REGULATING ACTOR-CRITIC COUPLING WITH UNCERTAINTY IN OFFLINE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

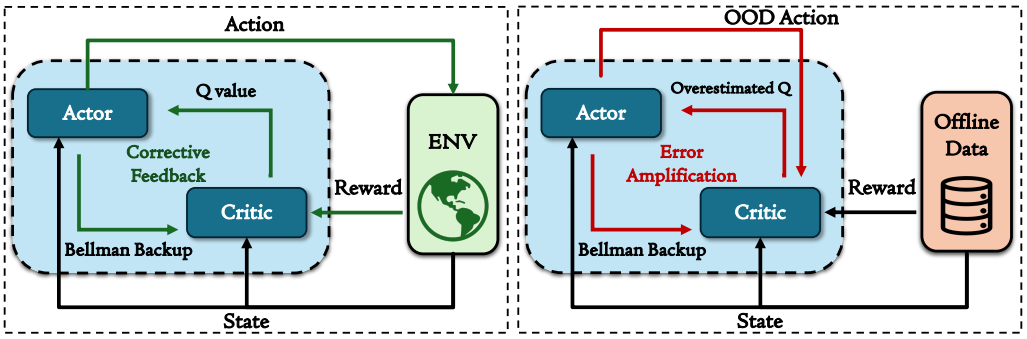
Offline reinforcement learning (RL) is particularly vulnerable to extrapolation errors induced by out-of-distribution (OOD) actions. While existing methods typically address this problem through policy constraints or conservative value estimation, we argue that the key difficulty in offline actor-critic learning lies in how estimation errors are amplified through the coupling between policy improvement and boostrapped value evaluation. In this work, we introduce Uncertainty-Regulated Actor-Critic (URAC), which uses uncertainty as a shared reliability signal to regulate both stages of this coupling. URAC learns a data-conditioned value anchor together with an uncertainty scale. On the critic side, it suppresses uncertain optimistic bootstrap values before they propagate through Bellman updates; on the actor side, it adaptively controls how much value-guided residual deviation from a behavior anchor is accepted. We further show that these two mechanisms provide complementary contractions of actor-side error admission and critic-side error transmission, yielding a multiplicative reduction in closed-loop error propagation. Across Gym-MuJoCo and NeoRL2 benchmarks, URAC achieves the highest aggregate performance among the evaluated methods, while remaining robust when offline data are limited or corrupted by noise. These results suggest that explicitly regulating actor-critic coupling provides an effective way to stabilize policy improvement under uncertain offline supervision.

1 INTRODUCTION

Reinforcement Learning (RL) has made significant progress in a wide range of real-world applications, such as sequential decision-making, robotics control, and game intelligence (Hu et al., 2025; Huang et al., 2024; Liu et al., 2025). The core advantage of RL lies in its ability to learn decision strategies that maximize long-term reward through continuous interaction with the environment (Sutton et al., 1998). However, standard RL typically relies on extensive online interaction, which can be costly and even unsafe in real-world scenarios such as autonomous driving and medical decision-making. Offline RL provides an attractive alternative by learning solely from a pre-collected dataset, which makes it possible to take advantage of existing large-scale datasets without introducing the risks associated with online exploration (Levine et al., 2020; He, 2023). However, this also introduces a fundamental challenge: the learned policy may choose actions that are unsupported in the dataset, whose values are not directly supervised by the offline data. This issue is particularly critical in offline actor-critic methods, where the actor may select such out-of-distribution (OOD) actions during policy improvement, while the critic must estimate their values through extrapolation. As a result, overestimated values can mislead policy improvement and cause severe performance degradation (Riahi Samani et al., 2026; Shi et al., 2023).

The challenge above is not simply a value estimation issue, but becomes more severe through the feedback structure of actor-critic learning. As illustrated in Figure 1, in online actor-critic learning, the policy can interact with the environment and receive corrective feedback for its actions. By contrast, offline actor-critic learns both the actor and the critic from a fixed dataset, so actions proposed during policy improvement cannot be validated by new environment feedback. The actor therefore improves the policy according to critic value estimates, while the critic is updated using Bellman targets induced by current actor. When an unsupported action is overestimated, the actor may select it during policy improvement; once selected, its value enters the Bellman target and the critic is trained toward this biased target in subsequent updates. As a result, extrapolation errors

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



(a) Online actor-critic (b) Offline actor-critic

Figure 1: Error amplification through actor-critic coupling. Online actor-critic can correct value errors through environment interaction, since policy actions are followed by new rewards and transitions. Offline actor-critic lacks this corrective feedback and must rely on a fixed dataset. As a result, an overestimated OOD action may be selected by the actor, enter the Bellman target, and be propagated through critic updates.

can be amplified through policy improvement and Bellman bootstrapping. From this perspective, the central challenge is not only to penalize overestimated critic values, but to regulate the coupling between actor-side policy improvement and critic-side Bellman target updates.

Existing offline RL methods address this problem mainly from one side of the loop. Policy-constrained methods restrict deviation from the behavior distribution (Fujimoto et al., 2019; Kumar et al., 2019; Wu et al., 2019; Jin et al., 2021), whereas conservative or uncertainty-aware methods suppress unreliable value estimates (Kumar et al., 2020; Kostrikov et al., 2021; An et al., 2021). Both strategies are effective, but they generally do not coordinate how residual value errors are admitted by policy improvement and then transmitted through Bellman bootstrapping using the same state-action-dependent reliability signal. This motivates a two-stage view of offline actor-critic error propagation: actor-side admission determines how much unsupported value error enters the target, while critic-side transmission determines how strongly the admitted error propagates through value learning.

Motivated by this coupling analysis, we use uncertainty to regulate actor-critic coupling rather than only penalizing critic values. Specifically, uncertainty controls how much optimistic bootstrap value is transmitted into the Bellman target and how much value-guided policy deviation is accepted. In this way, uncertainty serves as a shared reliability signal that jointly regulates actor-side admission and critic-side transmission.

In light of this, we propose Uncertainty-Regulated Actor-Critic (URAC), a controlled coupling framework for offline actor-critic learning. URAC first learns a conditional value distribution from truncated return samples and extracts two statistics: a value anchor and an uncertainty scale. The value anchor provides a reference for what value level is supported by offline data, while the uncertainty scale measures how strongly this guidance should be trusted. The critic uses these statistics to penalize overestimated bootstrap values before they are propagated through Bellman updates. The actor starts from a behavior anchor and predicts a residual improvement, whose accepted magnitude is controlled by an uncertainty-aware gate. As a result, URAC preserves value-driven improvement in well-supported regions while weakening harmful feedback in uncertain regions.

Our contributions are summarized as follows: (1) We identify a two-stage propagation pathway in offline actor-critic learning, separating actor-side admission of unsupported value errors from critic-side transmission through Bellman bootstrapping. (2) We propose URAC, which uses a shared state-action-dependent reliability signal to jointly regulate residual policy improvement and optimistic bootstrap transmission. (3) We provide theoretical analysis showing the complementary contraction effects of the two modules and demonstrate strong performance and robustness across standard offline RL benchmarks.

2 Related Work

Actor-side admission control. A common strategy in offline RL is to constrain policy improvement toward the support of the offline data. Behavior-regularized methods, including BRAC, BEAR, TD3+BC, AWR, AWAC and ReBRAC, regularize the learned policy toward behavior actions (Wu et al., 2019; Kumar et al., 2019; Fujimoto & Gu, 2021; Peng et al., 2019; Nair et al., 2020; Tarasov et al., 2023). Support-constrained methods, such as BCQ, PLAS and SPOT restrict action selection using learned behavior models, latent action spaces, or support estimates (Fujimoto et al., 2019; Zhou et al., 2021; Wu et al., 2022). These methods mainly reduce actor-side admission of unsupported actions, whereas URAC additionally regulates critic-side bootstrap transmission.

Uncertainty-aware value learning. Another line of work improves offline RL through conservative, pessimistic, or uncertainty-aware value estimation. CQL, Fisher-BRC, MCQ, and Cal-QL penalize overestimated Q-values (Kumar et al., 2020; Kostrikov et al., 2021; Lyu et al., 2022; Nakamoto et al., 2023), while EDAC and UWAC use critic disagreement or uncertainty to improve value robustness (An et al., 2021; Wu et al., 2021; Zhan et al., 2023). PBRL (Bai et al., 2022) estimates uncertainty from an ensemble of Q-functions and uses it to construct pessimistic value targets, primarily regulating the critic side of offline learning. Distributional methods such as QDQ and ANQ estimate value distributions and use their uncertainty for robust offline value learning (Zhang et al., 2024; Mao et al., 2025). RORL instead improves robustness by regularizing the policy and value function over local neighborhoods of perturbed observations (Yang et al., 2022). In contrast, URAC focuses on the two-stage actor-critic error pathway: uncertainty jointly determines how much value-guided policy deviation is admitted by the actor and how much optimistic bootstrap value is transmitted through critic learning.

Model-based and generative offline RL. Model-based offline RL learns dynamics models from offline data and uses them for policy optimization, planning, or synthetic rollout generation. Representative methods include MOPO, MOReL, COMBO, RAMBO and related pessimistic or uncertainty-aware model-based approaches (Yu et al., 2020; Kidambi et al., 2020; Yu et al., 2021; Rigter et al., 2022). Generative behavior-model methods, including SfBC, Diffusion-QL and IDQL, use learned behavior distributions or diffusion models to guide value-based policy improvement (Chen et al., 2022; Wang et al., 2022; Hansen-Estruch et al., 2023). Although these methods can improve data efficiency, they usually introduce additional modeling and sampling stages, and these extra stages can increase additional computational cost and implementation complexity.

