# THE LOW-RANK HANKEL HYPOTHESIS IN REINFORCEMENT LEARNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Identifying and exploiting structure in reinforcement learning (RL) can guide exploration and improve sample efficiency, particularly when environment interactions are costly. We investigate the *low-rank Hankel hypothesis*: value-function sequences evaluated along trajectories of high-performing policies exhibit low effective Hankel rank, and can therefore be approximated by low-order linear recurrences. We provide empirical evidence for this structure in learned critic outputs across Classical Control, MuJoCo, and Atari. Theoretically, geometric mixing yields low-rank approximations of conditional expected Hankel matrices. Motivated by these findings, we introduce Q-Hankeling, a simple method that regularizes Q-function learning using short trajectory chunks sampled from replay. The method learns linear recurrence coefficients jointly with the critic and penalizes recurrence residuals within these chunks. This provides a temporal-predictability inductive bias for value estimation, without explicitly constructing Hankel matrices or computing singular-value decompositions. We demonstrate how Q-Hankeling can be integrated into value-based methods, such as DQN and Efficient Rainbow, and actor–critic methods, such as TD3 and SAC. On Atari-100k, Q-Hankeling increases the mean human-normalized score from 0.463 to 0.530 across 26 games, an improvement of approximately 14%. Across the reported MuJoCo tasks, Q-Hankeling improves final return at 1M environment steps in seven of eight evaluated SAC/TD3 task–algorithm combinations.

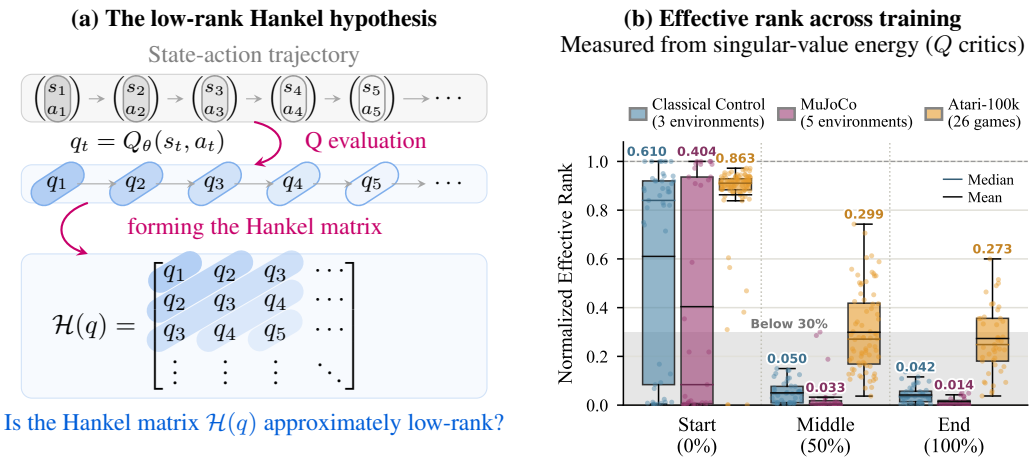



Figure 1: **Evidence of low effective Hankel rank across three benchmark suites. (a)** The low-rank Hankel hypothesis. **(b)** Points show normalized effective rank $r_\epsilon/d$ at $\epsilon = 0.1\%$[1]; boxes pool rollout ranks across seeds, environments, and algorithms within each benchmark suite. Classical Control (DQN) and MuJoCo (TD3/SAC) use Stable-Baselines3 (Raffin et al., 2021); Atari-100k uses Efficient Rainbow. Further details can be found in Appendix D.3. To exploit this structure, we introduce Q-Hankeling, which encourages the learned Q-function to possess a low-rank Hankel structure.

[1]For a Hankel matrix with singular values $\sigma_1 \geq \cdots \geq \sigma_d$, the normalized effective rank at tolerance $\epsilon$ is $r_\epsilon/d$, where $r_\epsilon = \min\{k : \sum_{i>k} \sigma_i^2 / \sum_i \sigma_i^2 < \epsilon\}$ and $d = \min(\text{rows}, \text{cols})$.



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.

