	0.3	0.4±0.2	1.9±1.6	1.7	1.1±0.9	0.0±0.0
hammer-cloned	0.7	0.6±0.2	0.3±0.0	1.5±0.7	5.0±3.8	0.5	1.5±0.4	0.7±0.6	0.7	2.1±0.2	**14.4**±9.7
AVG	N/A	21.5	23.6	28.6	**35.2**	13.5	20.8	22.2	28.1	—	21.1

91.3), and hammer-cloned-v1 (5.0 → 14.4). Furthermore, NEUBAY surpasses prior Bayesian-inspired methods on average performance and most per-task scores, despite their reliance on conservatism.[3]

When is NEUBAY better than conservative methods? We group the 33 datasets into 3 categories:

- **Low-quality datasets (11 tasks):** includes 3 D4RL random, 3 NeoRL Low, 2 Adroit door, 2 Adroit hammer, and AntMaze umaze-diverse. Consistent with the bandit insight (Sec. 3), NEUBAY ranks among the best methods in 5 of 11 tasks and achieves reasonable performance in 10 of 11.
- **Medium-quality, moderate coverage (7 tasks):** includes 3 D4RL medium-replay, 3 D4RL medium-expert[4], and Adroit pen-cloned. NEUBAY ranks among the best methods in 5 of 7 tasks, without any failures.
- **Medium-quality, narrow coverage (15 tasks):** includes 3 D4RL medium, 3 NeoRL Medium, 3 NeoRL High, Adroit pen-human, 5 AntMaze tasks. NEUBAY is weaker, ranking among the best methods in only 3 of 15 tasks, while falling short in 10. This gap from the bandit insight stems from the difficulty of posterior modeling on narrow datasets, where severe model error in high-dimensional spaces can mislead NEUBAY's RL agent. Future advances in world modeling may help close this gap.

5.2 What Matters in NEUBAY?

To understand which components are critical to the success of NEUBAY, we conduct sensitivity studies on the design choices introduced in Sec. 4 and ablation study on the uncertainty penalty, using the best hyperparameters identified in Sec. 5.1.

[3] An exception is MoDAP, which avoids conservatism by continuing model training during policy learning.

[4] D4RL expert and NeoRL High are treated as medium-quality, as their behavior policies are not fully optimal.

Table 3: **Sensitivity and ablation results averaged across benchmarks.** The highlighted setting (N=100, λ=0.0, ζ=1.0) corresponds to NEUBAY; ablations vary one hyperparameter at a time. Shading shows degradation level: light (3–10), medium (10–30), dark (>30). Full per-dataset results appear in Tab. 11.

	Ensemble size N			Unc. penalty coef. λ					Truncation threshold ζ			
Benchmark	100	20	5	0.0	0.04	0.2	1.0	5.0	1.0	0.999	0.99	0.9
D4RL Locomotion (12 tasks)	80.1	71.6	65.9	80.1	79.8	80.4	73.1	57.4	80.1	69.6	45.1	22.5
NeoRL Locomotion (9 tasks)	64.7	60.3	59.8	64.7	61.5	59.8	58.3	42.5	64.7	58.8	36.3	16.7
D4RL Adroit (3 non-zero tasks)	42.2	30.0	32.6	42.2	33.7	34.2	38.1	34.1	42.2	17.8	16.0	1.9
D4RL AntMaze (4 non-zero tasks)	43.2	33.2	28.8	43.2	5.0	3.2	4.2	12.0	43.2	35.8	38.4	32.7

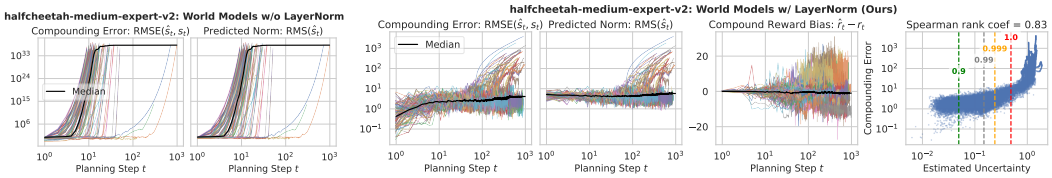

Figure 5: Effect of LayerNorm in world models on D4RL halfcheetah-medium-expert-v2. **Left: without LN. Right: with LN.** Rollouts are generated by a uniformly random policy, truncated by the episode limit ($T = 1000$) or `float32` overflow, but not uncertainty threshold. Vertical lines in the rightmost scatter plot mark uncertainty thresholds $\zeta \in \{0.9, 0.99, 0.999, 1.0\}$.

LayerNorm in the world model and uncertainty-based truncation jointly enables long-horizon rollouts. Fig. 5 compares pretrained world models on halfcheetah-medium-expert-v2 under the same random policy. Without LN (left), predicted state norms diverge quickly (around 10 steps here), driving exploding compounding errors. With LN (right), state norms remain bounded, which suppresses error growth and stabilizes reward predictions. This matches our intuition based on Eq. 5: by normalizing features at each step, LN constrains prediction magnitudes and thus compounding error. Moreover, the rightmost scatter plot shows what uncertainty-based truncation *would* accomplish: although we display full rollouts to reveal their divergence, applying thresholds would cut them off before entering high-error regions (here, around $\geq 10^1$). Thus, *LN prevents error explosion, while uncertainty cutoff provides a complementary safeguard.* Plotting setup and additional results on other datasets are shown in Sec. G.2.

Adaptive long-horizon planning is the *decisive* factor for success. We study the role of adaptive horizons by varying $\zeta \in \{0.9, 0.99, 0.999, 1.0\}$. Almost on all datasets, we find that the most aggressive choice, $\zeta = 1.0$, delivers the best performance. Fig. 1 shows a typical case, with full plots in Sec. G.3, a summary in Tab. 3, and complete results in Tab. 11. Using $\zeta = 1.0$ enables median horizons of **32–512 steps** and maximum horizons of **128–1000 steps** across tasks.

These horizons are surprisingly long, running counter to the conventional wisdom in model-based RL that long rollouts hurt performance (Janner et al., 2019). That belief is justified when horizons are *fixed*, worst-case compounding error grows exponentially and makes long rollouts impractical (e.g., Fig. 5). Our results show that with *adaptive* horizons, compounding errors are kept under control, allowing rollouts to extend far beyond what fixed horizons permit. In addition, the large ensemble of models together with the *neutral* Bayesian objective prevents the agent from overcommitting to any single erroneous prediction, yielding robust long-horizon planning.

On the other hand, performance often collapses to *near zero* (i.e., scores ≤ 5) with smaller thresholds such as 0.9 (16 datasets) or 0.99 (6 datasets). The consistent failure mode behind these cases is **severe value overestimation**, where estimated Q-values ($\mathbb{E}_{(h_t,a_t)\sim\mathcal{D}}[Q_\omega(h_t, a_t)]$) are much higher (2nd column in Fig. 1) compared to the actual performances. These results support our intuition in Eq. 6: without conservatism, Bayesian RL mitigates overestimation by relying more on imagined rewards (lower-biased as shown in Fig. 5) using long horizons than on bootstrapped values. Crucially, adaptive horizons allow the algorithm to "trust" the model only within its in-distribution confidence region. As indicated by Fig. 4, the region at $\zeta = 1.0$ is much larger than at $\zeta = 0.9$, yielding substantially longer horizons that provide useful data for both augmentation and mitigating overestimation.

Introducing conservatism to NEUBAY helps some tasks, but not on average. To study conservatism in NEUBAY, we penalize the rewards with $\lambda \frac{U_{\boldsymbol{\theta}}(\hat{s},\hat{a})}{\mathbb{E}_{(s,a)\sim\mathcal{D}}[U_{\boldsymbol{\theta}}(s,a)]}$, normalized by the dataset average so that λ is more comparable across datasets. As summarized in Tab. 3, a strong uncertainty penalty ($\lambda = 1.0$ or 5.0) generally hurts performance, while a small penalty ($\lambda = 0.04$) performs comparably to NEUBAY ($\lambda = 0$). The impact remains dataset-dependent, as detailed in Tab. 11.

Consistent with the bandit intuition, heavy penalties significantly worsen performance on 6 of 8 low-quality datasets, while leaving $*$-medium-expert datasets largely unaffected. In contrast, some tasks benefit substantially: hopper-random-v2 (24.5 $\rightarrow$ 48.2, a new SOTA), hopper-medium-v2 (54.2 $\rightarrow$ 105.8), and pen-human-v1 (20.8 $\rightarrow$ 35.9), but each requires a different λ, highlighting the need for tuning. However, penalties are not a universal remedy for narrow data: Walker2d-v3-Medium and halfcheetah-medium-v2 still fall short of the best baselines, even after tuning. Finally, in the AntMaze domain, penalties severely harm performance, reflecting the already poor quality of these datasets.

Context encoder learning rate and real data ratio have to be tuned per dataset. The best values of η_ϕ and κ are reported in Tab. 5-Tab. 7, with selective learning curves shown in Fig. 11- Fig. 12. We find the optimal encoder learning rates are generally smaller than those used in online POMDPs (Luo et al., 2024a). For example, 1×10^{-6} and 3×10^{-7} yield the best results on 7 datasets. Our intuition is that, in Bayesian offline RL, smaller learning rates help curb overestimation by slowing down learning. For the real data ratio, we find $\kappa = 0.05$, widely used in prior conservative algorithms (Yu et al., 2020), often yields poor performance. In Bayesian offline RL, large κ acts as a *softened* uncertainty penalty, limiting overtrust in the model while avoiding explicit penalties.

Larger ensemble size improves performance. As summarized in Tab. 3 and detailed in Tab. 11, reducing the ensemble size N to 20 or 5 clearly degrades performance, though not drastically on most tasks. This indicates that $N = 100$ is close to the practical limit of what ensembling can offer for these tasks, and future work may explore more scalable ways to approximate the posterior.

6 Conclusion and Future work

Practical recommendations for tuning NEUBAY. Drawing on our benchmarking and sensitivity studies in Sec. 5, we suggest the following guidelines when applying NEUBAY to new tasks or after modifying a key module (e.g., world model, uncertainty quantifier, or memory encoder):

- *Check dataset quality.* NEUBAY excels on low-quality or moderate-coverage datasets, but can underperform on narrow, medium-quality ones possibly due to current limits in posterior modeling.
- *Defaults first.* Start with large ensemble size, layer normalization in the world model, and uncertainty threshold $\zeta = 1.0$.
- *Context encoder rate.* Tune η_ϕ within a wide range, typically 3–300$\times$ smaller than the MLP head's learning rate.
- *Real data ratio.* Adjust κ with $[0.5, 0.8]$ as a robust starting range.
- *Overestimation control.* Monitor estimated values; if severe overestimation occurs, mitigate it by reducing the discount factor γ or lowering η_ϕ, as suggested by our analysis in Eq. 6.

Conclusion. In this paper, we revisit conservatism as the dominant principle in offline RL and show that it is not universally optimal. Instead, we advance a Bayesian perspective that models epistemic uncertainty and trains history-dependent agents to maximize expected rewards over a posterior of world models. Building on this principle, we propose NEUBAY, a practical algorithm that mitigates compounding error with layer normalization, reduces value overestimation with adaptive long-horizon planning, and stabilizes recurrent training. Experiments across diverse benchmarks show that NEUBAY generally matches or surpasses strong conservative baselines, excels particularly on low-quality and moderate-coverage datasets, and challenges the belief that short-horizon planning is necessary in model-based RL.

Future work. We see several promising directions for future work. Improving world models, through multi-step prediction or generative models, can help push the limits of Bayesian offline RL. Better uncertainty quantification remains key for planning, suggesting deeper connections to Bayesian inference are worth exploring. Finally, reducing sensitivity to hyperparameters and dataset characteristics, as well as advancing off-policy evaluation for tuning, would make NEUBAY more robust and practical.

References

Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pages 104–114. PMLR, 2020. 19

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021. 7, 18, 20, 22

Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. In *International Conference on Learning Representations*, 2020. 17

Kavosh Asadi, Dipendra Misra, and Michael Littman. Lipschitz continuity in model-based reinforcement learning. In *International conference on machine learning*, pages 264–273. PMLR, 2018. 19

Kavosh Asadi, Dipendra Misra, Seungchan Kim, and Michel L Littman. Combating the compounding-error problem with a multi-step model. *arXiv preprint arXiv:1905.13320*, 2019. 19

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 2, 5, 19

Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhi-Hong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. In *International Conference on Learning Representations*, 2022. 22

Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pages 1577–1594. PMLR, 2023. 2, 5

Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023. 20

Mohak Bhardwaj, Tengyang Xie, Byron Boots, Nan Jiang, and Ching-An Cheng. Adversarial model for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:1245–1269, 2023. 7, 18

Guy E Blelloch. Prefix sums and their applications. *School of Computer Science, Carnegie Mellon University Pittsburgh, PA, USA*, 1990. 20

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax. 25, 27

Anthony R Cassandra, Leslie Pack Kaelbling, and Michael L Littman. Acting optimally in partially observable stochastic domains. In *Aaai*, volume 94, pages 1023–1028, 1994. 23

Jiayu Chen, Wentse Chen, and Jeff Schneider. Bayes adaptive monte carlo tree search for offline model-based reinforcement learning. *arXiv preprint arXiv:2410.11234*, 2024. 18

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021a. 3

Xinyue Chen, Che Wang, Zijian Zhou, and Keith W Ross. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021b. 26

Xiong-Hui Chen, Yang Yu, Qingyang Li, Fan-Ming Luo, Zhiwei Qin, Wenjie Shang, and Jieping Ye. Offline model-based adaptable policy learning. *Advances in Neural Information Processing Systems*, 34:8432–8443, 2021c. 2, 7, 18, 29

Xuyang Chen, Guojian Wang, Keyu Yan, and Lin Zhao. Vipo: Value function inconsistency penalized offline reinforcement learning. *arXiv preprint arXiv:2504.11944*, 2025. 7, 18

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 20

Yunseon Choi, Li Zhao, Chuheng Zhang, Lei Song, Jiang Bian, and Kee-Eung Kim. Diversification of adaptive policy for effective offline reinforcement learning. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 3863–3871, 2024. 7, 18, 30

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018. 4

Richard Dearden, Nir Friedman, Stuart Russell, et al. Bayesian q-learning. *Aaai/iaai*, 1998:761–768, 1998. 20

Esther Derman, Daniel J Mankowitz, Timothy A Mann, and Shie Mannor. Soft-robust actor-critic policy-gradient. In *Uncertainty in Artificial Intelligence*, 2018. 3, 21

Esther Derman, Daniel Mankowitz, Timothy Mann, and Shie Mannor. A bayesian approach to robust reinforcement learning. In *Uncertainty in Artificial Intelligence*, pages 648–658. PMLR, 2020. 20

Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time, i. *Communications on pure and applied mathematics*, 28(1):1–47, 1975. 22

Ron Dorfman, Idan Shenfeld, and Aviv Tamar. Offline meta reinforcement learning–identifiability challenges and effective data collection strategies. *Advances in Neural Information Processing Systems*, 34:4607–4618, 2021. 20

Michael O'Gordon Duff. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. University of Massachusetts Amherst, 2002. 2, 3, 18, 20

Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021. 1

Daniel Ellsberg. Risk, ambiguity, and the savage axioms. *The quarterly journal of economics*, 75(4):643–669, 1961. 3, 20

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6, 2005. 19

Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I Jordan, Joseph E Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. In *International Conference on Machine Learning*, 2018. 18

Mattie Fellows, Clarisse Wibault, Uljad Berdica, Johannes Forkel, Jakob N Foerster, and Michael A Osborne. Sorel and torel: Two methods for fully offline reinforcement learning. *arXiv preprint arXiv:2505.22442*, 2025. 3, 6

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019. 4

Bernd Frauenknecht, Artur Eisele, Devdutt Subhasish, Friedrich Solowjow, and Sebastian Trimpe. Trust the model where it trusts itself-model-based actor-critic with uncertainty-aware rollout adaption. In *International Conference on Machine Learning*, pages 13973–14005. PMLR, 2024. 2, 6, 20

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020. 2, 3, 7, 24

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019. 2, 3, 19, 21, 22

Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016. 3

Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pages 3682–3691. PMLR, 2021. 22

Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, Aviv Tamar, et al. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6):359–483, 2015. 3, 20

Dibya Ghosh, Jad Rahme, Aviral Kumar, Amy Zhang, Ryan P Adams, and Sergey Levine. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability. *Advances in neural information processing systems*, 34:25502–25515, 2021. 20

Dibya Ghosh, Anurag Ajay, Pulkit Agrawal, and Sergey Levine. Offline rl policies should be trained to be adaptive. In *International Conference on Machine Learning*, pages 7513–7530. PMLR, 2022. 2, 3, 7, 18, 20, 23

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. 20

Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Thomas Paine, Sergio Gómez, Konrad Zolna, Rishabh Agarwal, Josh S Merel, Daniel J Mankowitz, Cosmin Paduraru, et al. Rl unplugged: A suite of benchmarks for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:7248–7259, 2020. 2, 19

Fredrik K Gustafsson, Martin Danelljan, and Thomas B Schon. Evaluating scalable bayesian deep learning methods for robust computer vision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 318–319, 2020. 4

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018a. 23, 26

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b. 26, 27

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023. 19

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024. 19

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 20

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, 32, 2019. 2, 5, 6, 9, 17, 19

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021. 19

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022. 19

Jihwan Jeong, Xiaoyu Wang, Michael Gimelfarb, Hyunwoo Kim, Baher Abdulhai, and Scott Sanner. Conservative bayesian model-based value expansion for offline policy optimization. In *International Conference on Learning Representations*, 2023. 2, 7, 18, 22

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pages 5084–5096. PMLR, 2021. 2

Steven Kapturowski, Georg Ostrovski, John Quan, Remi Munos, and Will Dabney. Recurrent experience replay in distributed reinforcement learning. In *International conference on learning representations*, 2018. 18

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33:21810–21823, 2020. 17, 22

Patrick Kidger and Cristian Garcia. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming workshop at Neural Information Processing Systems 2021*, 2021. 25

Byeongchan Kim and Min-hwan Oh. Model-based offline reinforcement learning with count-based conservatism. In *International Conference on Machine Learning*, pages 16728–16746. PMLR, 2023. 18

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. 7, 21, 22, 27

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019. 5, 21, 22

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020. 6, 7

Michail G Lagoudakis and Ronald Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003. 18

Hang Lai, Jian Shen, Weinan Zhang, Yimin Huang, Xing Zhang, Ruiming Tang, Yong Yu, and Zhenguo Li. On effective scheduling of model-based reinforcement learning. *Advances in Neural Information Processing Systems*, 34:3694–3705, 2021. 6

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017. 4, 5

Nathan Lambert, Kristofer Pister, and Roberto Calandra. Investigating compounding prediction errors in learned dynamics models. *arXiv preprint arXiv:2203.09637*, 2022. 2, 5, 19, 30

Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning: State-of-the-art*, pages 45–73. Springer, 2012. 2

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020. 2

Haoxin Lin, Yu-Yan Xu, Yihao Sun, Zhilong Zhang, Yi-Chen Li, Chengxing Jia, Junyin Ye, Jiaji Zhang, and Yang Yu. Any-step dynamics model improves future predictions for online and offline reinforcement learning. In *International Conference on Learning Representations*, 2025. 5, 7, 19, 25, 27, 30, 36

Zeyuan Liu, Zhirui Fang, Jiafei Lyu, and Xiu Li. Leveraging score-based models for generating penalization in model-based offline reinforcement learning. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, pages 1389–1398, 2025. 7

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 26

Chenhao Lu, Ruizhe Shi, Yuyao Liu, Kaizhe Hu, Simon Shaolei Du, and Huazhe Xu. Rethinking transformers in solving pomdps. In *Forty-first International Conference on Machine Learning*, 2024. 20

Chris Lu, Yannick Schroecker, Albert Gu, Emilio Parisotto, Jakob Foerster, Satinder Singh, and Feryal Behbahani. Structured state space models for in-context reinforcement learning. *Advances in Neural Information Processing Systems*, 36:47016–47031, 2023. 20, 27

Cong Lu, Philip Ball, Jack Parker-Holder, Michael Osborne, and Stephen J Roberts. Revisiting design choices in offline model based reinforcement learning. In *International Conference on Learning Representations*, 2021. 2, 19, 30

Carlos E Luis, Alessandro G Bottero, Julia Vinogradska, Felix Berkenkamp, and Jan Peters. Uncertainty representations in state-space layers for deep reinforcement learning under partial observability. *arXiv preprint arXiv:2409.16824*, 2024. 20

Fan-Ming Luo, Zuolin Tu, Zefang Huang, and Yang Yu. Efficient recurrent off-policy rl requires a context-encoder-specific learning rate. *Advances in Neural Information Processing Systems*, 37:48484–48518, 2024a. 2, 6, 10, 20, 26, 27

Fan-Ming Luo, Tian Xu, Xingchen Cao, and Yang Yu. Reward-consistent dynamics models are strongly generalizable for offline reinforcement learning. In *International Conference on Learning Representations*, 2024b. 18, 19

Michel Ma, Tianwei Ni, Clement Gehring, Pierluca D'Oro, and Pierre-Luc Bacon. Do transformer world models give better policy gradients? In *International Conference on Machine Learning*, pages 33855–33879. PMLR, 2024. 19

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 23

Steven Morad, Ryan Kortvelesy, Matteo Bettini, Stephan Liwicki, and Amanda Prorok. Popgym: Benchmarking partially observable reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. 20

Steven Morad, Chris Lu, Ryan Kortvelesy, Stephan Liwicki, Jakob Foerster, and Amanda Prorok. Recurrent reinforcement learning with memoroids. *Advances in Neural Information Processing Systems*, 37:14386–14416, 2024. 2, 6, 20, 26, 27

Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent model-free rl can be a strong baseline for many pomdps. In *International Conference on Machine Learning*, pages 16691–16723. PMLR, 2022. 6, 20, 26

Tianwei Ni, Michel Ma, Benjamin Eysenbach, and Pierre-Luc Bacon. When do transformers shine in rl? decoupling memory from credit assignment. In *Advances in Neural Information Processing Systems*, 2023. 2, 6, 20

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences. In *International Conference on Machine Learning*, pages 26670–26698. PMLR, 2023. 6, 20, 26, 27

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29, 2016. 20

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019. 4

Feiyang Pan, Jia He, Dandan Tu, and Qing He. Trust the model when it is confident: Masked model-based actor-critic. *Advances in neural information processing systems*, 33:10537–10546, 2020. 6, 19

Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. Stabilizing transformers for reinforcement learning. In *International conference on machine learning*, pages 7487–7498. PMLR, 2020. 20

Kwanyoung Park and Youngwoon Lee. Model-based offline reinforcement learning with lower expectile q-learning. In *International Conference on Learning Representations*, 2025. 7, 18, 25, 30, 36

Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8):10237–10257, 2023. 2

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014. 2

Zhongjian Qiao, Jiafei Lyu, Kechen Jiao, Qi Liu, and Xiu Li. Sumo: Search-based uncertainty estimation for model-based offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 20033–20041, 2025. 5, 7, 18

Rong-Jun Qin, Xingyuan Zhang, Songyi Gao, Xiong-Hui Chen, Zewen Li, Weinan Zhang, and Yang Yu. Neorl: A near real-world benchmark for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:24753–24765, 2022. 2, 7, 24, 30

Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. In *International Conference on Learning Representations*, 2017. 20

Martin Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*, pages 317–328. Springer, 2005. 19

Marc Rigter, Bruno Lacerda, and Nick Hawes. Risk-averse bayes-adaptive reinforcement learning. *Advances in Neural Information Processing Systems*, 34:1142–1154, 2021. 4

Marc Rigter, Bruno Lacerda, and Nick Hawes. Rambo-rl: Robust adversarial model-based offline reinforcement learning. *Advances in neural information processing systems*, 35:16082–16097, 2022. 3, 7, 18, 22

Marc Rigter, Bruno Lacerda, and Nick Hawes. One risk to rule them all: A risk-sensitive perspective on model-based offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2023. 18, 22

Julian Schrittwieser, Thomas Hubert, Amol Mandhane, Mohammadamin Barekatain, Ioannis Antonoglou, and David Silver. Online and offline reinforcement learning by planning with a learned model. *Advances in Neural Information Processing Systems*, 34:27580–27591, 2021. 19

Anya Sims, Cong Lu, Jakob N Foerster, and Yee W Teh. The edge-of-reach problem in offline model-based reinforcement learning. *Advances in Neural Information Processing Systems*, 37:63029–63056, 2024. 5, 20

Ivan Smirnov and Shangding Gu. Rlbenchnet: The right network for the right reinforcement learning task. *arXiv preprint arXiv:2505.15040*, 2025. 20

Padmanaba Srinivasan and William Knottenbelt. Offline model-based reinforcement learning with anti-exploration. *arXiv preprint arXiv:2408.10713*, 2024. 7, 18

Yihao Sun. Offlinerl-kit: An elegant pytorch offline reinforcement learning library. https://github.com/yihaosun1124/OfflineRL-Kit, 2023. 25

Yihao Sun, Jiaji Zhang, Chengxing Jia, Haoxin Lin, Junyin Ye, and Yang Yu. Model-bellman inconsistency for model-based offline reinforcement learning. In *International Conference on Machine Learning*, pages 33177–33194. PMLR, 2023. 7, 18, 22, 25, 27, 29, 30

Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pages 216–224. Elsevier, 1990. 17

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 1

Erik Talvitie. Model regularization for stable sample rollouts. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 780–789, 2014. 5, 19

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:11592–11620, 2023a. 7

Denis Tarasov, Alexander Nikulin, Dmitry Akimov, Vladislav Kurenkov, and Sergey Kolesnikov. Corl: Research-oriented deep offline reinforcement learning library. *Advances in Neural Information Processing Systems*, 36: 30997–31020, 2023b. 30

Masatoshi Uehara and Wen Sun. Pessimistic model-based offline reinforcement learning under partial coverage. In *International Conference on Learning Representations*, 2022. 2, 3, 19, 22

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 20

Nikos Vlassis, Mohammad Ghavamzadeh, Shie Mannor, and Pascal Poupart. Bayesian reinforcement learning. *Reinforcement Learning: State-of-the-Art*, pages 359–386, 2012. 20

Risto Vuorio, Mattie Fellows, Cong Lu, Clémence Grislain, and Shimon Whiteson. A bayesian solution to the imitation gap. *arXiv preprint arXiv:2407.00495*, 2024. 20

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003. PMLR, 2016. 26

Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. Robust markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013. 3, 21

Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020. 4

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline rl with no ood actions: In-sample learning via implicit value regularization. In *International Conference on Learning Representations*, 2023. 21

Shentao Yang, Shujian Zhang, Yihao Feng, and Mingyuan Zhou. A unified framework for alternating offline model training and policy learning. *Advances in Neural Information Processing Systems*, 35:17216–17232, 2022. 18

Zhihan Yang and Hai Nguyen. Recurrent off-policy baselines for memory-based continuous control. *arXiv preprint arXiv:2110.12628*, 2021. 20

Denis Yarats, David Brandfonbrener, Hao Liu, Michael Laskin, Pieter Abbeel, Alessandro Lazaric, and Lerrel Pinto. Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning. In *ICLR 2022 Workshop on Generalizable Policy Learning in Physical World*, 2022. 19

Kenny John Young, Aditya Ramesh, Louis Kirsch, and Jürgen Schmidhuber. The benefits of model-based generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 40254–40276. PMLR, 2023. 5

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020. 2, 3, 4, 7, 10, 17, 19, 22, 23, 25

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021. 7, 18

Yuanzhao Zhai, Yiying Li, Zijian Gao, Xudong Gong, Kele Xu, Dawei Feng, Ding Bo, and Huaimin Wang. Optimistic model rollouts for pessimistic offline policy optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16678–16686, 2024. 19

Jing Zhang, Linjiajie Fang, Kexin Shi, Wenjia Wang, and Bingyi Jing. Q-distribution guided q-learning for offline reinforcement learning: Uncertainty penalized q-value via consistency model. *Advances in Neural Information Processing Systems*, 37:54421–54462, 2024a. 25

Junjie Zhang, Jiafei Lyu, Xiaoteng Ma, Jiangpeng Yan, Jun Yang, Le Wan, and Xiu Li. Uncertainty-driven trajectory truncation for data augmentation in offline reinforcement learning. In *ECAI 2023*, pages 3018–3025. IOS Press, 2023. 6, 20

Runyu Zhang, Yang Hu, and Na Li. Soft robust mdps and risk-sensitive mdps: Equivalence, policy gradient, and sample complexity. In *The Twelfth International Conference on Learning Representations*, 2024b. 2, 3, 4, 21, 22

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. In *International Conference on Learning Representations*, 2020. 4, 20

Appendix

A Related Work

A.1 Offline RL

Conservative model-based algorithms. Offline model-based RL (MBRL) methods learn world models to generate synthetic rollouts and can be broadly grouped into two categories. The first is *Dyna-style* approaches (Sutton, 1990; Janner et al., 2019), which augment the offline dataset with synthetic rollouts for policy training. The second is *inference-time planning* (Argenson and Dulac-Arnold, 2020), which generates rollouts to select actions at deployment. Both typically incorporate *conservatism*. Our work falls within the Dyna-style family.

In Dyna-style offline MBRL, most methods enforce conservatism by constructing a *pessimistic MDP* that penalizes imagined state-action pairs, which we refer to as **uncertainty-penalized pessimism** in Sec. B. The earliest examples are MOReL (Kidambi et al., 2020) and MOPO (Yu et al., 2020), both approximating the uncertainty set with world ensembles. MOReL adopts strong pessimism by mapping any state-action pair with ensemble disagreement above a threshold to an absorbing state with a large penalty. This aggressive design disables value bootstrapping on uncertain regions and explains why MOReL supports long-horizon planning (up to 500 steps) even with severe compounding errors.

In contrast, MOPO applies a milder penalty to imagined rewards based on the aleatoric uncertainty, retaining bootstrapping and thus limiting rollouts to very short horizons (typically 1-5 steps).