3 Method

3.1 Problem Setup and the OOD Propagation Pathway

We consider offline reinforcement learning, where an agent learns only from a fixed dataset $\mathcal{D} = \{(s, a, r, s')\}$ collected by an unknown behavior policy. In actor-critic methods, the critic is trained with the bootstrapped Bellman target $y(x) = r + \gamma Q(s', \pi(s'))$, where $x = (s, a, r, s') \in \mathcal{D}$ and Q denotes the current critic.

The key difficulty in offline RL is that the actor may select out-of-distribution (OOD) actions whose values are weakly constrained by the data and must be estimated through critic extrapolation. Improving beyond the behavior policy may require such actions, but their values can be overestimated because the critic extrapolates beyond the support of the offline data. Once an overestimated OOD action enters the Bellman target, the resulting bias becomes part of the critic's training target and can be amplified through subsequent actor-critic updates.

Prior offline RL methods typically treat this issue as a localized critic extrapolation error to be penalized at the point of evaluation. We argue instead that it propagates through *two coupled stages* (i.e., actor and critic stages) and must be controlled at both stages simultaneously. First, on the actor side, critic overestimation affects the Bellman target by changing which action the policy selects at the next state. When an unsupported action is assigned a higher value than data-supported alternatives, the actor may choose it for policy improvement. The resulting target bias grows with the value gap between the unsupported action and the best data-supported alternative.

Second, since the critic shares parameters across state-action pairs, fitting an overestimated target at one transition changes the critic value at any other query point through coupled gradients. Theorem 4.1 formalizes this two-stage propagation through the decomposition in Eq. equation 9. Here, we use this decomposition to motivate the design of URAC.

This view suggests that offline overestimation cannot be fully addressed by penalizing critic magnitudes alone. Both stages of the propagation pathway must be controlled: the actor side, by reducing how much OOD value is admitted into the target; and the critic side, by reducing how much an admitted target perturbation is propagated through shared parameters.

3.2 Uncertainty Signal

Both stages of the OOD propagation pathway depend on whether value guidance at a queried (s, a) is supported by the offline data. We therefore use uncertainty as a state-action-dependent control signal that determines how much value guidance should be trusted.

We use a two-factor uncertainty signal: a **value anchor** $\mu(s, a)$ that provides a data-conditioned reference value, and an **uncertainty scale** $u(s, a)$ that measures the spread of plausible values around it. The variance $u(s, a)$ alone is insufficient because it cannot distinguish a noisy low-value region from a stable but unsupported optimistic estimate; pairing it with the anchor lets us identify whether the critic is optimistic relative to data and how strongly that optimism should be discounted. The downstream critic and actor modules use μ to identify the above-anchor component of value estimates and u to determine how strongly that component should be suppressed.

We instantiate $(\mu(s, a), u(s, a))$ using the consistency-model-based value estimator of Zhang et al. (2024). We construct proxy value samples $\hat{Q}(s_t, a_t) = \sum_{k=0}^{K-1} \gamma^k r_{t+k}$ from truncated discounted returns and train a consistency model conditioned on (s, a) to approximate $p_{\mathrm{CM}}(q \mid s, a)$. Drawing N samples $q_{\mathrm{CM}}^{(i)}(s, a) \sim p_{\mathrm{CM}}(q \mid s, a)$, we set

$$\mu(s, a) = \frac{1}{N} \sum_{i=1}^{N} q_{\mathrm{CM}}^{(i)}(s, a), \qquad u(s, a) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \big(q_{\mathrm{CM}}^{(i)}(s, a) - \mu(s, a)\big)^2 + \varepsilon}, \tag{1}$$

where $\varepsilon > 0$ is for numerical stability. The estimator is trained once before actor-critic learning and then frozen, so (μ, u) is fixed throughout URAC training. Implementation details are in Appendix A.

3.3 Uncertainty-Aware Critic

For $(s, a, r, s') \sim \mathcal{D}$ and $a' = \pi(s')$, the standard Bellman target directly transmits any extrapolation error in the target critic $\bar{Q}(s', a')$ into critic training. To address this issue, we use the uncertainty signal to regulate this transmission.

Define the positive deviation above the anchor and an uncertainty-dependent shrinkage weight:

$$d(s, a) = \big[\bar{Q}(s, a) - \mu(s, a)\big]_+, \qquad w(s, a) = \frac{u(s, a)}{u(s, a) + c}, \tag{2}$$

where $[x]_+ = \max(x, 0)$ and $c > 0$ calibrates how quickly shrinkage grows with uncertainty. The penalized bootstrap value is

$$\tilde{Q}(s, a) = \bar{Q}(s, a) - w(s, a) d(s, a), \tag{3}$$

Substituting Eq. equation 3 into the Bellman backup gives the penalized Bellman target:

$$y = r + \gamma\big(\bar{Q}(s', a') - w(s', a')[\bar{Q}(s', a') - \mu(s', a')]_+\big). \tag{4}$$

The anchor identifies the above-reference component of the critic estimate, while the uncertainty scale determines how strongly that component is suppressed: when the critic is at or below the anchor, the target is unchanged; when the critic exceeds the anchor in a high-uncertainty region, a large fraction of the excess is removed. The critic is trained with

$$L_Q = \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}} \left[\big(Q_\theta(s, a) - (r + \gamma \tilde{Q}(s', a'))\big)^2 \right]. \tag{5}$$

Theorem 4.2 shows that this penalization contracts the critic-side propagation term in Eq. equation 9, with the uncertainty-dependent contraction specified in Eq. equation 10. Thus, the critic module not only lowers target magnitudes but also contracts critic-side bias propagation.

3.4 Gated Residual Actor

Even with the uncertainty-aware critic, a greedy actor can still exploit residual overestimation, while a uniformly small residual region would be overly conservative in low-uncertainty states. We separate policy improvement into *proposal* and *acceptance*: the actor proposes a residual direction, and a gate decides how much of it is admitted.

Let $a_b(s) = \pi_b(s)$ be a behavior anchor from a behavior cloning policy. The actor predicts a residual $\Delta a_\phi(s)$, defining the proposal $a_p(s) = a_b(s) + \Delta a_\phi(s)$. We evaluate the proposal with $\mu_p(s) = \mu(s, a_p(s))$ and $u_p(s) = u(s, a_p(s))$, and predict a soft acceptance coefficient

$$g_\psi(s) = \mathrm{sigmoid}\big(\mathrm{MLP}_\psi[h_\theta(s, a_p(s)),\, \mu_p(s),\, u_p(s)]\big), \tag{6}$$

where $h_\theta(s, a_p(s)) \in \mathbb{R}^d$ is the penultimate-layer representation of the critic at the proposal action, so that $Q_\theta(s, a) = w^\top h_\theta(s, a) + b$. Accordingly, the gate takes $h_\theta(s, a_p(s))$, $\mu_p(s)$, and $u_p(s)$ as input so that acceptance reflects both how promising the proposal appears to the critic and how well that signal is supported by the offline data. The final actor action is

$$a_\pi(s) = a_b(s) + g_\psi(s)\Delta a_\phi(s), \tag{7}$$

clipped to the valid action range. Thus Δa_ϕ determines the improvement direction and g_ψ determines how much of it is accepted, with the gate input combining local critic geometry, a data-conditioned value reference, and a local uncertainty measure.

The actor is trained by maximizing

$$J_\pi(\phi, \psi) = \mathbb{E}_{s\sim\mathcal{D}}\left[Q_\theta(s, a_\pi(s)) - u_p(s)\|a_\pi(s) - a_b(s)\|_2^2\right]. \tag{8}$$

Under Eq. equation 7, the $u_p(s)\|a_\pi(s) - a_b(s)\|_2^2$ term acts on the *accepted* residual rather than the raw proposal: the gate selects whether to admit the proposal, and this term shrinks the admitted deviation as $u_p(s)$ grows. Theorem 4.3 shows that the target deviation is bounded as in Eq. equation 11, so the gate contracts the actor-side admission term. Together with Theorem 4.2, this yields the multiplicative closed-loop bound in Theorem 4.4 and Eq. equation 13.

3.5 Overall Algorithm

Algorithm 1 summarizes URAC. Phase 1 trains the consistency-model value estimator on the offline dataset $\mathcal{D}$ using truncated returns, producing the frozen (μ, u) used throughout phase 2. Phase 2 jointly trains the critic, residual actor, and behavior cloning policy. Full hyperparameter schedules and target-network details are in Appendix B.

4 Theoretical Analysis

We formalize the coupled OOD propagation view in Section 3.1 through a local perturbation analysis. The goal is not to prove global convergence of nonlinear offline actor-critic learning, but to characterize how an OOD target error is admitted by the actor and transmitted by the critic within one update. During critic updates, the target critic, actor, and uncertainty statistics are treated as fixed; during actor updates, the uncertainty outputs are detached. We assume local reliability of the frozen uncertainty signal around the behavior-anchored proposal region. Full assumptions and proofs are deferred to Appendix C.

For a transition $x = (s, a, r, s')$ and a query point $z = (\bar{s}, \bar{a})$, define the OOD preference gap

$$\delta(s') = \bar{Q}(s', a_{\mathrm{out}}(s')) - \bar{Q}(s', a_{\mathrm{in}}(s')),$$

where a_{out} is the highest-valued unsupported action and a_{in} is the best data-supported action.