# 1 INTRODUCTION

Reinforcement learning (RL) has achieved striking empirical success, yet learning effective policies from limited environment interaction remains a central challenge. Sample-efficient RL requires generalizing from limited trajectories to decisions beyond the observed transitions. RL theory shows that exploitable structure supports such generalization (Ok et al., 2018; Jin et al., 2020; Jiang et al., 2017; Sun et al., 2019). Structural assumptions can reduce exploration and estimation complexity by capturing dependencies across state–action pairs. In deep RL, however, useful structure is often introduced less explicitly, through representation learning, architectural inductive biases, auxiliary objectives, replay mechanisms, and regularization (Schwarzer et al., 2020; Laidlaw et al., 2023).

Low rank is one particularly successful notion of structure. Existing work has exploited low-dimensional factorizations of Bellman errors (Jiang et al., 2017; Sun et al., 2019), transition dynamics (Jin et al., 2020; Zhou et al., 2021; Agarwal et al., 2020), latent state representations (Du et al., 2019; Jedra et al., 2023), and state–action value matrices (Shah et al., 2020; Sam et al., 2023; Stojanovic et al., 2024). Related ideas also appear in successor and forward–backward representations for transferring knowledge across rewards and tasks (Dayan, 1993; Touati et al., 2022; Dubail et al., 2025). These formulations capture low-rank structure across states, actions, next states, or goals. We investigate a different axis along which low-dimensional structure may arise—*time*. Rather than asking whether $Q(s, a)$ is globally low rank over states and actions, we ask whether the sequence of values encountered along a trajectory has low temporal complexity.

Hankel matrices are ubiquitous across mathematics and engineering, with a classical theory developed in Peller (2003). Low-rank Hankel and related matrix representations underpin denoising in signal processing (Cadzow, 1988), time-series analysis (Golyandina et al., 2001; Agarwal et al., 2018), and system identification and realization in control theory (Ho & Kálmán, 1966; Fazel et al., 2013). Despite the relevance of this structure across these fields, its use as a structural prior for value sequences remains largely unexplored in RL.

In this work, we investigate the low-rank Hankel structure in RL. For a trajectory generated by $\pi$, consider the value sequences $q_t^\pi = Q^\pi(s_t, a_t)$ and $v_t^\pi = V^\pi(s_t)$. Arranging overlapping windows of either sequence as columns gives a Hankel matrix, as illustrated in Figure 1. We formulate the *low-rank Hankel hypothesis*: value sequences along trajectories of high-performing policies exhibit low effective Hankel rank. We investigate how consistently this structure appears and whether encouraging it improves sample efficiency.

**Contributions.** Below, we summarize our main contributions.

(i) *Empirical evidence of the low-rank Hankel structure.* We observe low effective Hankel rank in learned critic value sequences across Classical Control, MuJoCo, and Atari (§ 4.2). Figure 1 summarizes ranks across training; Figure 2 illustrates the corresponding singular-value decay.

(ii) *Mixing yields low-rank approximations of expected Hankel matrices.* Under geometric mixing, Theorem 4.2 represents the value sequence as a noisy order-$r$ linear recurrence whose conditional approximation error decays exponentially in $r$ at a rate determined by the mixing properties. Together with the recurrence–rank connection in Theorem 4.1, the mixing analysis establishes approximate low-rank Hankel structure in conditional expectation (§ 4.3, Eq. (16)).

(iii) *Exploiting the low-rank Hankel structure.* Motivated by Theorem 4.1, we introduce Q-Hankeling, which learns recurrence coefficients jointly with the critic and avoids Hankel-matrix construction and decomposition (§ 5, Algorithm 1). It integrates with DQN, Efficient Rainbow, SAC, and TD3.

(iv) *Empirical gains through Q-Hankeling.* On Atari-100k, Q-Hankeling raises mean human-normalized score from 0.463 to 0.530 across 26 games (§ 6, Figure 17). On the evaluated MuJoCo tasks, it improves final return at one million environment steps for seven of the eight reported SAC/TD3 task–algorithm combinations (§ 6, Table 1); Figure 10 reports steps to return thresholds.