Building on MOPO, several follow-up works redesign the uncertainty quantifier: MOBILE (Sun et al., 2023) uses inconsistencies in Bellman operators across ensemble members; Kim and Oh (2023) uses the inverse frequencies of state-action pairs; MoMo (Srinivasan and Knottenbelt, 2024) adopts energy-based models; SUMO (Qiao et al., 2025) employs k-nearest neighbors. Other works introduce alternative conservative mechanisms: COMBO (Yu et al., 2021) uses CQL regularizer; LEQ (Park and Lee, 2025) uses lower expectile regression. From the model side, several works improve the model learning or rollout sampling: Luo et al. (2024b) trains discriminators on (s, a, s') to resample model rollouts by fidelity; VIPO (Chen et al., 2025) augments the standard MLE loss with a value-consistency objective that aligns behavior-policy values under model and true dynamics.

A different line of work jointly trains adversarial world models and policies, rather than freezing the models after pretraining, which we refer to as **adversarial model-based pessimism** in Sec. B. Several adversarial objectives have been explored: RAMBO (Rigter et al., 2022) minimizes value estimates; Yang et al. (2022) minimizes the divergence between real and imagined state-action distributions; ARMOR (Bhardwaj et al., 2023) minimizes value differences between the current and a reference policy; Rigter et al. (2023) extends the uncertainty set to incorporate aleatoric uncertainty.

Our work differs from these prior lines of research in principle, core components, and algorithmic design. We replace conservatism with Bayesianism as the guiding principle, substituting the standard Markovian actor-critic with a history-dependent one required by Bayesian principle, and derive the NEUBAY algorithm with adaptive long-horizon planning, fundamentally distinct from prior methods. Moreover, prior works typically rely on tuning two critical hyperparameters: the *conservatism coefficient* λ and *rollout horizon* H, to achieve strong performance. In contrast, NEUBAY eliminates the need for λ by design and replaces H with an uncertainty quantile ζ, which remains fixed at 1.0 in main experiments. This makes NEUBAY easy to apply in practice.

Bayesian-inspired algorithms. Several offline RL works draw inspiration from Bayesian ideas, such as using model posteriors and connections to Bayes-adaptive MDPs (Duff, 2002). However, their formulations or algorithms typically differ from optimizing the epistemic POMDP in Eq. 4, which is the focus of our work. For example, although Ghosh et al. (2022) introduces the Bayesian model-based formulation in Eq. 4, their proposed APE-V algorithm adopts a *conservative model-free* approximation. Rather than maintaining a posterior over dynamics models, APE-V uses an ensemble of belief-state Q-functions as a surrogate posterior over values, trained purely via TD errors. Each Q-function is trained using SAC-N (An et al., 2021), which enforces conservatism by minimizing over Q-ensemble predictions.

Similar to our algorithm, **MAPLE** (Chen et al., 2021c) and **MoDAP** (Choi et al., 2024) learn an ensemble of models for recurrent policy training. Unlike our approach, however, they store the hidden states of recurrent policies in the replay buffer and reuse them to initialize the policy's context for rollouts and updates. This design is known to induce context staleness in recurrent RL (Kapturowski et al., 2018), whereas our algorithm avoids this issue by storing and sampling full histories directly. MAPLE also reintroduces an uncertainty penalty and terminates rollouts based on a predefined state boundary, making it a conservative method. MoDAP remains penalty-free but fine-tunes the world models during policy learning to maintain ensemble diversity; therefore, its objective departs from the epistemic POMDP, which requires freezing the posterior during policy optimization.

Other works are Bayesian-inspired in different ways. CBOP (Jeong et al., 2023) uses a model posterior to weight multi-step TD targets in an MVE-style update (Feinberg et al., 2018) and applies lower-confidence penalties. Chen et al. (2024) proposes a Bayesian Monte Carlo planning method with an uncertainty penalty, used as a policy-improvement operator.

In contrast, our work aims to stay as close as possible to the epistemic POMDP: we avoid conservatism and do not fine-tune models during policy learning. The approximation in the algorithm level arises in the rollout distribution: we begin rollouts from intermediate histories and truncate them using an uncertainty threshold to mitigate compounding errors. As discussed in Sec. 4.2, using long rollouts keeps this deviation small in practice.

Offline RL algorithms without conservatism. Although conservatism dominates modern offline RL, classic offline (batch) RL was originally developed without any conservatism (Lagoudakis and

Parr, 2003; Ernst et al., 2005; Riedmiller, 2005). These methods are based on *fitted Q-iteration*, which directly applies Markovian Q-learning to offline data and enjoys convergence guarantees when the dataset has full coverage. While effective on small-scale problems with data collected by a random policy (Riedmiller, 2005), such algorithms are known to fail in high-dimensional settings (Fujimoto et al., 2019). More recently, Agarwal et al. (2020) provides an *optimistic* perspective, showing that standard off-policy RL trained on the 50M transitions from DQN's replay buffer can outperform the behavior policy. Likewise, Yarats et al. (2022) demonstrates that off-policy RL can surpass conservative methods on diverse-coverage datasets collected by unsupervised agents.

In the model-based setting, MBPO (Janner et al., 2019), using short-horizon rollouts without conservatism, is known to underperform conservative counterparts such as MOPO in offline RL benchmarks (Yu et al., 2020). However, MuZero Unplugged (Schrittwieser et al., 2021) shows that MuZero, without conservatism, can achieve strong performance on the RL Unplugged suite (Gulcehre et al., 2020), which contains 200M Atari transitions and DMC control datasets using replay buffer from near-optimal RL agents. Zhai et al. (2024) observes that making MOPO optimistic instead of pessimistic can achieve strong performance on halfcheetah-random-v2, but worse on other D4RL tasks.

In summary, prior work on offline RL without conservatism largely relies on standard off-policy RL, which treats offline RL as optimizing a single MDP and thus follows an *optimistic* principle (Agarwal et al., 2020). Such optimism can work well when the dataset has broad state-action coverage but typically fails under limited coverage. Our approach takes a different non-conservative path: instead of being optimistic, it follows a neutral Bayesian principle that lies **between optimism and pessimism**. This allows NEUBAY to avoid the failure modes of optimistic methods and, for the first time to our knowledge, extends the effectiveness of non-conservative offline RL to *low-quality* and *moderate-coverage* datasets, while still benefiting from diverse coverage when available.

A.2 Model-Generated Rollouts in RL

Reducing compounding error and the scale of rollout horizon. Model-generated rollouts suffer from compounding prediction errors (Talvitie, 2014; Lambert et al., 2022), which can grow quickly with the rollout horizon. These errors cause the performance gap between the policy evaluated under the learned model and under the true MDP, as formalized by theory such as the simulation lemma (see Uehara and Sun (2022, Lemma 9)). To prevent large compounding errors from harming policy learning, most online and offline MBRL methods (Janner et al., 2019; Yu et al., 2020; Lu et al., 2021; Hafner et al., 2023; Hansen et al., 2024) restrict rollouts to very short horizons (typically **1-20 steps**). These approaches often share a minimalist setup: standard MLP world models without layer normalization, pure MLE training, and rollout procedures without uncertainty awareness.

Reducing compounding errors is key to enabling longer rollouts and closing the performance gap between model-based and true-environment evaluation. Because compounding error reflects a model's prediction accuracy on unseen states visited by the current policy, the underlying goal is to improve generalization (Asadi et al., 2019). We categorize existing techniques into three dimensions: *model architecture*, *training objective*, and *inference strategy*. On the architectural side, Ma et al. (2024) proposes a Transformer world model conditioned only on the initial state and future actions, achieving rollout horizons up to 500 steps in online RL. AMDPO (Lin et al., 2025) adopts a similar idea with RNNs and reaches 50-step horizons in offline RL. For inference strategies, Luo et al. (2024b) resamples model rollouts using a fidelity estimator and scales horizons to 100 steps. Some works improve generalization across multiple dimensions: Trajectory Transformer (Janner et al., 2021) treats offline RL as planning under a trajectory-level world model and uses beam search with a 15-step horizon, while Diffuser (Janner et al., 2022) maximizes an MLE lower bound with diffusion models and performs planning via the diffusion process, reaching horizons up to 384 steps.

NEUBAY keeps the training objective fixed to standard MLE, while contributing simple and effective components along the other two dimensions. Architecturally, we apply layer normalization (Ba et al., 2016) to stabilize prediction magnitudes (Sec. 4.1), increasing model smoothness (Asadi et al., 2018) and thus generalization; at inference time, we truncate rollouts using an uncertainty threshold as a proxy for compounding error (Sec. 4.2). Together, these components enable NEUBAY to successfully use horizons of **100-1000** steps.

Mechanism of adaptive horizon. As discussed in Sec. 4.2, several prior works have explored adaptive rollout horizons using uncertainty thresholds. In online MBRL, Pan et al. (2020) truncates

rollouts for states whose uncertainty ranks in the top 25% of the current-step batch and caps the horizon at 10. Similarly, Frauenknecht et al. (2024) computes a 95% uncertainty quantile from the batch of first-step predictions in current rollouts, again with a maximum horizon of 10. In offline MBRL, Zhang et al. (2023) uses truncation thresholds proportional to the maximum uncertainty in the offline dataset, with a coefficient of 0.5 and a maximum horizon of 5. Overall, prior work employs adaptive horizons in a *conservative* manner, combining small thresholds with strict horizon caps. In contrast, our approach removes any maximum-horizon constraint and uses a large quantile threshold.

Effect of rollout length on value estimation. Sims et al. (2024) identifies the *edge-of-reach* problem in offline conservative MBRL: when short-horizon rollouts are used, bootstrapping occurs on states whose Q-values are never directly trained, leading to substantial overestimation once uncertainty penalties vanish (e.g., under true dynamics). Closely related to our work, they emphasize that such overestimation induced by short rollouts constitutes a distinct failure mode in offline MBRL, separate from classic compounding errors. Their error-propagation analysis (Sims et al., 2024, Proposition 1) shows that, when training on rollouts starting from a real state s_t, the TD error on s_t decomposes into approximation errors along intermediate states and an *extrapolation error* at the truncated state $\hat{s}_{t+H}$, discounted by γ^H, mirroring the insight in Eq. 6. To mitigate this, they replace dynamics-based uncertainty penalties with value-based ones via pessimistic Q-ensembles (An et al., 2021), while still relying on short-horizon rollouts.

Our work extends this line of reasoning in two ways. (1) We show that the same overestimation mechanism arises under Bayesian Bellman backups. (2) Instead of conservative short-horizon rollouts, we use adaptive long-horizon rollouts that exploit the vanishing factor γ^H to naturally reduce extrapolation error, removing the need for designing conservative regularization.

A.3 Bayesian and Partially Observable RL

Bayesian RL. Bayesian RL (Vlassis et al., 2012; Ghavamzadeh et al., 2015) models epistemic uncertainty (also known as ambiguity (Ellsberg, 1961)) for purposes such as exploration (Osband et al., 2016), robustness (Rajeswaran et al., 2017; Derman et al., 2020), and generalization (Ghosh et al., 2021). Uncertainty can be incorporated in model-free methods (e.g., Bayesian Q-learning (Dearden et al., 1998)) or in model-based methods (e.g., BAMDPs (Duff, 2002)). Depending on the agent's attitude, Bayesian RL can be ambiguity-seeking, ambiguity-neutral, or ambiguity-averse. Our work is grounded in the epistemic POMDP formulation (Ghosh et al., 2021, 2022), an ambiguity-neutral, model-based framework for generalization. This formulation stems from BAMDPs, which optimally balance the exploration-exploitation tradeoff but incur test-time exploratory costs, as illustrated in Sec. 3 and known as the *cost of exploration* (Vuorio et al., 2024).

Epistemic POMDPs are also related to *meta-RL* (Beck et al., 2023), which likewise seeks to maximize expected performance over a distribution of MDPs. Each MDP is called a task in meta-RL. Bayesian meta-RL (Zintgraf et al., 2020; Dorfman et al., 2021) explicitly models this MDP posterior. The key difference is that in meta-RL the true environment is itself a distribution over MDPs (a particular POMDP), so task uncertainty is aleatoric. In contrast, the epistemic POMDP assumes a single underlying MDP, and its task uncertainty is purely epistemic. As a result, meta-RL methods cannot be directly applied to solve the epistemic POMDP.

Recurrent model-free RL for online POMDPs. Model-free RL offers a simple and effective way to tackle online POMDPs without explicitly learning belief-state representations (Ni et al., 2022; Yang and Nguyen, 2021; Smirnov and Gu, 2025). Memory is typically implemented with recurrent neural networks (RNNs) such as LSTMs (Hochreiter and Schmidhuber, 1997) or GRUs (Cho et al., 2014), but recent work shows that these architectures struggle with long-term memory (Parisotto et al., 2020; Ni et al., 2023). State-space models (SSMs) with linear recurrence (Orvieto et al., 2023; Gu and Dao, 2023) have emerged as a compute-efficient alternative to Transformers (Vaswani et al., 2017), balancing long-term memory with parallel optimization (Blelloch, 1990). Recent applications of SSMs to recurrent RL (Lu et al., 2023, 2024; Morad et al., 2024; Luo et al., 2024a; Luis et al., 2024) demonstrate strong performance on online POMDP benchmarks (Morad et al., 2023; Zintgraf et al., 2020; Ni et al., 2022).

B Formal Connections between Conservatism and Robustness

In this section, we place prior conservative algorithms in the (soft) robust MDP framework. For classic model-free and adversarial model-based pessimism, we make explicit how their updates correspond to particular uncertainty sets in *robust* MDPs (Wiesemann et al., 2013). For uncertainty-penalized pessimism, we connect it to *soft robust* MDP framework (Zhang et al., 2024b).[5] We follow the notation introduced in Sec. 2.

Classic model-free pessimism. Many model-free methods enforce an in-support (Fujimoto et al., 2019; Kumar et al., 2019) or in-sample (Kostrikov et al., 2022; Xu et al., 2023) constraint on the Bellman backup. This amounts to updating a pessimistic value function Q^{MF} such that, for a transition tuple $(s, a, r, d, s') \in \mathcal{D}$,

$$Q^{\mathrm{MF}}(s,a) \leftarrow r + \gamma(1-d) \max_{a' \in \mathcal{A},\ \text{s.t. } (s',a') \in \mathcal{D}} Q^{\mathrm{MF}}(s',a'). \tag{7}$$

This update is equivalent to assigning all out-of-dataset state–action pairs the minimal reward $-r_{\max}$, thereby enforcing a worst-case behavior. The corresponding uncertainty set $\mathfrak{M}_{\mathcal{D}}$ in the robust MDP framework (Eq. 3) can be written explicitly:

Proposition 1. *If the pessimistic update in Eq. 7 converges to a fixed point, then the induced uncertainty set for the corresponding robust MDP is*

$$\mathfrak{M}_{\mathcal{D}}^{\mathrm{MF}} = \left\{ m \;\middle|\; \begin{array}{ll} m(r, s' \mid s, a) = m_{\mathcal{D}}(r, s' \mid s, a), & \forall (s,a) \in \mathcal{D} \\ m(r, s' \mid s, a) = p(r)\, \mathbb{1}(s' = s_{\textit{absorb}}), \ \ \forall p \in \Delta([-r_{\max}, r_{\max}]), & \forall (s,a) \notin \mathcal{D} \end{array} \right\}, \tag{8}$$

where $m_{\mathcal{D}}$ is the empirical model (Eq. 2) and $s_{\textit{absorb}} \notin \mathcal{D}$ is an artificial absorbing state.