Theorem 4.1 (Baseline propagation) *Under the local assumptions in Appendix C, the one-step OOD-induced value change of standard offline actor-critic satisfies*

$$\Delta Q^{\mathrm{std}}(z) = \gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[K(z, x)[\delta(s')]_+\right] + O(\eta^2), \tag{9}$$

where $K(z, x) = \nabla_\theta Q_\theta(z)^\top \nabla_\theta Q_\theta(s, a)$. Here $[\delta(s')]_+$ is the actor-side admitted OOD increment, while $K(z, x)$ is the critic-side propagation factor.

Algorithm 1 Uncertainty-Regulated Actor-Critic (URAC)

Initialize: offline dataset $\mathcal{D}$, discount γ, consistency model $p_{\mathrm{CM}}(q \mid s, a)$, critic Q_θ, target critic $\bar{Q}$, residual proposal Δa_ϕ, gate g_ψ, and behavior policy π_b.
Stage 1: Uncertainty estimator
Construct the value-sample dataset D_Q from truncated returns on $\mathcal{D}$
Train $p_{\mathrm{CM}}(q \mid s, a)$ on D_Q and compute $(\mu(s, a), u(s, a))$ as in Eq. equation 1
Stage 2: Behavior anchor
Train π_b on $\mathcal{D}$ by behavior cloning
Stage 3: Actor–critic training
for each gradient step **do**
  Sample a minibatch of transitions $(s, a, r, s') \sim \mathcal{D}$
  Evaluate the anchor actions and compute the gated actions at s and s' by Eqs. equation 6 and equation 7
  Form the uncertainty-corrected bootstrap value and target by Eqs. equation 3 and equation 4
  Update critic Q_θ with Eq. equation 5
  Update $(\Delta a_\phi, g_\psi)$ with Eq. equation 8
  Update the target critic $\bar{Q}$
end for
return $Q_\theta,\ \Delta a_\phi,\ g_\psi,\ \pi_b$

Theorem 4.2 (Critic-side contraction) *For the corrected bootstrap value*

$$\widetilde{Q}(s, a) = \bar{Q}(s, a) - w(s, a)[\bar{Q}(s, a) - \mu(s, a)]_+, \qquad w(s, a) = \frac{u(s, a)}{u(s, a) + c},$$

the local sensitivity to the raw target value is

$$\frac{\partial \widetilde{Q}(s, a)}{\partial \bar{Q}(s, a)} = 1 - \rho(s, a), \qquad \rho(s, a) = w(s, a)\mathbf{1}\{\bar{Q}(s, a) > \mu(s, a)\}. \tag{10}$$

Thus, URAC replaces the critic-side propagation factor in Eq. equation 9 by $K(z, x)(1 - \rho(s', a'))$, up to an $O(\eta\epsilon_\mu)$ term when the local anchor has bounded error ϵ_μ.

For the gated residual actor, define the accepted residual magnitude as

$$\alpha_\psi(s) = g_\psi(s)\|\Delta a_\phi(s)\|_2.$$

Theorem 4.3 (Actor-side admission contraction) *If $\bar{Q}(s, \cdot)$ is locally M-Lipschitz in the action, then the behavior-anchored target y_b and the URAC target y_π satisfy*

$$|y_\pi - y_b| \le \gamma M\, \alpha_\psi(s') = \gamma M g_\psi(s')\|\Delta a_\phi(s')\|_2. \tag{11}$$

Moreover, under a local quadratic approximation of the actor objective with detached $u_p(s)$,

$$\alpha^\star_\psi(s) = O\left(\frac{1}{u_p(s)}\right). \tag{12}$$

Let $e_{\mathrm{std}}(s') = [\delta(s')]_+$ be the OOD increment admitted by a standard actor update, and let $e_{\mathrm{URAC}}(s') \le \beta_A e_{\mathrm{std}}(s')$ denote the corresponding increment under URAC, with $\beta_A \in [0, 1]$.

Theorem 4.4 (Closed-loop contraction) *Under the assumptions of Theorems 4.2 and 4.3, the URAC one-step OOD-induced value change satisfies*

$$|\Delta Q^{\mathrm{URAC}}(z)| \le (1 - \rho_{\min})\beta_A|\Delta Q^{\mathrm{std}}(z)| + O(\eta^2) + O(\eta\epsilon_\mu), \tag{13}$$

where $\rho_{\min}$ is the essential infimum of ρ. When the anchor is locally exact, the $O(\eta\epsilon_\mu)$ term vanishes and the bound becomes a pure multiplicative contraction.

Theorem 4.4 shows that URAC reduces the effective loop gain by jointly controlling actor-side admission and critic-side bootstrap transmission. Removing either side leaves only a single contraction factor: a critic-only variant has $\beta_A = 1$, while an actor-only variant has $\rho \equiv 0$. The multi-step bias-control extension is provided in Appendix C.5.

5 EXPERIMENTS

In this section, we conduct experiments to evaluate the performance, robustness, and coupling-control behavior of URAC. Additional experimental details, computational cost analysis and hyper-parameter statement are provided in Appendix D.

5.1 BENCHMARKS COMPARISON

We evaluate URAC on Gym-MuJoCo (Fu et al., 2020) and NeoRL2 (Gao et al., 2025) benchmarks. Gym-MujoCo is a standard offline continuous-control benchmark focusing on policy-quality mixtures and distribution-shifted value learning. NeoRL2 is a near-real-world offline RL benchmark covering realistic data and safety challenges. We compare URAC with BC (Torabi et al., 2018), CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2021), EDAC (An et al., 2021), TD3BC (Fujimoto & Gu, 2021), COMBO (Yu et al., 2021), QDQ (Zhang et al., 2024), ANQ (Mao et al., 2025), PBRL (Bai et al., 2022) and RORL (Yang et al., 2022). The experimental results for the baselines reported in this paper are derived from the original experiments conducted by the authors or from replication of their official code.

Table 1: Comparison of URAC and the other baselines on Gym-MuJoCo and NeoRL benchmarks. Results are averaged normalized scores over 5 random seeds. **Bold** indicates the best result and underline indicates the second-best result.

Task	BC	CQL	IQL	EDAC	TD3BC	COMBO	QDQ	ANQ	PBRL	RORL	URAC(Ours)
Gym-MuJoCo-v2											
Hopper-med	52.9	58.5	66.2	101.6	59.3	97.2	99.3	100.9	75.3	104.8	**106.4±2.1**
Halfcheetah-med	42.6	44.0	58.4	65.9	48.3	54.2	74.1	61.8	57.9	66.8	**76.3±0.9**
Walker2D-med	75.3	72.5	78.3	**92.5**	83.7	81.9	86.9	82.9	86.8	92.4	**92.5±0.3**
Hopper-med-replay	18.1	95.0	94.7	101.0	60.9	89.5	102.4	101.5	100.6	102.8	**105.4±0.2**
Halfcheetah-med-replay	36.6	45.5	44.2	61.3	44.6	55.1	63.7	55.5	45.1	61.9	**65.7±0.7**
Walker2D-med-replay	26.0	77.2	73.8	87.1	81.8	56.0	93.2	92.7	77.7	90.4	**94.6±1.3**
Hopper-med-expert	52.5	105.4	91.5	110.7	98.0	111.1	**113.5**	107.0	110.8	112.7	113.2±0.7
Halfcheetah-med-expert	55.2	91.6	86.7	**106.3**	90.7	90.0	99.3	94.2	105.9	**106.3**	99.4±2.2
Walker2D-med-expert	107.5	108.8	109.6	114.7	110.1	103.3	115.9	111.7	114.9	**120.2**	115.9±1.4
Total	466.7	698.5	703.4	841.1	677.4	738.3	848.3	808.2	775.0	858.3	**866.0**
NeoRL2											
Fusion	54.9	55.9	56.4	58.0	54.6	55.5	58.6	57.3	58.6	57.9	**59.7±3.1**
Pipeline	59.4	76.8	75.3	72.9	**79.6**	55.5	75.3	77.2	75.6	76.1	77.8±0.2
RocketRecovery	72.8	74.3	76.0	65.7	**79.7**	74.7	73.6	74.8	74.9	75.4	76.5±0.6
Simglucose	73.9	11.0	75.1	8.1	74.2	23.2	76.2	75.2	75.9	77.1	**77.4±2.5**
RandomFrictionHopper	28.7	28.0	30.8	34.7	29.5	34.1	40.8	40.1	45.6	41.5	**46.9±1.3**
DMSD	56.6	70.2	77.4	78.7	60.0	68.3	76.4	76.9	82.1	80.4	**82.3±0.2**
SafetyHalfCheetah	73.6	71.2	74.9	53.1	68.6	57.8	74.2	70.5	74.0	73.5	**77.1±1.1**
Total	419.9	387.4	465.9	371.2	446.2	369.1	475.1	472.0	486.7	481.9	**497.7**

Aggregated results are reported in Table 1. URAC achieves the highest total score on both Gym-MuJoCo and NeoRL2, showing that uncertainty-regulated actor-critic coupling is effective across standard locomotion benchmarks and more realistic offline RL scenarios. On Gym-MuJoCo, the improvement is most pronounced on medium and medium-replay datasets, where the data contain useful policy-improvement signals but still have imperfect action support. In this regime, standard actor-critic methods are vulnerable to value extrapolation and bootstrapped error amplification, while URAC suppresses uncertain bootstrap values on the critic side and adaptively limits residual policy deviation on the actor side. The NeoRL2 results further suggest that the proposed coupling-control mechanism can improve robustness beyond D4RL-style locomotion tasks.