## 2 Related Work

**Low-rank structure in reinforcement learning.** Low-dimensional structure supports efficient RL through linear and low-rank transition models (Jin et al., 2020; Zhou et al., 2021; Agarwal et al., 2020), latent-state representations (Du et al., 2019; Jedra et al., 2023), and Bellman- or witness-rank conditions (Jiang et al., 2017; Sun et al., 2019). Closer to our setting, Shah et al. (2020); Stojanovic et al. (2024) exploit low-rank state–action value matrices through matrix estimation. Sam et al. (2023) examine the additional structural assumptions needed to obtain efficient learning over long horizons in this setting. Our hypothesis concerns a different matrix: the Hankel matrix of scalar values evaluated along a trajectory. It can be investigated from sampled trajectories without assuming a low-rank factorization of the full state–action value matrix or transition kernel.

**Successor representations and mixing.** Successor representations describe states through expected discounted future occupancies (Dayan, 1993). Successor features and forward–backward representations use related descriptions to transfer across rewards, including in zero-shot RL (Touati et al., 2022). Of particular relevance, Dubail et al. (2025) show how bypassing initial transitions can expose low-rank structure in a shifted successor measure, and relate the required shift to local mixing properties. Both their work and ours investigate how dynamics can give rise to useful low-dimensional structure. The objects differ: they approximate a successor measure indexed by states and future states, whereas we study Hankel matrices indexed by positions within value trajectories. Our mixing analysis controls the conditional bias of a value recurrence and bounds stochastic fluctuations under additional conditions.

**Temporal abstraction and predictive learning.** Options formalize temporally extended actions within the semi-MDP framework (Sutton et al., 1999). More recent work studies action chunking in behavior cloning (Zhang et al., 2026) and RL. Q-chunking learns in a chunked action space for offline-to-online RL (Li et al., 2025), while CQN-AS learns values for action sequences (Seo & Abbeel, 2025). On the critic side, Tian et al. (2026) condition value estimates on trajectory segments and train with multi-step targets. These approaches use temporal context to change the actions evaluated or the information supplied to the critic. Temporal prediction also provides an auxiliary training signal: self-predictive representations improve data efficiency through prediction of future latent representations (Schwarzer et al., 2020). Q-Hankeling uses this predictive principle directly on scalar Q-values, learning a linear recurrence on trajectory chunks without requiring a sequence-conditioned critic or a chunked action space.

**Hankel matrices and linear recurrences.** Hankel matrices and operators have a well-developed classical theory (Peller, 2003), with applications to structured matrix estimation and system identification. In realization theory, Hankel rank reveals the order of a minimal linear realization (Ho & Kálmán, 1966). Singular spectrum analysis exploits low-rank Hankel trajectory matrices to estimate time-series components and construct recurrence-based forecasts (Golyandina et al., 2001). For system identification and realization, Fazel et al. (2013) formulate structured matrix estimation through Hankel rank minimization and study nuclear-norm relaxation, using $\|H\|_* = \sum_j \sigma_j(H)$ as a convex surrogate for rank. These methods motivate using Hankel structure as a prior on temporal sequences. Q-Hankeling encourages this structure in critic values through recurrence residuals, avoiding explicit Hankel-matrix construction and singular-value decompositions. Its penalty is not equivalent to nuclear-norm regularization.

## 3 Preliminaries

**Markov decision processes.** Consider a discounted Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $p(\cdot \mid s, a)$ is the distribution of the next state after taking action $a$ in state $s$, $r(s, a)$ is the expected immediate reward, and $\gamma \in (0, 1)$ is the discount factor. We assume $|r(s, a)| \leq r_{\max}$. For a given stationary Markov policy $\pi$, we define its state-action value and state value functions as:

$$Q^\pi(s, a) = \mathbb{E}_\pi[\textstyle\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a], \quad V^\pi(s) = \mathbb{E}_\pi[\textstyle\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s]. \tag{1}$$

Given an initial-state distribution $\mu$, the learning objective is to find a policy which maximizes $\mathbb{E}_{s_0\sim\mu}[V^\pi(s_0)]$ using as few environment interactions as possible. The value functions satisfy the Bellman equations