Proof. First, let $m \in \mathfrak{M}_{\mathcal{D}}^{\mathrm{MF}}$ and decompose $m(r, s' \mid s, a) = R(r \mid s, a) P(s' \mid s, a)$, and denote the empirical model as $m_{\mathcal{D}}(r, s' \mid s, a) = R_{\mathcal{D}}(r \mid s, a) P_{\mathcal{D}}(s' \mid s, a)$. By construction, $\mathfrak{M}_{\mathcal{D}}^{\mathrm{MF}}$ places no uncertainty on transitions: for $(s, a) \in \mathcal{D}$, $P(s, a)$ equals the empirical transition $P_{\mathcal{D}}(s, a)$, while for $(s, a) \notin \mathcal{D}$, $P(s, a)$ deterministically transitions to s_{absorb}. Thus, all epistemic uncertainty is in the reward function R.

Applying the robust MDP framework (Wiesemann et al., 2013), the optimal value function Q^* is Markovian and satisfies the robust Bellman optimality equation:

$$Q^*(s,a) = \min_{R(s,a) \in \mathfrak{M}_{\mathcal{D}}^{\mathrm{MF}}(s,a)} \mathbb{E}_{r \sim R(s,a)}[r] + \gamma \mathbb{E}_{s' \sim P(s,a)} \left[\max_{a' \in \mathcal{A}} Q^*(s', a') \right], \forall (s,a) \in \mathcal{S} \times \mathcal{A}, \tag{9}$$

$$Q^*(s,a) - \gamma \mathbb{E}_{s' \sim P(s,a)} \left[\max_{a' \in \mathcal{A}} Q^*(s', a') \right] = \min_{R(s,a) \in \mathfrak{M}_{\mathcal{D}}^{\mathrm{MF}}} \mathbb{E}_{r \sim R(s,a)}[r] = \begin{cases} \mathbb{E}_{r \sim R_{\mathcal{D}}(s,a)}[r] & (s,a) \in \mathcal{D}, \\ -r_{\max} & (s,a) \notin \mathcal{D}, \end{cases} \tag{10}$$

where the last line uses the fact that $R(\cdot \mid s, a)$ may be any distribution on $[-r_{\max}, r_{\max}]$, so the worst case is attained by a Dirac mass at $-r_{\max}$.

Therefore, we can simplify Eq. 10 by cases. (1) Absorbing state:

$$\forall a \in \mathcal{A}, \quad Q^*(s_{\text{absorb}}, a) - \gamma \max_{a' \in \mathcal{A}} Q^*(s_{\text{absorb}}, a') = -r_{\max}. \tag{11}$$

Since $Q^*(s_{\text{absorb}}, a)$ is a constant w.r.t. a, it follows that $Q^*(s_{\text{absorb}}, a) = -\frac{r_{\max}}{1-\gamma}, \forall a$, reaches the minimal return. (2) Unseen state-action pairs,

$$\forall (s,a) \notin \mathcal{D}, \quad Q^*(s,a) - \gamma \max_{a' \in \mathcal{A}} Q^*(s_{\text{absorb}}, a') = -r_{\max}. \tag{12}$$

This implies $Q^*(s,a) = -\frac{r_{\max}}{1-\gamma}, \forall (s,a) \notin \mathcal{D}$. (2) Seen state-action pairs,

$$\forall (s,a) \in \mathcal{D}, \quad Q^*(s,a) = \mathbb{E}_{r \sim R_{\mathcal{D}}(s,a)}[r] + \gamma \mathbb{E}_{s' \sim P_{\mathcal{D}}(s,a)} \left[\max_{a' \in \mathcal{A}} Q^*(s', a') \right] \tag{13}$$

[5] The term "soft robustness" is used differently in prior work: Derman et al. (2018) use it for a Bayesian formalism, while Zhang et al. (2024b) define it as a risk-sensitive MDP relaxing strict worst-case robustness. In this paper, we adopt both usages but cite them accordingly.

$$= \mathbb{E}_{r\sim R_{\mathcal{D}}(s,a)}[r] + \gamma \mathbb{E}_{s'\sim P_{\mathcal{D}}(s,a)}\left[\max_{a'\in\mathcal{A},\text{s.t. }(s',a')\in\mathcal{D}} Q^*(s',a')\right] \tag{14}$$

The last line follows that $Q^*(s, a_{\text{out}}) = -\frac{r_{\max}}{1-\gamma} \leq Q^*(s, a_{\text{in}}), \forall (s, a_{\text{in}}) \in \mathcal{D}, \forall (s, a_{\text{out}}) \notin \mathcal{D}$. Finally, Eq. 14 recovers the pessimism principle underlying Eq. 7. This includes many model-free offline RL algorithms, such as BCQ (Fujimoto et al., 2019, Equation 10), BEAR (Kumar et al., 2019, Definition 4.1), EMaQ (Ghasemipour et al., 2021, Theorem 3.3), IQL (Kostrikov et al., 2022, Corollary 2.1). □

Adversarial model-based pessimism. One class of model-based methods constructs an explicit uncertainty set around the empirical model (Uehara and Sun, 2022; Rigter et al., 2022):

$$\mathfrak{M}_{\mathcal{D}}^{\text{MB}} = \{m \mid \mathbb{E}_{(s,a)\sim\mathcal{D}}[\text{div}(m(s,a), m_{\mathcal{D}}(s,a))] \leq \epsilon\}, \tag{15}$$

where popular choices of the divergence $\text{div}(\cdot,\cdot)$ include total variation (TV) distance and KL divergence.

Uncertainty-penalized pessimism: soft robust MDP. Another line of model-based methods incorporates explicit uncertainty penalties into value updates (Yu et al., 2020; Kidambi et al., 2020; Jeong et al., 2023; Sun et al., 2023). For imagined transitions $(\hat{s}, \hat{a}, \hat{r}, \hat{d}, \hat{s}')$, the pessimistic update takes the form:

$$Q^{\text{MB}}(\hat{s}, \hat{a}) \leftarrow \hat{r} - \lambda U(\hat{s}, \hat{a}) + \gamma(1-\hat{d}) \max_{\hat{a}'\in\mathcal{A}} Q^{\text{MB}}(\hat{s}', \hat{a}'), \tag{16}$$

where $U : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^+$ is an uncertainty measure based on dataset $\mathcal{D}$ and learned world models, and $\lambda > 0$ controls the degree of pessimism. Similar uncertainty penalties have also been incorporated into model-free value functions (Bai et al., 2022; An et al., 2021).

We now provide a connection between Eq. 16 and the *soft robust MDP* framework of (Zhang et al., 2024b, Section 5). While our derivation follows a similar line to theirs, we include it here to be self-contained. Consider a robust MDP with a *policy-dependent* uncertainty set:

$$\max_{\pi}\min_{m} J(\pi, m) \quad \text{s.t.} \quad \mathbb{E}_{(s,a)\sim(m,\pi)}[\text{div}(m(s,a), m^*(s,a))] \leq \epsilon, \tag{17}$$

where the divergence constraint describes the uncertainty set, taken in expectation under occupancy measure induced by m and π. We use Lagrangian relaxation with a coefficient $\alpha \geq 0$ to transform the problem into a soft robust MDP:

$$\max_{\pi}\min_{m} J(\pi, m) + \alpha \mathbb{E}_{(s,a)\sim(m,\pi)}[\text{div}(m(s,a), m^*(s,a))]. \tag{18}$$

In dynamic programming form, for a given (s, a) pair, the inner optimization becomes

$$\min_{m(\cdot|s,a)} \mathbb{E}_{s'\sim m(\cdot|s,a)}\left[V^{\text{MB}}(s')\right] + \alpha\, \text{div}(m(\cdot \mid s,a), m^*(\cdot \mid s,a)), \tag{19}$$

where V^{MB} is the policy's state-value function. This inner problem can be transformed by duality, depending on the choice of divergence. For the KL divergence, one can apply Donsker and Varadhan's formula (Donsker and Varadhan, 1975)[6] to state that Eq. 19 is equivalent to

$$-\alpha \log\left(\mathbb{E}_{s'\sim m^*(\cdot|s,a)}\left[\exp\left(-\frac{V^{\text{MB}}(s')}{\alpha}\right)\right]\right) = \mathbb{E}_{s'\sim m^*}\left[V^{\text{MB}}(s')\right] - \frac{1}{2\alpha}\text{Var}_{m^*}[V^{\text{MB}}(s')] + O\left(\frac{1}{\alpha^3}\right), \tag{20}$$

where we use cumulant expansion.[7]

Therefore, the corresponding soft-robust Bellman optimality equation (Zhang et al., 2024b, Equation 15), ignoring higher-order terms, becomes

$$Q^{\text{MB}}(s,a) = R^*(s,a) - \frac{\gamma}{2\alpha}\text{Var}_{s'\sim P^*}[\max_{a'} Q^{\text{MB}}(s',a')] + \gamma \mathbb{E}_{s'\sim P^*}\left[\max_{a'} Q^{\text{MB}}(s',a')\right] \tag{21}$$

In practice, model-based RL methods approximate $m^* = (R^*, P^*)$ with an ensemble of learned models trained on $\mathcal{D}$. While the variance term in Eq. 21 reflects the aleatoric uncertainty of the true dynamics, ensemble-based variance also incorporates epistemic uncertainty by the law of total variance, thus blending both. Prior work has interpreted the penalty as aleatoric (Yu et al., 2020), epistemic (Sun et al., 2023), or both (Rigter et al., 2023); here we focus on its epistemic interpretation. Accordingly, the ensemble variance provides a practical surrogate for the penalty in Eq. 16.

[6] For any probability distributions $x, x^* \in \Delta^k$ (the k-dimensional simplex), any vector $y \in \mathbb{R}^k$, and $\alpha > 0$, the duality formula is $-\alpha \log(\langle x^*, \exp(-y/\alpha)\rangle) = \min_x \langle x, y\rangle + \alpha\, \text{KL}(x \,||\, x^*)$.

[7] For a random variable X, $\log(\mathbb{E}[\exp(tX)]) = t\mathbb{E}[X] + \frac{t^2}{2}\text{Var}[X] + O(t^3)$. We substitute $t = -1/\alpha$.

C Connection between Bayesianism and Partial Observability

Epistemic POMDP as a special class of POMDP. As noted by Ghosh et al. (2022, Appendix A), the Bayesian objective (Eq. 4) can be cast as a POMDP (Cassandra et al., 1994). We adapt their proof here. The POMDP's state space is $\mathcal{S}^+ = \mathcal{S} \times \mathfrak{M}_\mathcal{D}$ with the same action space $\mathcal{A}$, where $\mathfrak{M}_\mathcal{D} = \text{supp}(\mathbb{P}_\mathcal{D})$. The joint reward–transition function in the POMDP is

$$\mathbb{P}(r_{t+1}, s^+_{t+1} \mid s^+_t, a_t) = \mathbb{1}(m_{t+1} = m_t) m_t(r_{t+1}, s_{t+1} \mid s_t, a_t),$$

where the initial state is $s^+_0 := (s_0, m_0)$ with $s_0 \sim \rho$ and $m_0 \sim \mathbb{P}_\mathcal{D}$. It is partially observable because the agent only observes $s_t \in s^+_t$, while the model $m_t \equiv m_0 \in \mathfrak{M}_\mathcal{D}$ remains hidden but fixed throughout each episode.

D NEUBAY Algorithm Details

RL loss function. In recurrent off-policy RL, optimization is typically performed on full trajectories (rather than i.i.d. transition tuples) for compute efficiency. In our setting, each training trajectory is a *concatenation* of a real prefix and an imagined rollout: starting from an initial history $h_t \in \mathcal{D}$, a world model and the policy generate future steps until truncation at $t' \leq T$. Formally, let

$$\tau = (s_{0:t} \oplus s_{t+1:t'}, a_{0:t-1} \oplus a_{t:t'-1}, r_{1:t} \oplus \hat{r}_{t+1:t'}, d_{1:t} \oplus \hat{d}_{t+1:t'}),$$

where $\oplus$ denotes *concatenation*. Our RL loss on τ balances contributions from real and imagined segments:

$$L(Q_\omega, \pi_\nu; \tau, \kappa) := \frac{\kappa}{t} \sum_{j=0}^{t-1} l(Q_\omega, \pi_\nu; h_j, a_j, r_{j+1}, d_{j+1}, s_{j+1}) + \frac{1-\kappa}{t'-t} \sum_{j=t}^{t'-1} l(Q_\omega, \pi_\nu; \hat{h}_j, \hat{a}_j, \hat{r}_{j+1}, \hat{d}_{j+1}, \hat{s}_{j+1}). \tag{22}$$

Here, $\hat{h}_j$ denotes the imagined history (with $\hat{h}_t = h_t$) and $\kappa \in (0, 1)$ is the real data ratio. The per-step loss $l(Q_\omega, \pi_\nu; h_j, a_j, r_{j+1}, d_{j+1}, s_{j+1})$ is standard off-policy loss without conservatism, such as DQN (Mnih et al., 2013) for discrete control and SAC (Haarnoja et al., 2018a) for continuous control.

Planning stopping criteria. In our rollout subroutine (Algorithm 2), a rollout finishes when any of three conditions hold:

$$\text{done}_{t+1} := (U_{\boldsymbol{\theta}}(\hat{s}_t, \hat{a}_t) > \mathcal{U}(\zeta)) \vee (t+1 \geq T) \vee f_{\text{term}}(\hat{s}_t, \hat{a}_t, \hat{s}_{t+1}). \tag{23}$$

1. Uncertainty truncation: as described in Sec. 4.2, instead of enforcing a fixed horizon H, we truncate rollouts adaptively using an uncertainty threshold calibrated on the real dataset. This allows planning to extend as long as the model remains confident.
2. Timout truncation: to remain consistent with test-time evaluation, we impose a hard cap at the environment's maximum episode length T, regardless of rollout length.
3. Ground-truth termination: we retain the environment's rule-based terminal function to provide true terminal signals $\hat{d}_{t+1}$, following prior model-based RL methods (Yu et al., 2020). Including this prior knowledge makes our algorithm directly comparable to model-based baselines, which are our main focus.

Importantly, only the terminal signal disables bootstrapping in RL, while both uncertainty- and timeout-based truncations preserve bootstrapping[8], which aligns with the Bayesian objective.

E Dataset Details

The bandit dataset. As introduced in Sec. 3, we construct a skewed two-armed bandit dataset. We collect 10 trajectories of length $T = 100$, yielding $|\mathcal{D}| = 1000$ action–reward pairs. We use one-hot encoding on actions as inputs for reward models and agents. The dataset is split into training and validation sets with a 4:1 ratio. Since $\mathcal{D}$ only covers arm 0 with $p^*_0 = 0.5$, the true parameter of arm 1, p^*_1, is completely unseen.

[8] https://gymnasium.farama.org/tutorials/gymnasium_basics/handling_time_limits/.

At test time, we vary $p_1^* \in \{0.01, 0.3, 0.55, 0.7, 0.99\}$, where each choice defines a distinct bandit problem. Each problem is evaluated over 20 independent episodes, and all problems are assessed in parallel, yielding $5 \times 20 = 100$ evaluation runs. The normalized return during test time is computed by: $\frac{1}{T}\sum_{t=0}^{T-1} r_{t+1}$, where $r_{t+1} \sim \mathcal{B}(p^*_{a_t})$.