5.2 ABLATION AND SENSITIVITY STUDY

We first ablate the uncertainty design of URAC. As shown in Table 2, removing the value anchor consistently degrades performance, indicating that uncertainty scale alone is not sufficient for reliable coupling control. The anchor is important because it provides a data-supported reference for identifying optimistic value estimates. In addition, applying uncertainty only to the critic or the actor performs worse than the full model. This suggests that the two modules regulate complementary stages of the offline actor-critic loop: the critic reduces the transmission of unreliable bootstrap

Table 2: Ablation study of URAC's uncertainty design on D4RL Gym-MuJoCo tasks. ha = halfcheetah, wa = walker2d, ho=hopper, med = medium, e = expert

Method	ha-med	ho-med	wa-med	ha-med-e	ho-med-e	wa-med-e
URAC (full)	**76.3**	**106.4**	**92.5**	**99.4**	**113.2**	**115.5**
w/o value anchor	71.5	91.2	89.4	96.8	95.7	113.0
Uncertainty in Critic only	58.9	79.3	80.9	90.6	92.4	104.4
Uncertainty in Actor only	53.6	72.7	73.1	88.1	89.8	95.0

Table 3: Sensitivity analysis of the shrinkage constant c.

c	0.05	0.1	0.5	1.0	2.0
Hopper-med	104.3	105.2	106.0	**106.4**	103.7
HalfCheetah-med	73.1	75.3	75.9	**76.3**	76.0

values, while the actor controls how much value-guided residual deviation is admitted. Their combination gives the best overall performance.

We also study the sensitivity to c in Eq. (2), which controls the strength of critic-side shrinkage. As shown in Table 3, URAC performs consistently across a broad range of c values. Smaller c leads to stronger suppression of above-anchor bootstrap values, whereas larger c retains more of the original critic estimate. The relatively small performance variation indicates that URAC is not highly sensitive to the exact choice of c.

We further analyze the effect of actor-critic coupling strength by replacing the learned gate with a fixed coefficient g. As shown in Figure 2, performance is generally non-monotonic with respect to g. A small g keeps the policy too close to the behavior anchor and limits useful improvement, whereas a large g makes the actor overly dependent on critic estimates and increases the risk of amplifying extrapolation errors. The best fixed g also varies across tasks, showing that a manually chosen global coupling strength is insufficient. This supports the use of URAC's uncertainty-aware gate, which adaptively adjusts the accepted residual magnitude according to local reliability.

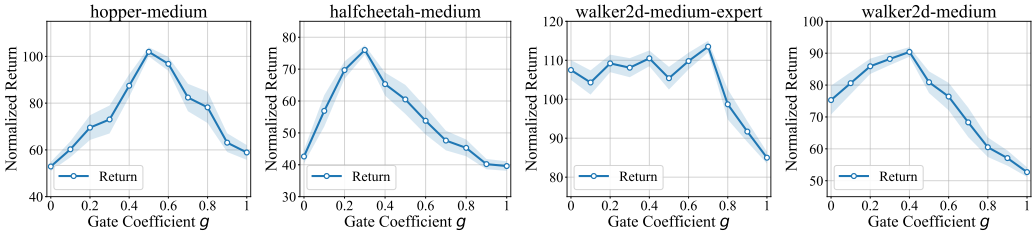
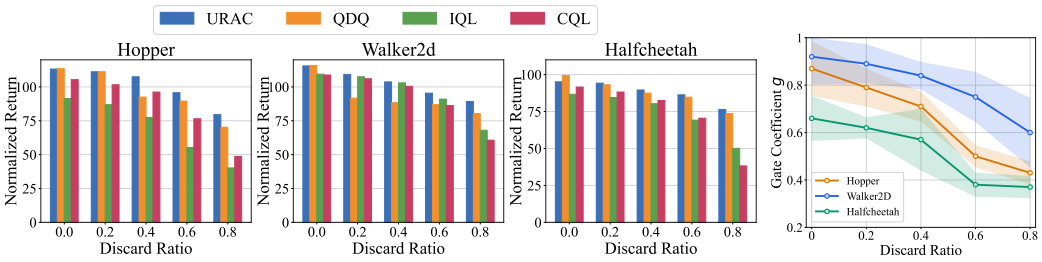


Figure 2: Effect of fixed gate coefficient g. As g increases, policy selection tends to prefer exploratory policies; as g decreases, policy selection tends to prefer conservative policies.

5.3 Limited and Noisy Data Results

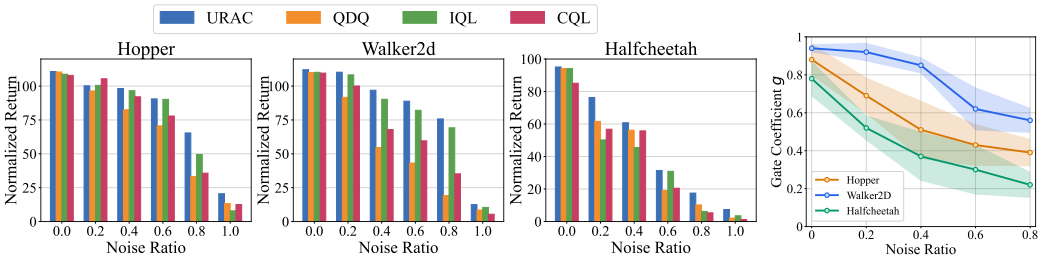
We further evaluate the robustness of URAC under two degraded offline-data settings: limited data and noisy data. For the limited-data setting, we randomly discard a fraction of transitions from the original medium-expert dataset, with the discard ratio controlling the remaining data coverage. For the noisy-data setting, we mix the original medium-expert dataset with noisy or low-quality samples, with the noise ratio controlling their proportion in the resulting training set. Full construction details are provided in Appendix D.4.

Figure 3a shows that URAC outperforms other algorithms across all discard ratios. As data coverage becomes sparse, value extrapolation becomes less reliable and standard actor-critic methods are more likely to amplify overestimated actions through Bellman bootstrapping. URAC mitigates this issue by suppressing uncertain bootstrap values on the critic side and limiting residual policy



(a) Performance under reduced datasets (b) Average gate coefficient

Figure 3: Robustness under limited-data settings. (a) Normalized return under different discard ratios on medium-expert datasets. (b) Average gate coefficient learned by URAC as the discard ratio increases. URAC maintains stronger performance while reducing the accepted residual deviation when data coverage becomes weaker.



(a) Performance under noisy datasets (b) Average gate coefficient

Figure 4: Robustness under noisy-data settings. (a) Normalized return under different noise ratios on medium-expert datasets. (b) Average gate coefficient learned by URAC as the noise ratio increases.

deviation on the actor side. Figure 3b shows that the average gate coefficient decreases as the discard ratio increases, indicating that URAC weakens actor-critic coupling when data support is reduced.

Figure 4a also demonstrates that URAC achieves superior performance across all noise ratios. Noisy data make critic guidance less trustworthy, which increases the risk that misleading value estimates are admitted by the actor and reinforced through future backups. As shown in Figure 4b, the learned gate also decreases with a stronger noise. These results jointly suggest that URAC improves robustness by adaptively reducing the effective coupling strength of offline reinforcement learning actor-critic under degraded data conditions.

6 Conclusion

This paper focuses on offline reinforcement learning from the perspective of closed-loop error amplification. Instead of treating OOD overestimation as an isolated critic-side extrapolation problem, we show that estimation errors can be admitted by policy improvement and further propagated through Bellman bootstrapping. To address this issue, we propose Uncertainty-Regulated Actor-Critic (URAC). URAC reduces the influence of overestimated bootstrap values before they are propagated through Bellman updates. On the actor side, URAC uses an uncertainty-aware gate to control the accepted residual deviation from behavior anchor. Theoretical analysis shows that the two modules weaken the error loop from both sides: the actor admits fewer unreliable deviations, and the critic transmits less biased bootstrap information. Experiments on standard offline RL benchmarks show that URAC improves offline actor-critic learning across both standard locomotion tasks and more realistic offline RL scenarios. Ablation studies further verify the necessity of the two-sided design. Additional evaluations with limited and noisy data demonstrate that this joint regulation remains effective when offline data support is sparse or value supervision is corrupted. In the future, we will study the adaptation of URAC to multi-task learning scenarios. We will also explore how to enhance the robustness of URAC under shifting environments.

AI USE STATEMENT

Generative AI tools were used to assist with language polishing and to improve the clarity and readability of the manuscript. All technical content and scientific claims were reviewed and verified by the authors.

REFERENCES

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.

Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhihong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. *arXiv preprint arXiv:2202.11566*, 2022.

Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. *arXiv preprint arXiv:2209.14548*, 2022.

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.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

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

Songyi Gao, Zuolin Tu, Rong-Jun Qin, Yi-Hao Sun, Xiong-Hui Chen, and Yang Yu. Neorl-2: Near real-world benchmarks for offline reinforcement learning with extended realistic scenarios. *arXiv preprint arXiv:2503.19267*, 2025.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

Haoyang He. A survey on offline model-based reinforcement learning. *arXiv preprint arXiv:2305.03360*, 2023.

Edward S Hu, Jie Wang, Xingfang Yuan, Fiona Luo, Muyao Li, Gaspard Lambrechts, Oleh Rybkin, and Dinesh Jayaraman. Real-world reinforcement learning of active perception behaviors. *arXiv preprint arXiv:2512.01188*, 2025.

Sili Huang, Jifeng Hu, Zhejian Yang, Liwei Yang, Tao Luo, Hechang Chen, Lichao Sun, and Bo Yang. Decision mamba: Reinforcement learning via hybrid selective sequence modeling. *Advances in Neural Information Processing Systems*, 37:72688–72709, 2024.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International conference on machine learning*, pp. 5084–5096. PMLR, 2021.

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.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

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.

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.

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.

Anjie Liu, Jianhong Wang, Samuel Kaski, Jun Wang, and Mengyue Yang. A principle of targeted intervention for multi-agent reinforcement learning. *arXiv preprint arXiv:2510.17697*, 2025.