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

$$V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma \mathbb{E}_{s'\sim p(\cdot|s,a)}[V^\pi(s')]\right]. \tag{3}$$

**Policy-induced dynamics and mixing phenomena.** For a stationary Markov policy $\pi$, define $P^\pi(\cdot|s) := \mathbb{E}_{a\sim\pi(\cdot|s)}[p(\cdot \mid s,a)]$, and $R^\pi(s) := \mathbb{E}_{a\sim\pi(\cdot|s)}[r(s,a)]$, with $(P^\pi f)(s) := \mathbb{E}_{s'\sim P^\pi(\cdot|s)}[f(s')]$. Then $V^\pi = R^\pi + \gamma P^\pi V^\pi$; in finite state spaces, $V^\pi = (I - \gamma P^\pi)^{-1} R^\pi$. A stationary distribution satisfies $\nu^\pi P^\pi = \nu^\pi$. Writing $\|\mu - \nu\|_{\mathrm{TV}} := \sup_B |\mu(B) - \nu(B)|$ for total-variation distance, where $B$ ranges over measurable sets, uniform geometric mixing means that there exist $C > 0$ and $\rho \in (0,1)$ such that

$$\sup_{s\in\mathcal{S}} \|(P^\pi)^t(s,\cdot) - \nu^\pi\|_{\mathrm{TV}} \le C\rho^t, \qquad t \ge 0. \tag{4}$$

For finite irreducible, aperiodic chains, this holds with a unique stationary distribution (Levin & Peres, 2017, Theorem 4.9).

**Notation.** We use the usual $\ell_1$, Euclidean, and Frobenius norms, $\|\cdot\|_1$, $\|\cdot\|_2$, and $\|\cdot\|_{\mathrm{F}}$. For a function $f$, write $\|f\|_\infty := \sup_s |f(s)|$ and $\|f\|_\nu := (\mathbb{E}_{s\sim\nu}[f(s)^2])^{1/2}$ for its supremum and weighted $L^2(\nu)$ norms.

# 4 The Low-Rank Hankel Hypothesis

## 4.1 Hankel matrices and approximate recurrences

We start by introducing Hankel matrices, the central objects of this section. Formally, for a scalar sequence $q = (q_1, \ldots, q_T)$, the $m \times n$ Hankel matrix is

$$\mathcal{H}_{m,n}(q)_{i,j} = q_{i+j-1}, \qquad 1 \le i \le m, \quad 1 \le j \le n, \quad m+n-1 = T. \tag{5}$$

Each column is a shifted window of the sequence.

When the infinite Hankel matrix $\mathcal{H}_\infty(q) = (q_{i+j-1})_{i,j\ge1}$ is only approximately low rank, we seek an approximate recurrence for $q$. In the exact case, Kronecker's theorem (Kronecker, 1881; Al'pin, 2017) states that its rank is $r \ge 1$ if and only if $q$ satisfies a linear recurrence of minimal order $r$: $q_t = \sum_{i=1}^r c_i q_{t-i}$ for all $t > r$, with coefficients independent of $t$. For compact $\mathcal{H}_\infty(q)$, the Adamyan–Arov–Krein theorem (Peller, 1998, Theorem 6.1) provides a sequence $\bar{q}$ with Hankel rank at most $r$ and $\|q - \bar{q}\|_{\ell^2} \le \sigma_{r+1}(\mathcal{H}_\infty(q))$. Using the recurrence coefficients of $\bar{q}$ gives $q_t = \sum_{i=1}^r c_i q_{t-i} + \epsilon_t$ for $t > r$, with $\|\epsilon\|_{\ell^2} \le (1 + \|c\|_1)\sigma_{r+1}(\mathcal{H}_\infty(q))$. Thus the recurrence error is small when the neglected singular value is small and the coefficients remain bounded.

For finite matrices, the following theorem relates approximate low rank to approximate recurrences in both directions.