D4RL locomotion benchmark. We evaluate on the standard D4RL locomotion benchmark (Fu et al., 2020), comprising 12 datasets formed by the Cartesian product of tasks (halfcheetah, hopper, walker2d) and dataset types (random-v2, medium-v2, medium-replay-v2, medium-expert-v2). This benchmark is the most widely used in offline RL research. The underlying environments are OpenAI Gym tasks: HalfCheetah-v2 ($\mathcal{S} \subset \mathbb{R}^{17}, \mathcal{A} \subset \mathbb{R}^6$), Hopper-v2 ($\mathcal{S} \subset \mathbb{R}^{11}, \mathcal{A} \subset \mathbb{R}^3$), and Walker2d-v2 ($\mathcal{S} \subset \mathbb{R}^{17}, \mathcal{A} \subset \mathbb{R}^6$). The maximum episode step T is 1000. Hopper-v2 and Walker2d-v2 have termination functions, while HalfCheetah-v2 does not.

Dataset sizes vary: 100k-200k transitions for medium-replay, 1M for random and medium, and 2M for medium-expert. These datasets also differ qualitatively. The random dataset is collected with a uniformly random policy; medium-replay corresponds to the replay buffer of an agent trained to a medium-level policy; medium itself is generated directly from a medium-level policy; and medium-expert is a mixture of trajectories from both a medium policy and an expert policy. Based on these properties, we categorize random as *low-quality*, medium-replay and medium-expert as *moderate coverage*, and medium as *narrow coverage*.

Performance is reported in terms of normalized scores, following the D4RL and NeoRL conventions:

$$\text{normalized score} = \frac{\text{score} - \text{random score}}{\text{expert score} - \text{random score}} \times 100.$$

It is worth noting that the expert policies in these locomotion tasks are not strictly optimal. As a result, it is possible for algorithms to achieve normalized scores greater than 100 (e.g., around 120). Therefore, we consistently categorize all medium-$*$ datasets as medium-quality.

NeoRL locomotion benchmark. The locomotion benchmark in the NeoRL (Qin et al., 2022) has been widely used in recent model-based offline RL methods. The setup closely mirrors the D4RL locomotion benchmark, with nearly identical environments: HalfCheetah-v3 ($\mathcal{S} \subset \mathbb{R}^{18}, \mathcal{A} \subset \mathbb{R}^6$), Hopper-v3 ($\mathcal{S} \subset \mathbb{R}^{12}, \mathcal{A} \subset \mathbb{R}^3$), and Walker2d-v3 ($\mathcal{S} \subset \mathbb{R}^{18}, \mathcal{A} \subset \mathbb{R}^6$). Compared to D4RL, NeoRL increases the state dimensionality by one in each environment.

For each environment, NeoRL provides three datasets (Low, Medium, High), collected using policies of the corresponding performance levels. This leads to 9 datasets in total. Compared to their D4RL counterparts, these policies are more deterministic, leading to smaller data coverage, which we categorize as *narrow coverage* in this paper. Dataset sizes range from roughly 200k to 1M transitions. To avoid ambiguity, throughout the paper we denote NeoRL datasets with capitalized environment names and the v3 suffix (e.g., *HalfCheetah-v3-Medium*), while D4RL datasets are written in lowercase with the v2 suffix (e.g., *halfcheetah-medium-v2*).

D4RL Adroit benchmark. The Adroit benchmark is widely regarded as substantially more challenging than locomotion tasks. It involves controlling a 28-DoF robotic arm to perform high-dimensional manipulation tasks: Pen ($\mathcal{S} \subset \mathbb{R}^{45}, \mathcal{A} \subset \mathbb{R}^{24}$, $T = 100$), Door ($\mathcal{S} \subset \mathbb{R}^{39}, \mathcal{A} \subset \mathbb{R}^{28}$, $T = 200$), Hammer ($\mathcal{S} \subset \mathbb{R}^{46}, \mathcal{A} \subset \mathbb{R}^{26}$, $T = 200$). Among these, Pen includes a termination function, while Door and Hammer do not. Rewards combine a dense component based on distances with a sparse component that grants a large bonus once a distance threshold is satisfied.

In addition to the high dimensionality and sparsity of rewards, most Adroit datasets are either small or of limited quality. The human demonstration datasets (human-v1) contain only 5k–10k transitions, whereas the cloned datasets (cloned-v1), constructed by mixing behavior cloning with human demonstrations, contain 500k–1M transitions. This leads to 6 datasets in total. Accordingly, we categorize human datasets as *narrow coverage* and cloned datasets as *moderate coverage*.

Dataset performance levels vary significantly, as shown in the $\pi_{\mathcal{D}}$ column of Tab. 2. Pen-human-v1 and pen-cloned-v1 achieve normalized scores in the range of 70–90, whereas door-human-v1, door-cloned-v1, hammer-human-v1, and hammer-cloned-v1 yield scores below 10. Accordingly, we categorize pen-$*$ datasets as *medium-quality*, and the remaining ones as *low-quality*. This disparity explains why most algorithms fail to achieve meaningful results on the door and hammer datasets.

Finally, we exclude the relocate tasks from our experiments, as both prior baselines and our method consistently obtain near-zero scores in this setting.

D4RL AntMaze benchmark. AntMaze is a particularly challenging benchmark in D4RL due to its sparse reward structure. It involves controlling a MuJoCo Ant ($\mathcal{S} \subset \mathbb{R}^{29}, \mathcal{A} \subset \mathbb{R}^{8}$) to navigate in different maze layouts (umaze, medium, large). The agent receives reward 1 only when reaching a goal position in the X–Y plane within a threshold, after which the episode terminates, so the maximum return is 1. Goals are randomly sampled from a small region but remain unobserved to the agent, making evaluation noisy for RL algorithms. The maximum episode length is $T = 700$ for umaze and $T = 1000$ for medium and large mazes.

For each layout, two datasets (play and diverse) are provided, each containing 1M transitions. The *diverse* variants introduce a wider set of start positions compared to *play*. However, all AntMaze datasets are generated by a hierarchical controller: a high-level breadth-first search (BFS) planner selects waypoints, which are then executed by a low-level learned policy. As a result, trajectories are highly structured and largely restricted to narrow corridors that connect goals without collisions. This planner-driven generation induces narrow coverage, a property noted in prior work (Zhang et al., 2024a).

Performance in AntMaze is generally low relative to other D4RL domains. In particular, umaze-diverse has an average score of only 1.0, which we categorize as *low-quality*. The remaining AntMaze datasets achieve scores above 10.0, which we categorize as *medium-quality* (though they could also be viewed as low-quality when considered outside the AntMaze domain).

Following prior model-based RL work (LEQ (Park and Lee, 2025)[9], ADMPO (Lin et al., 2025)[10]), we adopt the same terminal functions in AntMaze. As in LEQ, we also shift the reward by -1.0. Since termination reveals information about the reward function (termination implies success), we do not compare against other model-based or model-free algorithms. Instead, our baselines in AntMaze are the methods reported in LEQ and ADMPO.

F Implementation Details

Our implementation is built on JAX (Bradbury et al., 2018) and Equinox (Kidger and Garcia, 2021)[11].

F.1 World Model Ensemble and Planning

World model ensemble. Throughout our experiments, we train an ensemble of 128 MLPs on each dataset. Following MOPO (Yu et al., 2020), for continuous control tasks, we rank the models by MSE on a validation set consisting of 1000 held-out transitions from $\mathcal{D}$. For the bandit task, we rank by negative log-likelihood (NLL) that captures uncertainty, since the true reward follows Bernoulli distribution. The top N models are then selected to form the world model ensemble $\mathbf{m}_{\boldsymbol{\theta}}$, which remains fixed during subsequent policy training. In our main experiments (Sec. 5.1), we use $N = 100$. For the sensitivity analysis (Sec. 5.2), we vary this number to 5 and 20.

Following MOBILE (Sun et al., 2023; Sun, 2023)[12], each MLP $m_\theta(s, a)$ has 4 hidden layers with a width of 200 for the locomotion benchmarks, and 5 layers with a width of 400 for Adroit benchmark. As described in Sec. 4.1, we apply LayerNorm after each hidden layer; this design choice is further ablated in Sec. 5.2. The basic block consists of (Linear $\rightarrow$ LayerNorm $\rightarrow$ leaky ReLU) when LayerNorm is enabled, and (Linear $\rightarrow$ leaky ReLU) when it is disabled. For the bandit task, we use a small reward-model ensemble that has 2 hidden layers with a width of 16. The weights of each MLP are initialized independently using the default `equinox.nn.Linear` scheme, i.e., $\mathrm{Unif}[-\frac{1}{\sqrt{\dim_{\text{in}}}}, \frac{1}{\sqrt{\dim_{\text{in}}}}]$, where $\dim_{\text{in}}$ denotes the input feature dimension.

For continuous control tasks, during ensemble training, we sample transition tuples (s, a, r, s') from $\mathcal{D}$ using a batch size of 256 to estimate the MLE loss. The inputs (s, a) and outputs (r, s') are standardized by subtracting the mean and dividing by the standard deviation computed over the

[9] https://github.com/kwanyoungpark/LEQ.
[10] https://github.com/HxLyn3/ADMPO.
[11] https://github.com/patrick-kidger/equinox.
[12] https://github.com/yihaosun1124/mobile.

dataset; during inference, predictions are inverse-transformed to restore the original scale. We use AdamW optimizer (Loshchilov and Hutter, 2019) with a weight decay coefficient of 5×10^{-5}, and a learning rate of 1×10^{-3} for locomotion tasks and 3×10^{-4} for Adroit tasks. Training is terminated early if the validation MSE fails to improve by more than 0.01 relative within five consecutive epochs, following the early stopping procedure in MOBILE. In the bandit task, the model learning rate is 1×10^{-3}, batch size is 128, and improvement threshold is 0.001 absolute.

Planning. At the start of planning, following Sec. 4.2, we sample a batch of initial histories $h_t \sim \mathcal{D}$, drawn uniformly across time steps. The batch size is fixed at 100, regardless of the ensemble size N. Since the values of N used in our experiments $\{5, 20, 100\}$ are divisors of 100, we distribute the histories evenly across ensemble members. The recurrent policy π_ν initializes its hidden state z_t with h_t and then interacts with each world model to generate corresponding rollout until reaching the stopping criterion described in Sec. D. The uncertainty threshold $\mathcal{U}(\zeta)$ for truncation is fixed at $\zeta = 1.0$ in our main experiments and varied to $\{0.9, 0.99, 0.999\}$ in the sensitivity analysis. The entire planning process is parallelized on a GPU and thus introduces only negligible time costs.

For the bandit task, since there are no transition dynamics and hence no compounding error, we directly optimize the Bayesian objective: planning starts at $t = 0$ and truncates at $t = T$.

F.2 Recurrent Off-Policy RL

Off-policy loss implementation. For continuous control tasks, we follow a recent recurrent off-policy RL algorithm RESeL (Luo et al., 2024a) to use REDQ (Chen et al., 2021b) as the per-step loss for $l(\cdot)$, used in the overall RL loss defined by Eq. 22. REDQ builds on SAC (Haarnoja et al., 2018a,b)[13], maintaining an ensemble of 10 critic MLPs and sampling 2 of them to form bootstrapped targets in the critic loss, while the actor maximizes the average Q-value over all ensemble members. Although REDQ substantially reduces value overestimation, it does not deliberately *underestimate* values. Moreover, REDQ and RESeL were designed for *online* RL; in our framework, they are thus *not* considered conservative algorithms.

For the bandit task, we adopt dueling DQN (Wang et al., 2016) following memoroid (Morad et al., 2024) as the discrete control algorithm. Exploration is ϵ-greedy, annealed from 1.0 to 0.1 over the first 10% of gradient steps.

Agent architecture implementation. As introduced in Sec. 4.3, our agent consists of a recurrent actor $\pi_\nu : \mathcal{H}_t \to \Delta(\mathcal{A})$ and a recurrent critic $Q_\omega : \mathcal{H}_t \times \mathcal{A} \to \mathbb{R}^{10}$. The critic outputs an ensemble of 10 Q-values, following the REDQ design adopted in RESeL. Both actor and critic maintain their own RNN encoders, $\nu_\phi : \mathcal{H}_t \to \mathcal{Z}$ and $\omega_\phi : \mathcal{H}_t \to \mathcal{Z}$, which share the same architecture but are optimized independently. As mentioned earlier, we adopt the *memoroid* framework (Morad et al., 2024)[14] to use the linear recurrent unit (LRU) (Orvieto et al., 2023) as the backbone encoder for both ν_ϕ and ω_ϕ. The actor and critic MLP heads have 2 or 3 hidden layers with a hidden size of 256. For the bandit task, there is only a critic MLP head with 2 hidden layers.

The LRU begins with a nonlinear preprocessing layer that projects the raw input history h_t into a 256-dimensional feature space (preserving the time dimension). This is followed by a stack of two LRU layers, each performing a linear recurrence update parameterized by a complex-valued diagonal matrix. Each layer maintains a hidden state of size 128, resulting in a recurrent representation $z_t \in \mathbb{C}^{2\times 128}$. From this recurrent state, the model produces a real-valued output vector $\tilde{z}_t \in \mathbb{R}^{128}$ via a nonlinear projection. During training and inference, $\tilde{z}_t$ is fed into the actor or critic MLP heads, while the complex hidden state z_t is preserved for recurrent updates during policy inference.

We fix the recurrent architecture, including hidden sizes, across all experiments. However, NEUBAY is compatible with any RNN encoder, and we leave a study of architectural variations to future work.

Tape-based batching. Classic recurrent RL relies on *segment-based batching* (Ni et al., 2022), where sequences are padded to a fixed length, forming 3D tensors of shape $(\text{batch size}, \text{sequence length}, \text{dim})$ with NaN masks for shorter sequences. This wastes memory

[13] Our REDQ implementation is adapted from the SAC-N Equinox codebase: https://github.com/Howuhh/sac-n-jax.

[14] https://github.com/proroklab/memoroids.

and lowers sample efficiency. Memoroid (Morad et al., 2024) introduces *tape-based batching*, which exploits the monoid algebra of linear RNNs such as LRUs. Instead of padding, variable-length sequences are concatenated into a single 2D "tape", making the effective batch size equal to the sum of raw sequence lengths. Inline resets of hidden states prevent leakage across sequences within a tape (Lu et al., 2023), and computation over the tape is parallelized using associative scan in JAX (Bradbury et al., 2018). To enable JIT compilation, a fixed tape length is enforced by dropping any trailing timesteps that exceed this length. We refer readers to Morad et al. (2024) for full details.

In our implementation, we adopt tape-based batching and we set the tape length to be larger than the maximal episode length T in each task. To reduce computation time, we choose a relatively small tape length, similar to prior work in online POMDPs (Morad et al., 2024; Luo et al., 2024a).

Training hyperparameters. Tab. 4 summarizes the modules and hyperparameters fixed in our experiments. Consistent with IQL (Kostrikov et al., 2022) and MOBILE (Sun et al., 2023), we apply cosine learning rate decay only to the actor network (both the RNN encoder and MLP head), but not to the critic, in order to promote stability during the later stages of training.