Jiafei Lyu, Xiaoteng Ma, Xiu Li, and Zongqing Lu. Mildly conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1711–1724, 2022.

Yixiu Mao, Yun Qu, Qi Wang, and Xiangyang Ji. Adaptive neighborhood-constrained q learning for offline reinforcement learning. *arXiv preprint arXiv:2511.02567*, 2025.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36:62244–62269, 2023.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Ali Riahi Samani, Xujiang Zhao, and Feng Chen. Distribution shift, generalization and ood challenge in offline reinforcement learning: a comprehensive survey. *Neural Computing and Applications*, 38(6):150, 2026.

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.

Laixi Shi, Robert Dadashi, Yuejie Chi, Pablo Samuel Castro, and Matthieu Geist. Offline reinforcement learning with on-policy q-function regularization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 455–471. Springer, 2023.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

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, 2023.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

Jialong Wu, Haixu Wu, Zihan Qiu, Jianmin Wang, and Mingsheng Long. Supported policy optimization for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:31278–31291, 2022.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Yue Wu, Shuangfei Zhai, Nitish Srivastava, Joshua Susskind, Jian Zhang, Ruslan Salakhutdinov, and Hanlin Goh. Uncertainty weighted actor-critic for offline reinforcement learning. *arXiv preprint arXiv:2105.08140*, 2021.

Rui Yang, Chenjia Bai, Xiaoteng Ma, Zhaoran Wang, Chongjie Zhang, and Lei Han. Rorl: Robust offline reinforcement learning via conservative smoothing. *Advances in neural information processing systems*, 35:23851–23866, 2022.

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.

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.

Wenhao Zhan, Masatoshi Uehara, Wen Sun, and Jason D Lee. Provable reward-agnostic preference-based reinforcement learning. *arXiv preprint arXiv:2305.18505*, 2023.

Jing Zhang, Linjiajie Fang, Kexin Shi, Wenjia Wang, and Bing-Yi 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, 2024.

Wenxuan Zhou, Sujay Bajracharya, and David Held. Plas: Latent action space for offline reinforcement learning. In *Conference on Robot Learning*, pp. 1719–1735. PMLR, 2021.

A Additional Details of the Uncertainty Signal Estimator

This appendix provides the implementation details omitted from Section 3.2. The uncertainty estimator is an auxiliary module used only to produce fixed reliability statistics $\mu(s, a)$ and $u(s, a)$; it is not jointly optimized with the actor and critic during URAC training.

A.1 Proxy Value Dataset

The offline dataset is organized as trajectories

$$\tau = \big((s_0, a_0, r_0), \ldots, (s_T, a_T, r_T)\big).$$

For each time step t in a trajectory, we construct a truncated discounted return

$$\hat{Q}(s_t, a_t) = \sum_{k=0}^{\min(K-1,\, T-t)} \gamma^k r_{t+k}.$$

If fewer than K rewards remain before the end of the trajectory, the summation is truncated at termination. We then form the proxy value dataset

$$D_Q = \{((s_t, a_t), \hat{Q}(s_t, a_t))\}.$$

These proxy targets are used only as noisy value samples for distribution learning. They are not treated as exact ground-truth long-horizon Q-values.

A.2 Consistency-Model Training

We adopt the consistency-model-based conditional value estimator of Zhang et al. (2024). A model conditioned on (s, a) is trained on D_Q to represent a data-conditioned value distribution

$$p_{\mathrm{CM}}(q \mid s, a).$$

We follow the training recipe of the cited estimator for the noise schedule and consistency/distillation objective.

The use of a consistency model is important because the truncated returns in D_Q are noisy proxy value samples rather than exact long-horizon Q-labels. Instead of fitting a single deterministic target, the consistency model learns a conditional distribution over plausible values around each queried state-action pair. Sampling from this distribution allows URAC to extract two complementary statistics: the first moment provides a data-conditioned value anchor $\mu(s, a)$, while the sample spread provides an uncertainty scale $u(s, a)$. This distributional view is especially useful in offline RL, where value estimates outside well-supported regions are not directly supervised and point estimates can be overconfident.

The effectiveness of consistency-model-based value distribution estimation has been empirically validated in QDQ (Zhang et al., 2024), where the sampled value distribution is used to obtain practical uncertainty statistics for offline value learning. URAC builds on this estimator rather than introducing a new value-distribution learner. Its role in URAC is to provide a fixed, data-conditioned reliability signal that can be shared by the critic-side bootstrap correction and the actor-side residual gate.

We emphasize, however, that $\mu(s, a)$ and $u(s, a)$ are not assumed to be globally calibrated uncertainty estimates over arbitrary far-OOD actions. They are used as local reliability statistics around the behavior-anchored proposal region queried by the residual actor. Accordingly, the estimator is trained once before actor-critic learning and then frozen, so that the uncertainty signal acts as an external reliability statistic rather than as another participant in the actor–critic feedback loop.

A.3 Moment Extraction and Detached Use in URAC

For a queried state-action pair (s, a), we draw N samples

$$q_{\mathrm{CM}}^{(1)}(s, a), \ldots, q_{\mathrm{CM}}^{(N)}(s, a) \sim p_{\mathrm{CM}}(q \mid s, a),$$

and compute the value anchor and uncertainty scale using Eq. equation 1. The sample mean $\mu(s, a)$ serves as a data-conditioned reference value, while $u(s, a)$ measures the spread of plausible values around this reference.

The estimator is trained once before actor-critic learning and then frozen. During URAC training, all outputs of the estimator are treated as detached reliability statistics. That is, no gradient is back-propagated to the consistency model parameters, and no gradient is propagated through the dependence of $\mu(s, a)$ or $u(s, a)$ on the queried action. Thus, the uncertainty signal acts as a fixed external reliability statistic rather than as another participant in the actor–critic feedback loop.

For the critic update, μ and u are evaluated at the next-state actor action $a' = \pi(s')$ inside the Bellman target, but the resulting values are detached:

$$\mu'(s') = \mathrm{sg}[\mu(s', \mathrm{sg}(a'))], \qquad u'(s') = \mathrm{sg}[u(s', \mathrm{sg}(a'))],$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient. These detached statistics are then used to construct the uncertainty-corrected bootstrap value.

For the actor update, the estimator is queried at the proposal action

$$a_p(s) = a_b(s) + \Delta a_\phi(s),$$

before the residual proposal is gated. We compute

$$\mu_p(s) = \mathrm{sg}[\mu(s, \mathrm{sg}(a_p(s)))], \qquad u_p(s) = \mathrm{sg}[u(s, \mathrm{sg}(a_p(s)))].$$

Therefore, $\mu_p(s)$ and $u_p(s)$ affect the gate input and the residual penalty as fixed reliability features, but actor gradients are not back-propagated through the uncertainty estimator or through the query action $a_p(s)$. The actor still receives gradients through the critic value term $Q_\theta(s, a_\pi(s))$, the residual action $a_\pi(s) = a_b(s) + g_\psi(s)\Delta a_\phi(s)$, and the gate parameters.

Although the estimator is frozen and detached within each update, its queried values can change across training iterations because the proposal action changes as the actor is updated. The anchor μ identifies whether the queried value lies above a data-supported reference, while the uncertainty scale u determines how strongly this above-anchor deviation should be penalized or suppressed.

B Algorithmic Details

The main-text algorithm intentionally keeps only the high-level URAC pipeline. The implementation follows the same three-stage structure.

First, we train the uncertainty estimator on D_Q and use it to evaluate the reliability statistics (μ, u). Second, we train a behavior cloning policy π_b on the offline dataset and use $a_b(s) = \pi_b(s)$ as the behavior anchor. Third, we optimize the critic and the gated residual actor around this anchor. Each actor–critic update uses the following steps: sample a minibatch of transitions, evaluate the anchor actions and the gated actions at the current and next states, build the uncertainty-corrected Bellman target, update the critic with Eq. equation 5, update the residual proposal and gate with Eq. equation 8, and refresh the target critic.

C Proofs for Theoretical Analysis

This appendix provides the formal derivations for the theoretical claims in Section 4. The analysis is local and one-step unless otherwise specified. Its purpose is not to prove global convergence of nonlinear offline actor-critic learning, but to make explicit how OOD-induced target errors are admitted by the actor and then propagated by the critic through shared parameters. We also state the additional assumptions under which the local contraction can be unrolled into a multi-step bias-control bound.

Notation. Let $x = (s, a, r, s') \sim \mathcal{D}$ denote a sampled transition and let $z = (\bar{s}, \bar{a})$ denote an arbitrary query state-action pair. The critic being updated is Q_θ, while the target critic used inside the Bellman backup is denoted by $\bar{Q}$. The empirical critic NTK between the query point z and the training input (s, a) is

$$K_\theta(z, x) = \nabla_\theta Q_\theta(z)^\top \nabla_\theta Q_\theta(s, a). \tag{14}$$

For readability, we write $K(z, x)$ when the dependence on θ is clear.

Assumptions. We use the following local assumptions throughout the appendix.