**Theorem 4.1** (Approximate Hankel rank and recurrences)**.** *Let $q \in \mathbb{R}^T$, $H = \mathcal{H}_{m,n}(q)$, $T = m+n-1$, and $1 \le r < \min(m,n)$.*

(i) *If $H_{:,1:r}$ has full column rank, there exist coefficients $c_1, \ldots, c_r$ such that*

$$q_t = \sum_{i=1}^r c_i q_{t-i} + \epsilon_t, \qquad r < t \le T, \qquad \max_{r<t\le T} |\epsilon_t| \le K_r(H)\,\sigma_{r+1}(H). \tag{6}$$

(ii) *Conversely, for any $c \in \mathbb{R}^r$ and the corresponding residuals $\epsilon_t = q_t - \sum_{i=1}^r c_i q_{t-i}$,*

$$\min_{\mathrm{rank}\, A \le r} \|H - A\|_{\mathrm{F}} \le J_{m,n,r}(c) \max_{r<t\le T} |\epsilon_t|. \tag{7}$$

*Here $K_r(H)$ and $J_{m,n,r}(c)$ are finite conditioning factors specified in the proof. Part (ii) requires no full-column-rank assumption.*

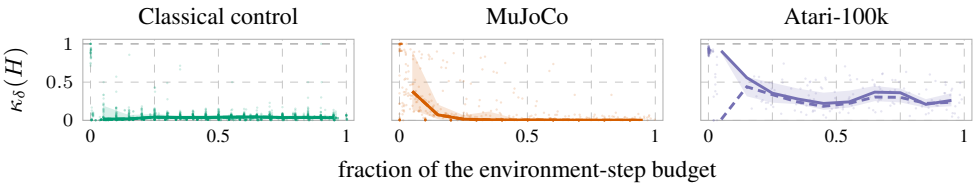



Figure 3: Normalized effective rank $\kappa_\delta$ of the rollout Hankel matrix over training. Solid: $q_t$ (median over ten budget bins; band: IQR; dots: analysis ticks); dashed: $v_t$.

The bounds are informative when their conditioning factors remain moderate. In the exact case, rank $r$ gives an order-$r$ recurrence under the assumption in part (i), while an exact order-$r$ recurrence always gives rank at most $r$. A relative rank-$r$ Frobenius approximation error of at most $\varepsilon$ also implies $\max_t |\epsilon_t| \le K_r(H)\varepsilon\|H\|_{\mathrm{F}}$. The constructions, proofs, and a counterexample showing why the forward assumption is needed appear in Appendix A.

### 4.2 Empirical evidence and scope

**Measurement.** We study critics trained without a Hankel penalty, following standard baselines: DQN on Classical Control and SAC and TD3 on MuJoCo use Stable-Baselines3 implementations (Raffin et al., 2021) with configurations based on RL Zoo (Raffin, 2020); Atari-100k uses Data-Efficient Rainbow (van Hasselt et al., 2019) in the DrQ configuration (Yarats et al., 2021). § 6 gives implementation details and benchmark-specific settings. At fixed environment-step checkpoints, we freeze the policy and critic and collect fresh rollouts using greedy actions for value-based policies, the actor mean for SAC, and the noiseless actor output for TD3. For each rollout, we form

$$q_t = Q_\theta(s_t, a_t), \qquad H = \mathcal{H}_{\lfloor T/2 \rfloor, \lceil T/2 \rceil + 1}(q). \tag{8}$$

For continuous-control policies, we stack three rollout matrices vertically. Writing $d = \min(\mathrm{rows}(H), \mathrm{cols}(H))$, we measure effective rank and its normalized ratio by

$$r_\delta(H) = \min\Big\{k \in \{0, \dots, d\} : \sum\nolimits_{j=k+1}^{d} \sigma_j(H)^2 \le \delta\|H\|_{\mathrm{F}}^2\Big\}, \qquad \kappa_\delta(H) = \frac{r_\delta(H)}{d}. \tag{9}$$

We use $\delta = 10^{-3}$, retaining $99.9\%$ of the Frobenius energy, and average the ratios over rollout samples and seeds.