The REDQ (SAC) entropy coefficient α is auto-tuned with a target entropy of $-\dim(\mathcal{A})$ (Haarnoja et al., 2018b) for all datasets except for D4RL and NeoRL Hopper datasets and D4RL AntMaze domain. In the D4RL hopper domain, we find our algorithm is sensitive to α, consistent with prior recurrent RL work (Luo et al., 2024a). To address this, we follow MOBILE and fix $\alpha = 0.2$ across all hopper datasets (four in D4RL and three in NeoRL), without further tuning.

In AntMaze, sparse-reward navigation makes SAC sensitive to the choice of α, e.g., ADMPO (Lin et al., 2025) uses a fixed $\alpha = 0.05$ and disables the entropy term backup in critic loss. We follow their insights but instead tune the target entropy $\in \{-\dim(\mathcal{A}), -5\dim(\mathcal{A}), -10\dim(\mathcal{A})\}$. We find $-10\dim(\mathcal{A})$ works best overall and report it in our main results.

For the bandit task, we sweep the RNN encoder learning rate $\eta_\phi \in \{1 \times 10^{-6}, 3 \times 10^{-6}, 1 \times 10^{-5}, 3 \times 10^{-5}, 1 \times 10^{-4}\}$ in the critic network. Consistent with observations from ReSEL (Luo et al., 2024a) and our continuous control results, a small learning rate is crucial for stable training. We use $\eta_\phi = 3 \times 10^{-6}$ in our bandit experiments.

Table 4: Fixed hyperparameters used in our recurrent agents. The last block (actor and policy entropy) is only used in continuous control.

Module or Hyperparameter	Value
Actor and critic RNN encoders	2-layer LRUs (Orvieto et al., 2023)
RNN hidden state size	256
Actor and critic heads	2-layer MLPs (3 layers in Adroit)
Basic block of MLP head	(Linear $\rightarrow$ LayerNorm $\rightarrow$ leaky ReLU)
MLP head hidden size	256
Batch size (i.e., tape length)	2048 (1024 in Adroit, 1000 in bandit)
Update-to-data (UTD) ratio	0.05 (0.02 in bandit)
Gradient steps	2M (3M in AntMaze, 20k in bandit)
Replay buffer size	Full size, i.e., (60M in AntMaze, 1M in bandit, 40M otherwise)
Discount factor γ	0.99
Critic head's learning rate	1×10^{-4}
Gradient norm clipping	1000 (10000 in Adroit, 1 in bandit and AntMaze)
Actor head's learning rate	1×10^{-4}
Actor's learning rate decay	Cosine decay to 0.0
Entropy coef. α's learning rate	1×10^{-4}
Entropy coef. α	Auto-tuned with target $-\dim(\mathcal{A})$ (AntMaze uses $-10\dim(\mathcal{A})$ while Hopper uses fixed $\alpha = 0.2$)

Table 5: Best hyperparameters per dataset in the D4RL locomotion benchmark. We sweep $\eta_\phi \in \{3 \times 10^{-7}, 1 \times 10^{-6}, 3 \times 10^{-6}, 1 \times 10^{-5}, 3 \times 10^{-5}\}$ and $\kappa \in \{0.05, 0.5, 0.8\}$.

Dataset	RNN encoder lr η_ϕ	Real data ratio κ
halfcheetah-random-v2	3×10^{-5}	0.8
hopper-random-v2	1×10^{-5}	0.5
walker2d-random-v2	3×10^{-5}	0.5
halfcheetah-medium-replay-v2	1×10^{-5}	0.05
hopper-medium-replay-v2	3×10^{-7}	0.5
walker2d-medium-replay-v2	1×10^{-6}	0.5
halfcheetah-medium-v2	3×10^{-5}	0.8
hopper-medium-v2	1×10^{-6}	0.8
walker2d-medium-v2	3×10^{-6}	0.5
halfcheetah-medium-expert-v2	3×10^{-5}	0.8
hopper-medium-expert-v2	3×10^{-6}	0.5
walker2d-medium-expert-v2	1×10^{-6}	0.8

Table 6: Best hyperparameters per dataset in the NeoRL locomotion benchmark. We sweep $\eta_\phi \in \{3 \times 10^{-7}, 1 \times 10^{-6}, 3 \times 10^{-6}, 1 \times 10^{-5}, 3 \times 10^{-5}\}$ and $\kappa \in \{0.5, 0.8\}$.

Dataset	RNN encoder lr η_ϕ	Real data ratio κ
HalfCheetah-v3-Low	1×10^{-5}	0.8
Hopper-v3-Low	3×10^{-6}	0.8
Walker2d-v3-Low	3×10^{-7}	0.5
HalfCheetah-v3-Medium	3×10^{-5}	0.5
Hopper-v3-Medium	3×10^{-6}	0.5
Walker2d-v3-Medium	3×10^{-7}	0.8
HalfCheetah-v3-High	3×10^{-5}	0.5
Hopper-v3-High	3×10^{-6}	0.5
Walker2d-v3-High	3×10^{-7}	0.8

Table 7: Best hyperparameters per dataset in the D4RL Adroit benchmark. We sweep $\eta_\phi \in \{1 \times 10^{-6}, 3 \times 10^{-6}, 1 \times 10^{-5}, 3 \times 10^{-5}, 1 \times 10^{-4}\}$ and $\kappa \in \{0.5, 0.8\}$. For the remaining Adroit datasets, all hyperparameter settings yield near-zero performance.

Dataset	RNN encoder lr η_ϕ	Real data ratio κ
pen-human-v1	3×10^{-5}	0.5
pen-cloned-v1	1×10^{-5}	0.8
hammer-cloned-v1	1×10^{-5}	0.5

Table 8: Best hyperparameters per dataset in the D4RL AntMaze benchmark. We sweep $\eta_\phi \in \{1 \times 10^{-6}, 3 \times 10^{-6}, 1 \times 10^{-5}, 3 \times 10^{-5}\}$ and $\kappa \in \{0.5, 0.8, 0.95\}$. For the large maze datasets, all hyperparameter settings yield near-zero performance.

Dataset	RNN encoder lr η_ϕ	Real data ratio κ
antmaze-umaze-v2	3×10^{-6}	0.95
antmaze-umaze-diverse-v2	3×10^{-6}	0.95
antmaze-medium-play-v2	1×10^{-5}	0.95
antmaze-medium-diverse-v2	1×10^{-6}	0.95

F.3 Computation Details

All experiments are conducted on a single NVIDIA L40S GPU. In the locomotion benchmarks, the world model ensemble with $N = 100$ contains fewer than 15M parameters in total. The recurrent encoder has fewer than 600k parameters, the actor MLP head fewer than 200k, and the critic MLP head (with an ensemble size of 10) fewer than 1.5M. We parallelize both the world model and

critic ensembles using `jax.vmap`, which makes ensemble computation efficient for both training and inference. Thus, ensembling does not pose a computational bottleneck.

For each random seed, we train a world model ensemble and then train the agent on that ensemble, making evaluation more robust to variability across the world ensembles. The world ensemble is pretrained on the offline dataset and its checkpoint is saved without further fine-tuning. Training a world ensemble takes within 12-24 GPU hours (with three runs in parallel), but this is a one-time cost. Unlike agent training, where we sweep hyperparameters, the world ensemble is trained only once per dataset, so its computational overhead is minimal in comparison. The agent training takes within 18 GPU hours (also with three runs in parallel).

G Further Results and Discussion

G.1 Full Benchmarking Results

Table 9: Comparison of offline RL methods on the **NeoRL locomotion** benchmark. We report mean normalized scores for all baselines, with ±std for competitive baselines. The **best mean score** is bolded, and marked methods are statistically similar under a t-test. Our results use 6 seeds, each evaluated at the final step with 20 episodes.

	Model-free		Conservative model-based							Bayesian-inspired		Ours
Dataset	EDAC	CQL	MOPO	COMBO	MOBILE	LEQ	ADMPO	ScorePen	VIPO	MAPLE	MoDAP	NeuBAY
hc-Low	31.3	38.2	40.1	32.9	54.7±3.0	33.4±1.6	52.8±1.2	49.6±1.2	**58.5**±0.1	33.4	53.9±1.1	53.1±1.1
hp-Low	18.3	16.0	6.2	17.9	17.4±3.9	24.2±2.3	22.3±0.1	21.1±2.3	**30.7**±0.3	22.7	26.1±4.7	30.3±2.9
wk-Low	40.2	44.7	11.6	31.7	37.6±2.0	65.1±2.3	55.9±3.8	51.4±1.4	**67.6**±0.7	33.9	51.3±7.8	43.9±5.9
hc-Med	54.9	54.6	62.3	50.8	77.8±1.4	59.2±3.9	69.3±1.7	77.4±1.0	80.9±0.2	69.5	81.0±2.3	**81.1**±0.8
hp-Med	44.9	64.5	1.0	56.3	51.1±13.3	**104.3**±5.2	51.5±5.0	90.9±1.3	66.3±0.2	27.7	44.2±15.3	95.7±11.5
wk-Med	57.6	57.3	39.9	53.8	62.2±1.6	45.2±19.4	70.1±2.4	65.8±1.6	**76.8**±0.1	40.7	70.8±3.1	50.5±8.8
hc-High	81.4	77.4	65.9	62.2	83.0±4.6	71.8±8.0	84.0±0.8	81.4±1.0	**89.4**±0.6	—	84.1±8.3	68.3±23.6
hp-High	52.5	76.6	11.5	63.2	87.8±26.0	95.5±13.9	87.6±4.9	86.3±1.3	**107.7**±0.5	—	52.4±3.2	96.8±7.7
wk-High	75.5	75.3	18.0	71.8	74.9±3.4	73.7±1.1	**82.2**±1.9	78.0±1.8	81.7±1.0	—	73.6±2.8	62.7±14.5
AVG	50.7	56.1	28.5	49.0	60.7	63.6	64.0	66.9	**73.3**	—	59.7	64.7

Table 10: Comparison of offline model-based RL methods on the **D4RL AntMaze** benchmark. We report mean normalized scores for all baselines, with ±std for competitive baselines. The **best mean score** is bolded, and marked methods are statistically similar under a t-test. Our results use 6 seeds, each evaluated at the final step with 100 episodes. In addition, we include the average dataset performance, denoted as $\pi_{\mathcal{D}}$, to provide a reference for **data quality**.

Dataset	$\pi_{\mathcal{D}}$	CBOP	MOBILE	MOBILE†	ADMPO	LEQ	NeuBAY
umaze	28.8	0.0±0.0	0.0±0.0	77.0	88.4±1.2	**94.4**±6.3	66.1±20.1
umaze-diverse	1.0	0.0±0.0	0.0±0.0	20.4	**81.7**±8.6	71.0±12.3	74.4±12.1
medium-play	19.6	0.0±0.0	0.0±0.0	64.6	23.9±6.3	**58.8**±33.0	12.8±19.2
medium-diverse	11.3	0.0±0.0	0.0±0.0	1.6	24.1±5.7	**46.2**±23.2	19.4±12.4
large-play	10.5	0.0±0.0	0.0±0.0	2.6	8.3±4.1	**58.6**±9.1	0.0±0.0
large-diverse	10.6	0.0±0.0	0.0±0.0	7.2	0.0±0.0	**60.2**±18.3	0.5±1.1
AVG	N/A	0.0	0.0	28.9	37.7	**64.9**	28.9

Source of benchmarked baseline results. For the D4RL locomotion benchmark in Tab. 1, we adopt the results of CQL, EDAC, and MOPO from the MOBILE paper (Sun et al., 2023, Table 1), where the MOPO results correspond to the tuned variant (denoted as MOPO* therein). Results for the remaining baselines are taken directly from their respective original publications. For MAPLE, we report the improved variant that employs an ensemble size of 142 (instead of the default 14) as described in (Chen et al., 2021c), to enable a fairer comparison with our method, which uses an ensemble size of 100. Finally, we note that prior works may differ in the total number of gradient steps used for training compared to ours (2M steps in this benchmark); for example, MOPO, MOBILE, ADMPO, and VIPO report results after 3M steps, while LEQ and SUMO use 1M steps. We retain these numbers as reported, since we believe this setting reflects the most faithful comparison with previously published results.

For the NeoRL locomotion benchmark in Tab. 9, we use EDAC results from the MOBILE paper (Sun et al., 2023), CQL and MOPO from the NeoRL paper (Qin et al., 2022), and COMBO from the MoDAP paper (Choi et al., 2024). All other baselines are taken from their original publications. For MoDAP on Hopper-v3-High, we report the improved result obtained with an ensemble size of 40, as presented by the authors to demonstrate the benefit of larger ensembles.

For the D4RL Adroit benchmark in Tab. 2, we report BC, IQL, and ReBRAC results from the CORL paper (Tarasov et al., 2023b), and MOPO from the MOBILE paper (Sun et al., 2023). Results for other baselines are taken from their original publications. Standard deviations for MOPO and MoMo were not available, so we omit them.

For the D4RL AntMaze benchmark in Tab. 10, we report CBOP and MOBILE from the LEQ paper (Park and Lee, 2025), and a different tuning of MOBILE which we denote as MOBILE† from the ADMPO paper (Lin et al., 2025). Results for other baselines are taken from their original publications. We omit standard deviations for MOBILE† since they are not reported in the ADMPO paper.

Statistical tests for benchmarking results. To assess statistical significance, we conduct Welch's one-sided t-test ($p < 0.05$). In the reported tables, we highlight all methods whose average scores are not significantly different from the best-performing method.

G.2 Further Discussion on Compounding Error

Plotting setup. We provide additional details on the plots in Fig. 5. RMS denotes the root-mean-square, $\mathrm{RMS}(x) = \sqrt{\frac{1}{k}\sum_{i=1}^{k} x_i^2}$ for $x \in \mathbb{R}^k$, which normalizes the ℓ_2 norm to be dimension-invariant. RMSE denotes the root-mean-square error, $\mathrm{RMSE}(x, y) = \mathrm{RMS}(x - y)$ for $x, y \in \mathbb{R}^k$.

For each world ensemble, we collect 100 rollouts (one per ensemble member) and compute compounding error following the common practice (Lambert et al., 2022). Starting from a sampled state $s_0 \sim \mathcal{D}$ (not necessarily an initial state, but treated as the planning root), we draw $m_\theta \sim \mathbf{m}_{\boldsymbol{\theta}}$ and generate trajectories: $\hat{a}_t \sim \mathrm{Unif}[-1, 1]^{|\mathcal{A}|}, (\hat{r}_{t+1}, \hat{s}_{t+1}) \sim m_\theta(\hat{s}_t, \hat{a}_t), (r_{t+1}, s_{t+1}) = m^*(s_t, \hat{a}_t), \forall t \leq T$, where m^* denotes the deterministic ground-truth model in MuJoCo.

Besides the individual trajectories, we also plot a black curve showing the *median* across rollouts at each planning step. For truncated rollouts (due to episode timeout, termination or numerical overflow), we apply forward filling (`pandas.DataFrame.ffill`) so that the median remains well-defined. For the scatter plot, we aggregate all state-action pairs from these rollouts and show the relation between estimated uncertainty $U_{\boldsymbol{\theta}}(\hat{s}_t, \hat{a}_t)$ and the next-state error $\mathrm{RMSE}(\hat{s}_{t+1}, s_{t+1})$.