1. **Fixed-target update.** During one critic update, the target critic $\bar{Q}$, the actor π, and the uncertainty statistics (μ, u) are treated as fixed. This follows the standard target-network convention in bootstrapped actor-critic updates.
2. **Detached uncertainty statistics.** The consistency-model uncertainty estimator is trained before actor-critic learning and then frozen. During actor updates, its outputs are detached:
$$\mu_p(s) = \mathrm{sg}[\mu(s, \mathrm{sg}(a_p(s)))], \qquad u_p(s) = \mathrm{sg}[u(s, \mathrm{sg}(a_p(s)))],$$
where $a_p(s) = a_b(s) + \Delta a_\phi(s)$ and $\mathrm{sg}(\cdot)$ denotes stop-gradient. Thus, gradients are not propagated through the uncertainty estimator or through its dependence on the proposal action. In the actor-side analysis, $u_p(s)$ is therefore treated as a fixed coefficient.
3. **Local differentiability.** $Q_\theta(z)$ and $Q_\theta(s, a)$ are differentiable in θ, and the Taylor remainder after one gradient step of size η is $O(\eta^2)$.
4. **Behavior-support partition.** For analysis only, the action space at each state s is partitioned into a data-supported region $\mathcal{A}_{\mathrm{in}}(s)$ and an unsupported region $\mathcal{A}_{\mathrm{out}}(s)$. The algorithm does not require estimating this partition.
5. **Local reliability of the uncertainty signal.** We do not assume that the frozen estimator is globally calibrated for arbitrary far-OOD actions. The analysis only requires local reliability on the behavior-anchored proposal region
$$\mathcal{A}_q(s) = \{a_b(s) + \lambda \Delta a_\phi(s) : \lambda \in [0, 1]\}.$$
Let $Q_{\mathrm{ref}}(s, a)$ be a local data-supported reference value. We assume that, for actions queried in this local region,
$$|\mu(s, a) - Q_{\mathrm{ref}}(s, a)| \le \epsilon_\mu. \tag{15}$$
This assumption allows the anchor to be imperfect. The resulting error appears as an additive $O(\eta\epsilon_\mu)$ term in the closed-loop bound.
6. **Local action regularity.** For the actor-side result, $\bar{Q}(s, \cdot)$ is locally M-Lipschitz in action around the behavior anchor and the accepted residual action. For the local accepted-magnitude optimizer, $Q_\theta(s, \cdot)$ is locally smooth along the residual direction.
7. **Magnitude convention for signed kernels.** The empirical NTK $K(z, x)$ can be signed. For clarity, the main text states the no-cancellation form. When signed kernels are considered, we use the magnitude proxy
$$\mathcal{B}^{\mathrm{std}}(z) = \gamma\eta \mathbb{E}_{x\sim\mathcal{D}}\left[|K(z, x)|[\delta(s')]_+\right].$$
If $K(z, x) \ge 0$ on the relevant propagation path, this proxy reduces to the unsigned expression used in the main text.

C.1 Baseline OOD Propagation

We first prove Theorem 4.1. At each next state s', define the best data-supported and unsupported target-critic values as

$$q_{\mathrm{in}}(s') = \max_{a'\in\mathcal{A}_{\mathrm{in}}(s')} \bar{Q}(s', a'), \qquad q_{\mathrm{out}}(s') = \max_{a'\in\mathcal{A}_{\mathrm{out}}(s')} \bar{Q}(s', a'). \tag{16}$$

The OOD preference gap is

$$\delta(s') = q_{\mathrm{out}}(s') - q_{\mathrm{in}}(s'). \tag{17}$$

A positive $\delta(s')$ means that the target critic ranks an unsupported action above the best data-supported alternative.

Consider two Bellman targets. The data-supported counterfactual target is

$$y_{\mathrm{in}}(x) = r + \gamma q_{\mathrm{in}}(s'), \tag{18}$$

while the standard greedy actor-critic target is

$$y_{\mathrm{std}}(x) = r + \gamma \max\{q_{\mathrm{in}}(s'), q_{\mathrm{out}}(s')\}. \tag{19}$$

Since

$$\max\{q_{\text{in}}, q_{\text{out}}\} = q_{\text{in}} + [q_{\text{out}} - q_{\text{in}}]_+,$$

we have

$$y_{\text{std}}(x) - y_{\text{in}}(x) = \gamma[\delta(s')]_+. \tag{20}$$

Thus, $[\delta(s')]_+$ is the actor-side OOD increment admitted into the target.

Define the squared Bellman losses

$$L_{\text{std}}(\theta) = \frac{1}{2}\mathbb{E}_{x\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - y_{\text{std}}(x)\right)^2\right], \tag{21}$$

and

$$L_{\text{in}}(\theta) = \frac{1}{2}\mathbb{E}_{x\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - y_{\text{in}}(x)\right)^2\right]. \tag{22}$$

Let

$$\theta^+_{\text{std}} = \theta - \eta\nabla_\theta L_{\text{std}}(\theta), \qquad \theta^+_{\text{in}} = \theta - \eta\nabla_\theta L_{\text{in}}(\theta).$$

Since both targets are fixed with respect to θ during this update,

$$\nabla_\theta L_{\text{std}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - y_{\text{std}}(x)\right)\nabla_\theta Q_\theta(s,a)\right], \tag{23}$$

$$\nabla_\theta L_{\text{in}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[\left(Q_\theta(s,a) - y_{\text{in}}(x)\right)\nabla_\theta Q_\theta(s,a)\right]. \tag{24}$$

Subtracting the two parameter updates gives

$$\theta^+_{\text{std}} - \theta^+_{\text{in}} = \eta\mathbb{E}_{x\sim\mathcal{D}}\left[\left(y_{\text{std}}(x) - y_{\text{in}}(x)\right)\nabla_\theta Q_\theta(s,a)\right] \tag{25}$$

$$= \gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[[\delta(s')]_+\nabla_\theta Q_\theta(s,a)\right], \tag{26}$$

where the second equality uses Eq. equation 20.

The OOD-induced value change at query point z is

$$\Delta Q^{\text{std}}(z) = Q_{\theta^+_{\text{std}}}(z) - Q_{\theta^+_{\text{in}}}(z).$$

By a first-order Taylor expansion around θ,

$$\Delta Q^{\text{std}}(z) = \nabla_\theta Q_\theta(z)^\top\left(\theta^+_{\text{std}} - \theta^+_{\text{in}}\right) + O(\eta^2) \tag{27}$$

$$= \gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[\nabla_\theta Q_\theta(z)^\top\nabla_\theta Q_\theta(s,a)[\delta(s')]_+\right] + O(\eta^2) \tag{28}$$

$$= \gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[K(z,x)[\delta(s')]_+\right] + O(\eta^2). \tag{29}$$

This proves Eq. equation 9. The factor $[\delta(s')]_+$ is actor-side admission, and $K(z,x)$ is critic-side propagation through shared parameters.

C.2 Critic-Side Contraction with Imperfect Anchor

We next prove Theorem 4.2. URAC uses the corrected bootstrap value

$$\widetilde{Q}(s,a) = \bar{Q}(s,a) - w(s,a)[\bar{Q}(s,a) - \mu(s,a)]_+, \tag{30}$$

where

$$w(s,a) = \frac{u(s,a)}{u(s,a)+c}, \qquad c > 0. \tag{31}$$

During target construction, μ, u, and w are treated as fixed reliability statistics.

Local transmission factor. We first compute the sensitivity of the corrected value with respect to the raw target value. If $\bar{Q}(s,a) \le \mu(s,a)$, then

$$[\bar{Q}(s,a) - \mu(s,a)]_+ = 0, \qquad \widetilde{Q}(s,a) = \bar{Q}(s,a),$$

and therefore

$$\frac{\partial\widetilde{Q}(s,a)}{\partial\bar{Q}(s,a)} = 1. \tag{32}$$

If $\bar{Q}(s,a) > \mu(s,a)$, then

$$\widetilde{Q}(s,a) = \bar{Q}(s,a) - w(s,a)(\bar{Q}(s,a) - \mu(s,a)) = (1 - w(s,a))\bar{Q}(s,a) + w(s,a)\mu(s,a),$$

so

$$\frac{\partial \widetilde{Q}(s,a)}{\partial \bar{Q}(s,a)} = 1 - w(s,a). \tag{33}$$

Combining the two cases gives

$$\frac{\partial \widetilde{Q}(s,a)}{\partial \bar{Q}(s,a)} = 1 - w(s,a)\mathbf{1}\{\bar{Q}(s,a) > \mu(s,a)\}. \tag{34}$$

Define

$$\rho(s,a) = w(s,a)\mathbf{1}\{\bar{Q}(s,a) > \mu(s,a)\}. \tag{35}$$

Then the local transmission factor is

$$\eta_Q(s,a) := \frac{\partial \widetilde{Q}(s,a)}{\partial \bar{Q}(s,a)} = 1 - \rho(s,a). \tag{36}$$

Since $w(s,a) = u(s,a)/(u(s,a)+c)$, we have $0 \le w(s,a) < 1$. Moreover, in the optimistic regime $\bar{Q}(s,a) > \mu(s,a)$,

$$\frac{\partial \rho(s,a)}{\partial u(s,a)} = \frac{c}{(u(s,a)+c)^2} > 0. \tag{37}$$

Thus, the contraction becomes stronger as uncertainty increases in above-anchor regions, and it vanishes below the anchor.