Further results and discussion. Fig. 6 shows compounding error analyses on the other D4RL halfcheetah datasets, complementing the results on halfcheetah-medium-expert-v2 in Fig. 5. The severity of numerical explosion depends strongly on dataset coverage: the medium dataset, collected from a single narrow policy, exhibits far more rollouts with divergence than either the random dataset or the medium-replay dataset, which cover a broader distribution of states. We adopt a random policy for rollouts, as this reflects the exploratory behavior of policies in the early stages of training.

Moreover, for world ensembles with LN, we find that the uncertainty threshold $\zeta = 1.0$ (used in our main experiments) reliably separates severe error regions, since $\zeta = 1.0$ corresponds to the boundary of in-distribution data. While the Spearman rank coefficient, often used to benchmark uncertainty estimation accuracy (Lu et al., 2021), is not particularly high in our case, we argue that this metric is less critical for Bayesian RL. Unlike approaches that penalize with uncertainties at every step, we only use uncertainty as a binary cutoff for truncation. As a result, our method is inherently more robust to imperfect uncertainty ranking, requiring only that severe errors lie beyond the $\zeta = 1.0$ boundary.

For the D4RL hopper and walker2d datasets, we did not observe numerical explosion in either world ensemble using the random policy. This stability is largely due to the environments' termination functions f_{term}, which typically halt random-policy rollouts within ≈50 steps, as shown in Fig. 6 for hopper-medium-expert-v2. Without such early termination, we expect the same kind of runaway divergence observed in halfcheetah to be unavoidable, since the worst-case error already exceeds 10^3 for 10 steps.

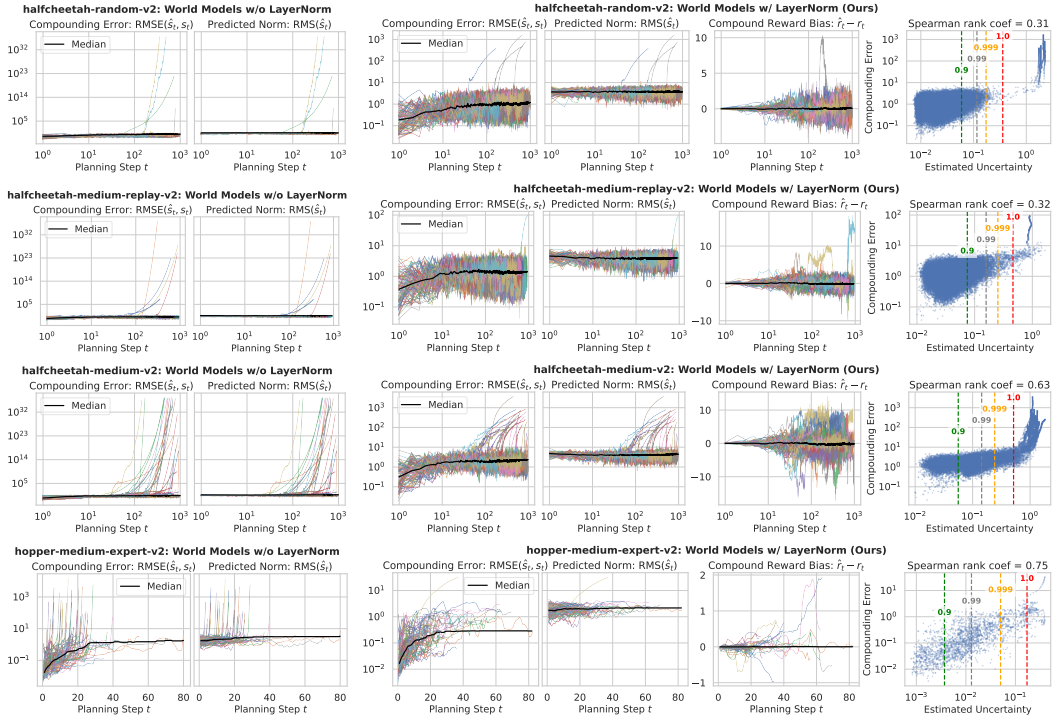

Figure 6: Effect of LayerNorm in world models on halfcheetah-random-v2 (1st row), halfcheetah-medium-replay-v2 (2nd row), halfcheetah-medium-v2 (3rd row), and hopper-medium-expert-v2 (4th row). **Left: without LN. Right: with LN.** We use a uniformly random policy to collect these rollouts, either truncated by the episode limit ($T = 1000$) or `float32` overflow (but not uncertainty threshold), or terminated by the termination function f_{term} (only applicable to hopper-medium-expert-v2). Vertical lines in the rightmost scatter plot mark uncertainty thresholds $\zeta \in \{0.9, 0.99, 0.999, 1.0\}$. The short rollouts in hopper-medium-expert-v2 are mostly caused by termination function.

G.3 Further Discussion on Adaptive Long-Horizon Planning

We report the full ablation results on truncation thresholds to complement Fig. 1, with the maximum rollout horizon for each batch of training rollouts shown in the last columns.

A key observation is that using a quantile threshold of $\zeta = 0.9$ corresponds *roughly* to a horizon cap of 1–10, which mirrors the short fixed horizons commonly adopted in prior work. This shows that such prior choices are not suitable for guiding the design of Bayesian RL, where adaptive long horizons are essential.

In AntMaze, $\zeta = 0.9$ yields the best performance on umaze but performs worst on the other mazes. In the successful umaze case, however, $\zeta = 0.9$ still produces maximum horizons on a scale comparable to larger ζ values (unlike other benchmarks), indicating that long horizons remain the decisive factor. We also find that while $\zeta = 0.9$ initially suffers from severe value overestimation, this effect is later reduced in umaze, which may explain its relative success. By contrast, in medium mazes the overestimation persists throughout training, leading to poor performance.

We further count the number of complete failures (i.e., scores ≤ 5.0) under different thresholds using Tab. 11. **With $\zeta = 0.9$, 16 datasets fail; with $\zeta = 0.99$, 6 datasets fail; and with $\zeta = 0.999$, none fail.** This suggests that a safe range for ζ lies between 0.999 and 1.0. Although $\zeta = 0.999$ often performs similarly to 1.0 (as the resulting adaptive horizons are close), we observe clear advantages of 1.0 on tasks such as D4RL walker2d-random-v2 and pen-cloned-v1. **Thus, we recommend using $\zeta = 1.0$ as a starting point for our algorithm.**

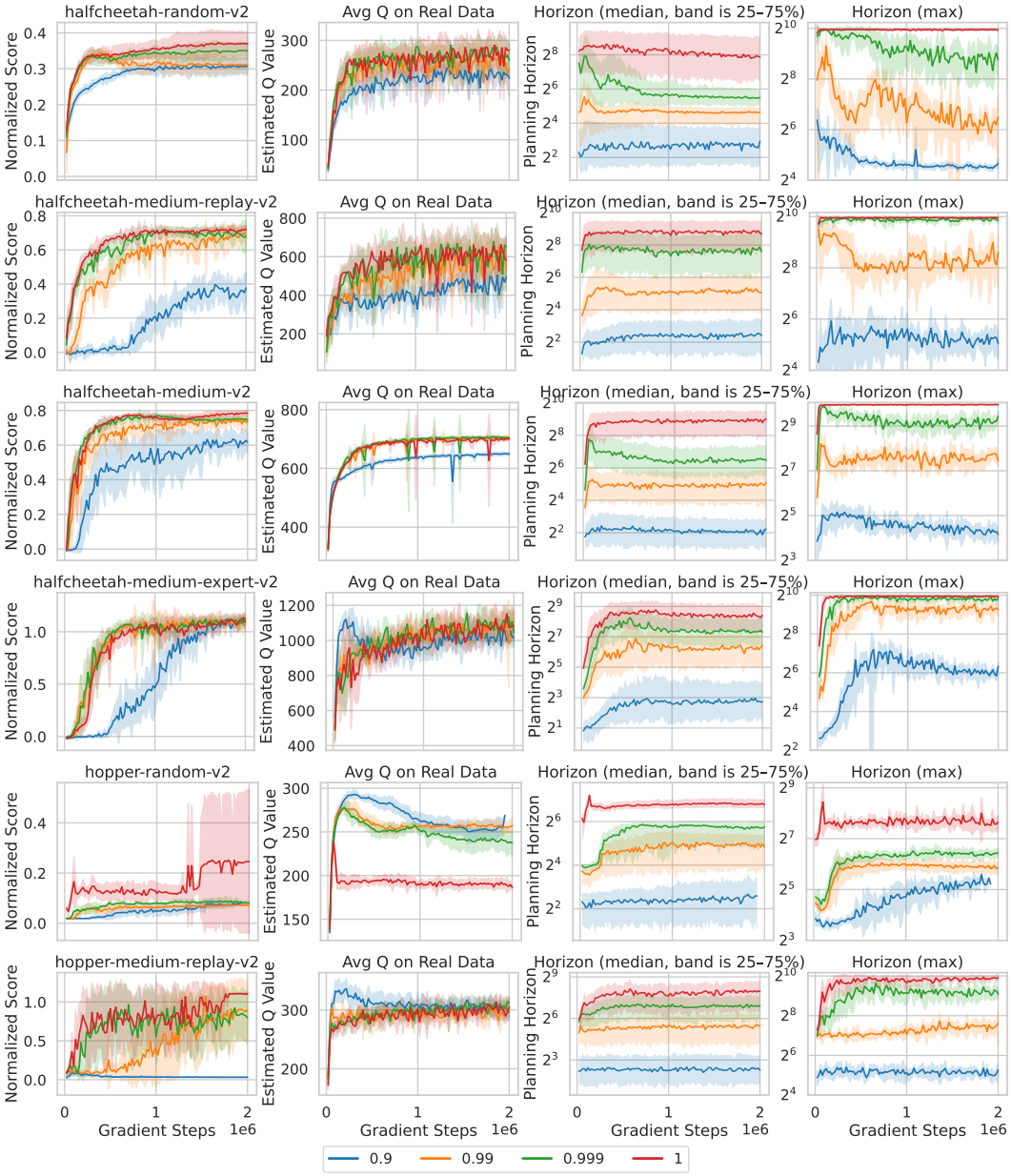

Figure 7: Ablation on the uncertainty quantile ζ for rollout truncation in D4RL locomotion datasets. Results for the remaining datasets are shown in the next figure.

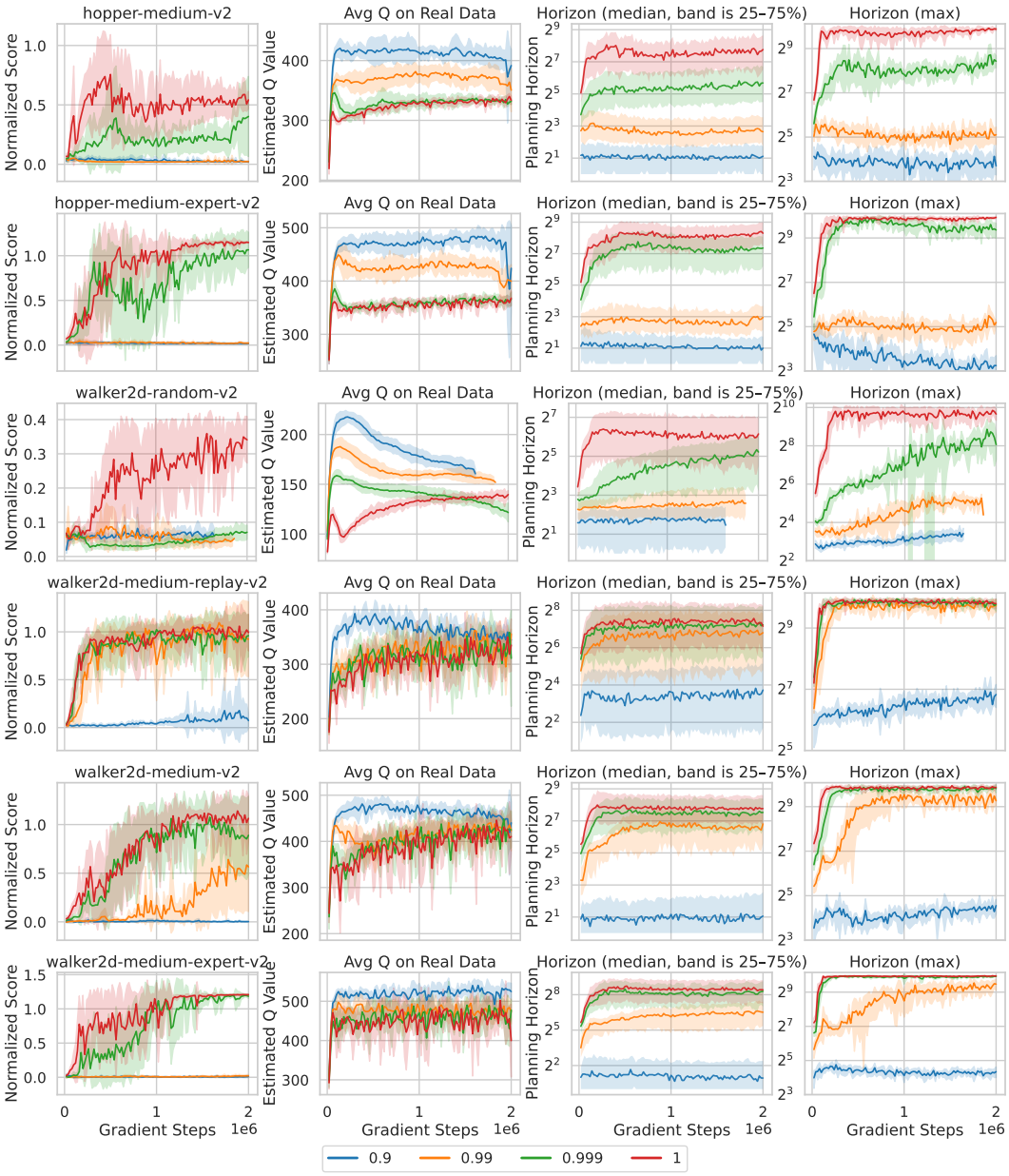

Figure 8: Ablation on the uncertainty quantile ζ for rollout truncation in D4RL locomotion datasets.