For a small perturbation $d\bar{Q}(s',a')$, Eq. equation 36 gives

$$d\widetilde{Q}(s',a') = (1 - \rho(s',a'))d\bar{Q}(s',a') + o(d\bar{Q}).$$

Therefore, the corrected Bellman target

$$y^{\mathrm{URAC}} = r + \gamma \widetilde{Q}(s',a')$$

transmits only a fraction $1 - \rho(s',a')$ of a local optimistic perturbation. Substituting this into the NTK derivation in Appendix C.1 yields the effective propagation factor

$$K(z,x)(1 - \rho(s',a')). \tag{38}$$

Effect of imperfect anchor. We now show how the local anchor error ϵ_μ affects the contraction. Let

$$b_+(s,a) = [\bar{Q}(s,a) - Q_{\mathrm{ref}}(s,a)]_+$$

be the optimistic bias relative to the local data-supported reference Q_{ref}. Suppose the anchor error satisfies Eq. equation 15. We claim that

$$[\widetilde{Q}(s,a) - Q_{\mathrm{ref}}(s,a)]_+ \le (1 - w(s,a))[\bar{Q}(s,a) - Q_{\mathrm{ref}}(s,a)]_+ + \epsilon_\mu. \tag{39}$$

To prove this, write

$$b = \bar{Q}(s,a) - Q_{\mathrm{ref}}(s,a), \qquad e = \mu(s,a) - Q_{\mathrm{ref}}(s,a),$$

with $|e| \le \epsilon_\mu$. Then

$$\widetilde{Q}(s,a) - Q_{\mathrm{ref}}(s,a) = b - w(s,a)[b - e]_+.$$

If $b < 0$, then the left-hand side is non-positive after the one-sided correction, or its positive part is zero, so Eq. equation 39 holds. If $b \ge 0$ and $b - e > 0$, then

$$b - w[b - e] = (1 - w)b + we \le (1 - w)b + \epsilon_\mu.$$

If $b \ge 0$ and $b - e \le 0$, then $b \le e \le \epsilon_\mu$, and therefore

$$[b - w[b - e]_+]_+ = b \le \epsilon_\mu \le (1 - w)b + \epsilon_\mu.$$

Thus, Eq. equation 39 holds in all cases.

This shows that the critic-side correction contracts optimistic bias up to the local anchor error. When this target-level error is propagated through one critic update, it contributes an additional $O(\eta\epsilon_\mu)$ term under the same bounded-kernel condition used in the NTK expansion.

C.3 Actor-Side Admission Contraction

We prove Theorem 4.3. Let the behavior anchor be

$$a_b(s) = \pi_b(s),$$

and let the residual actor propose

$$a_p(s) = a_b(s) + \Delta a_\phi(s).$$

URAC accepts only a gated fraction of this proposal:

$$a_\pi(s) = a_b(s) + g_\psi(s)\Delta a_\phi(s), \qquad g_\psi(s) \in [0,1]. \tag{40}$$

Define the accepted residual magnitude as

$$\alpha_\psi(s) = g_\psi(s)\|\Delta a_\phi(s)\|_2. \tag{41}$$

For a transition $x = (s, a, r, s')$, define the behavior-anchored target and the URAC actor target:

$$y_b = r + \gamma\bar{Q}(s', a_b(s')), \qquad y_\pi = r + \gamma\bar{Q}(s', a_\pi(s')). \tag{42}$$

By local M-Lipschitzness of $\bar{Q}(s', \cdot)$,

$$|y_\pi - y_b| = \gamma\left|\bar{Q}(s', a_\pi(s')) - \bar{Q}(s', a_b(s'))\right| \tag{43}$$

$$\leq \gamma M\|a_\pi(s') - a_b(s')\|_2 \tag{44}$$

$$= \gamma M g_\psi(s')\|\Delta a_\phi(s')\|_2 \tag{45}$$

$$= \gamma M \alpha_\psi(s'). \tag{46}$$

This proves the first part of Theorem 4.3. It also shows that the Bellman target deviation is linear in the accepted action deviation.

We now show why the optimal accepted magnitude decreases with uncertainty. Fix a state s, write

$$a_b = a_b(s), \qquad \Delta a = \Delta a_\phi(s), \qquad u_p = u_p(s).$$

If $\Delta a = 0$, then $\alpha_\psi = 0$ and the claim is trivial. Otherwise, define the unit residual direction

$$v = \frac{\Delta a}{\|\Delta a\|_2}.$$

Let

$$\alpha = g\|\Delta a\|_2,$$

so that

$$a_b + g\Delta a = a_b + \alpha v.$$

Since u_p is detached, the local actor objective along this direction can be written as

$$\max_{\alpha \in [0, \|\Delta a\|_2]} \; Q_\theta(s, a_b + \alpha v) - u_p\alpha^2. \tag{47}$$

Assume $Q_\theta(s, \cdot)$ is locally smooth along v. Using a local quadratic approximation,

$$Q_\theta(s, a_b + \alpha v) \approx Q_\theta(s, a_b) + \alpha\nabla_a Q_\theta(s, a_b)^\top v - \frac{L}{2}\alpha^2. \tag{48}$$

Let

$$B_v = \nabla_a Q_\theta(s, a_b)^\top v.$$

Then Eq. equation 47 becomes the one-dimensional surrogate

$$\max_{\alpha \in [0, \|\Delta a\|_2]} \; Q_\theta(s, a_b) + \alpha B_v - \left(\frac{L}{2} + u_p\right)\alpha^2.$$

Its maximizer is

$$\alpha^\star = \left[\frac{[B_v]_+}{L + 2u_p}\right]_{[0, \|\Delta a\|_2]}, \tag{49}$$

where $[\cdot]_{[0,\|\Delta a\|_2]}$ denotes clipping to the feasible interval. The positive part appears because a direction with negative directional gain should not be accepted.

If the directional gain B_v is locally bounded, then

$$\alpha^\star = O\left(\frac{1}{u_p}\right). \tag{50}$$

Thus, increasing uncertainty shrinks the optimal accepted residual magnitude. The corresponding gate is $g^\star = \alpha^\star / \|\Delta a\|_2$ after clipping, so the gate should be interpreted as a mechanism for controlling the accepted deviation rather than as the primary object of the Lipschitz bound.

C.4 Closed-Loop Multiplicative Contraction

We now prove Theorem 4.4. The baseline decomposition from Appendix C.1 can be written as

$$\Delta Q^{\mathrm{std}}(z) = \gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[K(z,x)e_{\mathrm{std}}(s')\right] + O(\eta^2), \tag{51}$$

where

$$e_{\mathrm{std}}(s') = [\delta(s')]_+$$

is the OOD increment admitted by the standard actor.

For URAC, let $e_{\mathrm{URAC}}(s')$ denote the corresponding increment after gated residual admission. We define the actor-side admission factor $\beta_A \in [0,1]$ by

$$e_{\mathrm{URAC}}(s') \le \beta_A e_{\mathrm{std}}(s'). \tag{52}$$

This factor summarizes the reduction in admitted target deviation induced by the accepted residual magnitude α_ψ.

From Appendix C.2, the critic-side correction transmits local optimistic target perturbations with factor $1-\rho(s',a')$. Thus, up to the imperfect-anchor error term, URAC satisfies

$$\Delta Q^{\mathrm{URAC}}(z) = \gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[K(z,x)(1-\rho(s',a'))e_{\mathrm{URAC}}(s')\right] + O(\eta^2) + O(\eta\epsilon_\mu). \tag{53}$$

Let

$$\rho_{\min} = \operatorname*{ess\,inf}_{(s',a')} \rho(s',a')$$

over the relevant next-state actions entering the backup. Then

$$1-\rho(s',a') \le 1-\rho_{\min}.$$

If $K(z,x) \ge 0$ on the relevant propagation path, then using Eq. equation 52 gives

$$\begin{aligned}
|\Delta Q^{\mathrm{URAC}}(z)| &\le \gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[K(z,x)(1-\rho(s',a'))e_{\mathrm{URAC}}(s')\right] + O(\eta^2) + O(\eta\epsilon_\mu) && (54)\\
&\le (1-\rho_{\min})\beta_A\gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[K(z,x)e_{\mathrm{std}}(s')\right] + O(\eta^2) + O(\eta\epsilon_\mu) && (55)\\
&= (1-\rho_{\min})\beta_A|\Delta Q^{\mathrm{std}}(z)| + O(\eta^2) + O(\eta\epsilon_\mu). && (56)
\end{aligned}$$

This proves the no-cancellation version of the closed-loop bound.

For signed empirical kernels, the same argument gives the magnitude-safe bound

$$|\Delta Q^{\mathrm{URAC}}(z)| \le (1-\rho_{\min})\beta_A\mathcal{B}^{\mathrm{std}}(z) + O(\eta^2) + O(\eta\epsilon_\mu), \tag{57}$$

where

$$\mathcal{B}^{\mathrm{std}}(z) = \gamma\eta\mathbb{E}_{x\sim\mathcal{D}}\left[|K(z,x)|[\delta(s')]_+\right].$$

When $K(z,x) \ge 0$, this reduces to the main-text form.

The multiplicative structure follows directly: the critic module reduces bootstrap transmission by $1-\rho$, while the actor module reduces the admitted OOD increment by β_A. If the critic module is removed, then $\rho \equiv 0$. If the actor module is removed, then $\beta_A = 1$. Thus, each module controls a different stage of the closed-loop pathway, and their effects combine multiplicatively.

C.5 From Local Contraction to Multi-Step Bias Control

The previous results analyze one critic update. We now show how the local contraction can be unrolled into a multi-step bias-control bound under uniform stability assumptions. This result is not a global convergence guarantee to the optimal value function; it only states that OOD-induced bias remains bounded when the local loop gain is uniformly below one.

Let $b_t(z)$ denote the OOD-induced value bias at update step t, measured relative to the data-supported counterfactual update. Consider a norm $\|\cdot\|_{\mathcal{Z}}$ over a query region $\mathcal{Z}$. Suppose the standard actor-critic update induces a local bias-propagation operator with norm bounded by L_{std}:

$$\|b_{t+1}^{\mathrm{std}}\|_{\mathcal{Z}} \le L_{\mathrm{std}}\|b_t^{\mathrm{std}}\|_{\mathcal{Z}} + \epsilon_t, \tag{58}$$

where ϵ_t collects residual approximation error, stochastic optimization error, and newly injected target noise.