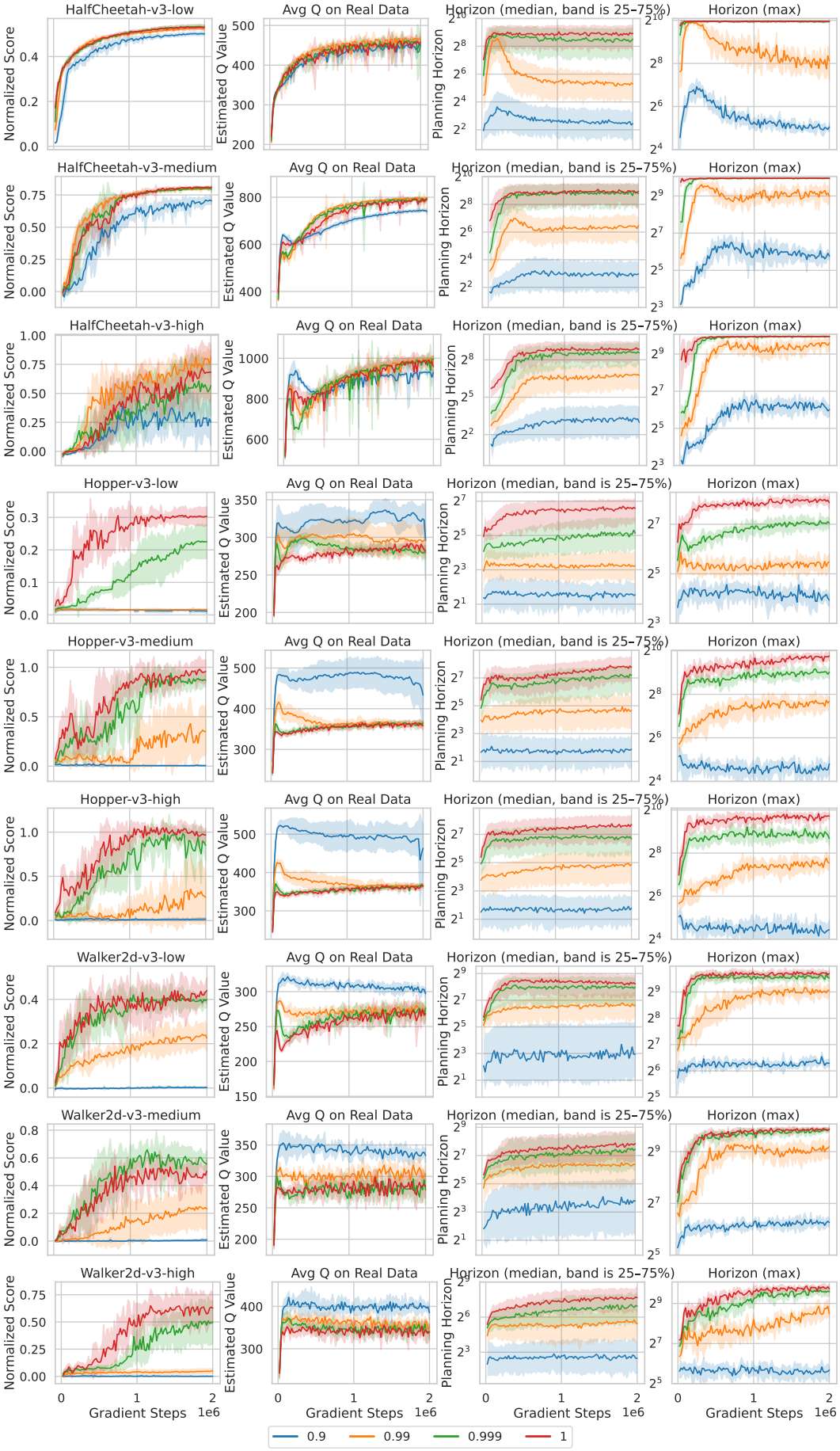

Figure 9: Ablation on the uncertainty quantile ζ for rollout truncation in NeoRL datasets.

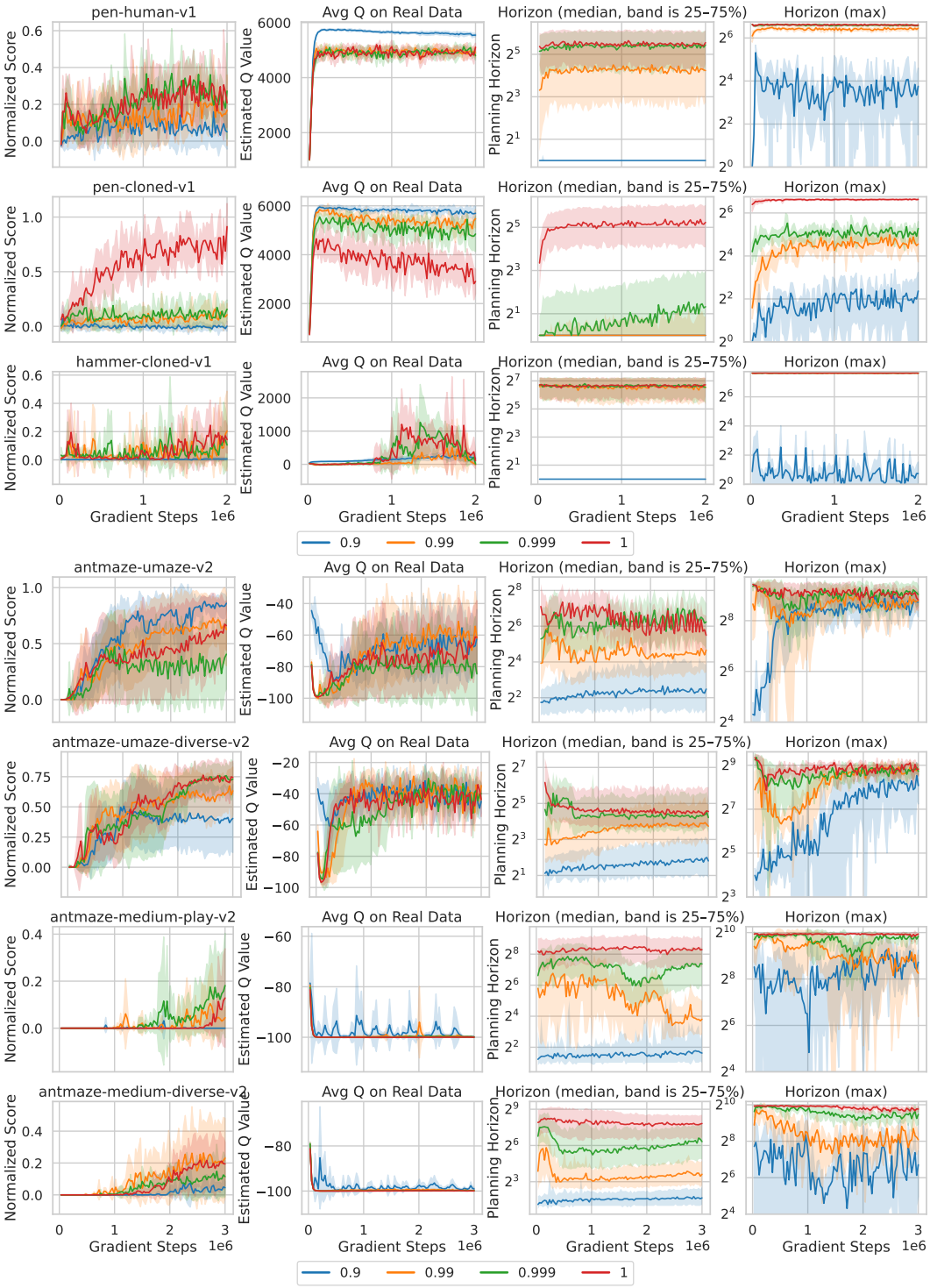

Figure 10: Ablation on the uncertainty quantile ζ for rollout truncation in three D4RL Adroit datasets and four D4RL AntMaze datasets. Adroit benchmark has short maximum episode steps: $T = 100 < 2^7$ in pen and $T = 200 < 2^8$ in hammer, which limits the rollout horizon. Maximum episode steps are $T = 700$ in umaze and $T = 1000$ in medium maze. Results on the remaining Adroit and AntMaze datasets are omitted as our algorithm has near-zero performance. Note that in AntMaze, successful episodes terminate early, so horizon lengths are partially confounded by this effect.

G.4 Sensitivity and Ablation Results

For AntMaze, we consistently observe that using a real data ratio ≤ 0.5 results in poor performance. This is consistent with the findings of LEQ (Park and Lee, 2025), which emphasize the importance of higher ratios (e.g., 0.75) in this domain. In our case, possibly due to the lack of explicit conservatism, NEUBAY requires an even higher ratio of 0.95. This echoes the results of ADMPO (Lin et al., 2025), which also employ a 0.95 ratio in the medium and large AntMaze tasks.

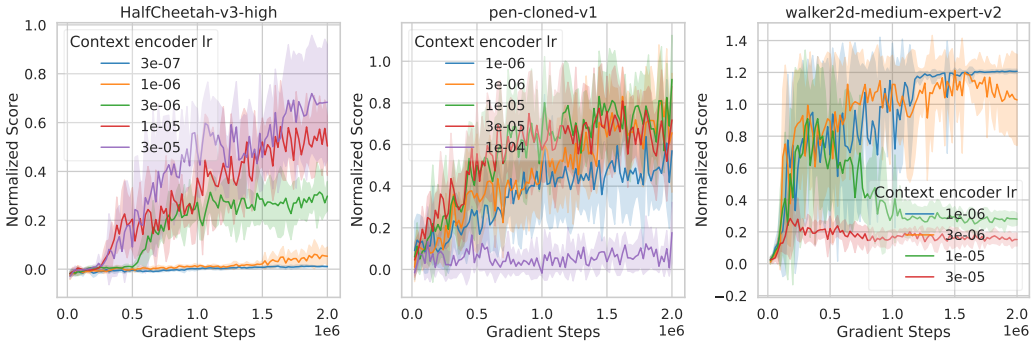

Figure 11: Selective learning curves on datasets where performance is *sensitive* to the **context encoder learning rate**, favoring high (left), medium (middle), and low (right) values.

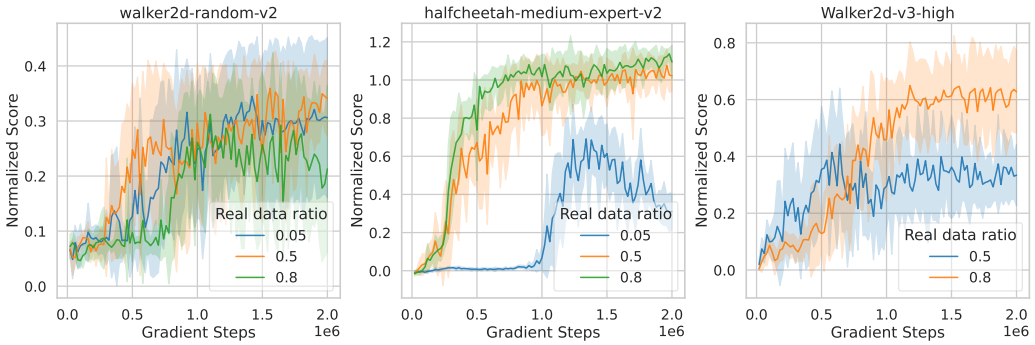

Figure 12: Selective learning curves on datasets where performance is *sensitive* to the **real data ratio**.

Table 11: **Sensitivity and ablation results per dataset.** The highlighted setting (N=100, λ=0.0, ζ=1.0) is the main result; ablations vary one hyperparameter at a time. Red shading shows **degradation** level: light (3–10), medium (10–30), dark (>30). Green shading shows **improvement** level: light (3–10), medium (10–30), dark (>30).

	Ensemble size N			Unc. penalty coef. λ					Truncation threshold ζ			
Dataset	100	20	5	0.0	0.04	0.2	1.0	5.0	1.0	0.999	0.99	0.9
halfcheetah-random-v2	37.0	39.7	38.2	37.0	35.6	32.7	19.3	1.2	37.0	35.0	31.2	30.7
hopper-random-v2	24.5	15.1	13.3	24.5	48.2	40.1	31.6	18.8	24.5	8.0	7.1	8.7
walker2d-random-v2	34.1	20.5	8.2	34.1	33.0	34.2	23.3	0.0	34.1	7.1	5.3	6.0
halfcheetah-medium-replay-v2	72.1	68.6	66.7	72.1	72.0	70.1	63.5	52.1	72.1	67.8	68.6	37.7
hopper-medium-replay-v2	110.6	81.8	75.2	110.6	110.9	97.6	95.3	28.8	110.6	79.8	89.7	3.2
walker2d-medium-replay-v2	99.3	91.3	97.8	99.3	87.4	93.5	86.7	72.8	99.3	96.0	92.3	7.4
halfcheetah-medium-v2	78.6	73.7	74.2	78.6	77.5	70.7	59.8	54.6	78.6	74.5	74.0	61.8
hopper-medium-v2	54.2	37.1	48.8	54.2	52.2	105.8	74.0	64.8	54.2	40.3	2.3	2.2
walker2d-medium-v2	106.4	103.1	55.8	106.4	96.6	77.9	93.6	81.9	106.4	88.9	56.5	0.2
halfcheetah-medium-expert-v2	109.4	107.3	97.8	109.4	107.7	112.7	110.3	96.4	109.4	112.3	109.6	109.6
hopper-medium-expert-v2	114.8	100.2	96.8	114.8	114.4	113.5	110.3	110.1	114.8	106.5	2.2	1.8
walker2d-medium-expert-v2	120.6	120.6	118.5	120.6	121.5	116.2	109.3	107.4	120.6	118.5	2.0	1.1
halfcheetah-v3-Low	53.0	51.4	52.2	53.0	52.3	47.3	37.8	26.3	53.0	53.3	52.7	50.1
hopper-v3-Low	30.3	26.6	29.3	30.3	30.0	24.3	15.5	12.7	30.3	22.5	1.6	1.1
walker2d-v3-Low	43.9	41.9	35.4	43.9	46.1	45.7	53.0	53.2	43.9	39.8	22.5	0.2
halfcheetah-v3-Medium	81.1	80.5	77.3	81.1	81.2	79.3	68.1	56.6	81.1	79.5	79.9	70.4
hopper-v3-Medium	95.7	88.6	94.1	95.7	89.3	81.6	72.1	33.4	95.7	86.8	35.2	0.6
walker2d-v3-Medium	50.5	34.4	39.9	50.5	43.9	50.2	44.8	40.9	50.5	55.5	23.1	0.7
halfcheetah-v3-High	68.3	59.9	53.8	68.3	65.9	67.5	71.1	54.1	68.3	56.7	80.4	25.2
hopper-v3-High	96.8	100.4	99.3	96.8	89.6	75.3	90.2	46.4	96.8	85.0	26.1	1.6
walker2d-v3-High	62.7	59.1	57.0	62.7	55.3	67.2	72.2	58.5	62.7	50.0	4.8	0.0
pen-human-v1	20.8	19.3	13.7	20.8	25.8	24.0	35.9	34.8	20.8	27.2	16.8	5.0
pen-cloned-v1	91.3	63.4	75.6	91.3	68.2	76.0	77.2	67.2	91.3	15.7	10.6	0.2
hammer-cloned-v1	14.4	7.4	8.6	14.4	7.0	2.7	1.4	0.2	14.4	10.5	20.5	0.6
antmaze-umaze-v2	66.1	71.1	65.3	66.1	19	9.3	16.7	28.4	66.1	40.6	62.6	86.0
antmaze-umaze-diverse-v2	74.4	52.5	37.8	74.4	0.8	3.3	0.2	19.8	74.4	73.0	63.3	40.4
antmaze-medium-play-v2	12.8	2.2	0.8	12.8	0.0	0.0	0.0	0.0	12.8	18.2	4.5	0.0
antmaze-medium-diverse-v2	19.4	6.8	11.3	19.4	0.0	0.0	0.0	0.0	19.4	11.4	23.1	4.5