For URAC, define the local loop-gain factor

$$\chi = (1 - \rho_{\min})\beta_A.$$

Under the uniform bound implied by Theorem 4.4, the URAC bias recursion satisfies

$$\|b_{t+1}^{\mathrm{URAC}}\|_{\mathcal{Z}} \le \chi L_{\mathrm{std}} \|b_t^{\mathrm{URAC}}\|_{\mathcal{Z}} + \epsilon_t + C\eta\epsilon_\mu + O(\eta^2), \tag{59}$$

where $C > 0$ absorbs the local bounded-kernel constants. Let

$$\lambda = \chi L_{\mathrm{std}}.$$

If $\lambda < 1$ and $\epsilon_t \le \epsilon$ uniformly, then unrolling Eq. equation 59 gives

$$\|b_T^{\mathrm{URAC}}\|_{\mathcal{Z}} \le \lambda^T \|b_0^{\mathrm{URAC}}\|_{\mathcal{Z}} + \frac{1-\lambda^T}{1-\lambda}\left(\epsilon + C\eta\epsilon_\mu + O(\eta^2)\right). \tag{60}$$

Consequently,

$$\limsup_{T\to\infty} \|b_T^{\mathrm{URAC}}\|_{\mathcal{Z}} \le \frac{\epsilon + C\eta\epsilon_\mu + O(\eta^2)}{1-\lambda}. \tag{61}$$

In the idealized case where $\epsilon = 0$, $\epsilon_\mu = 0$, and higher-order terms vanish, the OOD-induced bias decays geometrically at rate λ. With bounded residual error, the long-horizon bias remains bounded within a radius that decreases as the URAC loop-gain factor $\chi = (1 - \rho_{\min})\beta_A$ becomes smaller. Thus, the local contraction yields global-in-time control of OOD-induced bias under uniform stability assumptions, but it does not establish global convergence of nonlinear actor-critic learning.

D Additional Experimental Details

This appendix provides additional implementation details for the empirical evaluation in Section 5. The main text focuses on performance comparison and result analysis, while this appendix describes the experimental settings needed to reproduce the results. We include the computing resources, hyperparameters, ablation variants, and the construction protocols for the limited-data and noisy-data experiments.

D.1 Computing Resources

All experiments are conducted on NVIDIA A100 GPUs. Each training run uses a single A100 GPU. URAC contains two main computational stages: the auxiliary uncertainty-estimator stage and the actor-critic training stage. The consistency-model uncertainty estimator is trained before actor-critic learning and then frozen. On the representative Hopper-medium task, training this consistency model takes approximately 15 minutes on a single A100 GPU. After the uncertainty estimator is trained, one URAC actor-critic training run on Hopper-medium takes approximately 2 hours on a single A100 GPU.

The consistency-model cost is therefore relatively small compared with the main actor-critic optimization cost. When the same dataset is used across multiple random seeds or ablation variants, the trained uncertainty estimator can be reused, so this preprocessing cost can be amortized across runs. For robustness experiments under limited-data or noisy-data settings, the estimator is retrained on the corresponding modified dataset before URAC training.

The exact wall-clock time may vary across environments and experimental settings. Different tasks, dataset sizes, ablation variants, and robustness settings require different amounts of data preprocessing, uncertainty-estimator training, and actor-critic optimization. In general, the total compute scales approximately linearly with the number of random seeds and dataset variants evaluated.

D.2 Hyperparameters

The hyperparameters used for URAC and the consistency-model uncertainty estimator are summarized in Table 4. The consistency model is trained before actor-critic learning and remains frozen during policy optimization. Unless otherwise specified, we use the hyperparameter configuration in Table 4 throughout our experiments.

Table 4: Hyperparameters used in URAC and the consistency-model uncertainty estimator.

Hyperparameter	Value
URAC actor-critic training	
Optimizer	AdamW for critic, residual actor, and gate
Learning rate	Critic: 3×10^{-4}; actor: 3×10^{-4}; gate: 3×10^{-4}
Batch size	256
Target update rate	0.005
Critic architecture	input–256–256–1
Residual actor architecture	input–256–256–output
Gate architecture	(critic feature + $\mu(s,a)$ + $\log u(s,a)$)–256–256–1
CM samples	16
Number of iterations	1e6
Standard shrinkage constant c	1.0
Consistency-model uncertainty estimator	
Optimizer	AdamW
Learning rate	3×10^{-4}
Weight decay	1×10^{-5}
Batch size	512 as parser default
Architecture	input–256–ResidualBlock(256)–ResidualBlock(256)–1

D.3 Details of Ablation Studies

We provide additional details for the ablation variants used in Table 2. All ablation variants are trained under the same evaluation protocol as the full URAC model, and only the specified component is modified. The purpose of these ablations is to isolate the roles of the value anchor, critic-side uncertainty correction, and actor-side uncertainty gating.

w/o value anchor. This variant removes the value anchor from both the critic-side correction and the actor-side gate. In the critic, the data-supported reference value $\mu(s,a)$ is removed from the uncertainty correction, so the critic no longer identifies whether a bootstrap value lies above a data-grounded reference. In the actor, $\mu(s,a_p)$ is also removed from the gate input, where a_p denotes the residual proposal action. As a result, the gate is guided only by the critic feature and the uncertainty scale. This ablation tests whether the uncertainty scale alone is sufficient for coupling control, without an explicit value reference indicating what value level is supported by the offline data.

Uncertainty in Critic only. This variant keeps the uncertainty-aware critic used in URAC, but replaces the gated residual actor with a standard behavior-regularized actor. Specifically, the actor is trained by maximizing the critic value while being regularized toward the behavior policy:

$$J_\pi^{\text{critic-only}} = \mathbb{E}_{s\sim\mathcal{D}}\left[Q_\theta(s,\pi(s)) - \lambda_{\text{bc}}\|\pi(s)-\pi_b(s)\|_2^2\right],$$

where π_b is the behavior cloning policy and λ_{bc} controls the strength of behavior regularization. Thus, this variant keeps critic-side uncertainty correction but removes the uncertainty-aware residual gate on the actor side. It evaluates whether critic-side uncertainty alone is sufficient to prevent closed-loop error amplification when actor-side admission is controlled only by behavior regularization.

Uncertainty in Actor only. This variant keeps the uncertainty-gated residual actor, but replaces the URAC critic correction with a CQL-style conservative critic penalty. The critic is trained with the standard Bellman error together with a conservative regularization term:

$$\mathcal{L}_Q^{\text{actor-only}} = \mathcal{L}_{\text{Bellman}} + \alpha_{\text{cql}}\mathcal{L}_{\text{CQL}},$$

where $\mathcal{L}_{\text{CQL}}$ penalizes high Q-values on actions not supported by the dataset following the standard CQL objective. In continuous-control tasks, this term is estimated using sampled actions as in CQL. This ablation removes the URAC uncertainty-aware bootstrap correction, while retaining the

actor-side uncertainty gate. It tests whether actor-side uncertainty control alone is sufficient when critic-side bootstrap transmission is not explicitly regulated by the URAC uncertainty signal.

Overall, these ablations show whether each component is necessary for coupling control. The full URAC model differs from these variants by using the same uncertainty signal to jointly regulate both critic-side bootstrap transmission and actor-side residual acceptance.

D.4 Details of Limited and Noisy Data Experiments

We evaluate robustness under limited-data and noisy-data settings on the D4RL Gym-MuJoCo medium-expert datasets, including Hopper-medium-expert, Walker2d-medium-expert, and HalfCheetah-medium-expert. For each modified dataset, all methods are trained from scratch using the same data construction protocol and evaluated with the same normalized return metric as in the main experiments.

Limited-data setting. The limited-data experiment simulates reduced data coverage by randomly discarding a portion of the original offline dataset. Given an original dataset $\mathcal{D}$ and a discard ratio $r \in [0, 1)$, we construct a reduced dataset

$$\mathcal{D}_r^{\text{limited}} \subset \mathcal{D}, \qquad |\mathcal{D}_r^{\text{limited}}| = (1 - r)|\mathcal{D}|.$$

The removed fraction r is reported as the *Discard Ratio* in Figure 3. A larger discard ratio means that fewer transitions remain for offline training, which reduces data coverage and makes value-guided policy improvement more dependent on extrapolated critic estimates. For fairness, URAC and all baselines are trained on the same reduced dataset under each discard ratio.

Noisy-data setting. The noisy-data experiment simulates imperfect demonstrations by adding noisy data into the original offline dataset. Let $\mathcal{D}_{\text{clean}}$ denote the original medium-expert dataset and $\mathcal{D}_{\text{noisy}}$ denote the noisy data added to the training set. For a given noise ratio ρ, we construct a mixed dataset

$$\mathcal{D}_\rho^{\text{noisy}} = \mathcal{D}_{\text{clean}} \cup \mathcal{D}_{\text{noisy}},$$

where the proportion of noisy samples in the training dataset is controlled by ρ. The value of ρ is reported as the *Noise Ratio* in Figure 4. A larger noise ratio indicates that a larger fraction of the training data comes from noisy or low-quality samples, which can degrade value supervision and increase the risk of misleading critic estimates. All algorithms are trained and evaluated under the same noisy dataset for each noise ratio.

Interpretation of gate statistics. In Figures 3b and 4b, we report the average gate coefficient of URAC under different data conditions. The gate coefficient should not be interpreted as uncertainty itself. Instead, it measures the accepted magnitude of the residual deviation from the behavior anchor after considering critic features, the value anchor, and the uncertainty scale. Therefore, changes in the average gate reflect how URAC adjusts actor-side coupling strength when the offline data become less complete or noisier.