**Observation.** Figure 2 provides an illustrative example from MountainCar: after training begins, the normalized Hankel singular values decay sharply beyond the leading mode. Figure 1 summarizes the rank ratios at the start, middle, and end of training, and Figure 3 tracks them at every analysis tick. Initial critics exhibit a broad range of ratios; the rank then collapses within the first tenth of the budget on Classical Control and MuJoCo and the first quarter on Atari. By the middle of training, the pooled ratio is below $0.06$ on Classical Control and MuJoCo and about $0.3$ on Atari, remaining low or decreasing further by the end. For trained MuJoCo critics, one to four singular values account for $99.9\%$ of the energy in matrices with over a thousand rows. These observations support the hypothesis that high-performing learned policies produce value trajectories with low effective Hankel rank. Further spectrum and rank plots for every five-seed family appear in Appendices E.3 and E.4.3.

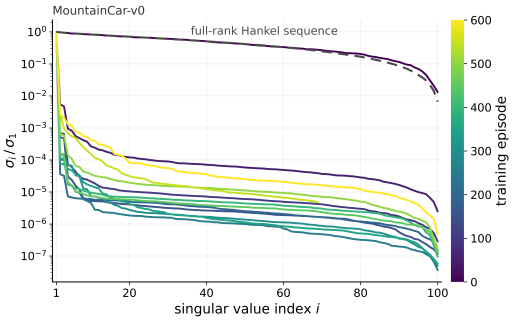



Figure 2: Normalized Hankel singular values $\sigma_i/\sigma_1$ for baseline DQN on MountainCar-v0 across training episodes (colour). Dashed: full-rank reference.

**Scope.** The evidence concerns learned critic outputs on the sampled trajectories and horizons, rather than exact policy values or all possible rollouts. Under the deterministic evaluation rule, $q_t$ equals $\max_a Q_\theta(s_t, a)$ for value-based policies and $Q_\theta(s_t, \pi_\phi(s_t))$ for actor-critic policies; comparing these sequences therefore provides no independent evidence about state values. In particular,

evaluating SAC at the actor mean does not recover its stochastic policy's state value. Effective rank depends on the energy threshold: at $\delta = 10^{-2}$, a single mode suffices on nearly every Atari game. A large constant component can dominate uncentred energy, so Appendix E.4.3 reports threshold sweeps, centred matrices, and per-environment ranks. Atari's higher ratios also indicate variation across domains. Finally, low rank within individual rollouts does not establish a common recurrence across replay windows; learning shared coefficients is an additional modelling choice of the method.

### 4.3 Mixing, recurrence bias, and stochastic fluctuations

We show how mixing gives an approximate linear recurrence for the value sequence, with a conditional approximation error controlled by the mixing rate and a separate stochastic fluctuation term. Fix a stationary policy $\pi$ with a finite, irreducible, and aperiodic transition matrix $P = P^\pi$. Write $\nu = \nu^\pi$ and let $C, \rho$ be its mixing constants in Eq. (4). Set $R = R^\pi$, $V = V^\pi = (I - \gamma P)^{-1} R$, and define

$$M_j(s) := (P^j V)(s) = \mathbb{E}_\pi[V(s_{t+j}) \mid s_t = s]. \tag{10}$$

For an order $r \geq 1$ and coefficients $c \in \mathbb{R}^r$, define the conditional bias $b_r(c) := M_r - \sum_{i=1}^r c_i M_{r-i}$. The value sequence satisfies the noisy recurrence

$$V(s_{t+r}) = \sum_{i=1}^r c_i V(s_{t+r-i}) + b_r(c)(s_t) + \xi_{t,r}(c), \tag{11}$$

where $\mathbb{E}_\pi[\xi_{t,r}(c) \mid s_t] = 0$. Writing $\epsilon_{t,r}(c) = b_r(c)(s_t) + \xi_{t,r}(c)$ for the recurrence residual, stationarity gives the orthogonal decomposition

$$\mathbb{E}_\pi[\epsilon_{t,r}(c)^2] = \|b_r(c)\|_\nu^2 + \mathbb{E}_\pi[\xi_{t,r}(c)^2]. \tag{12}$$

To preserve constant sequences, impose $\mathbf{1}^\top c = 1$ and choose conditional-bias optimal coefficients

$$c^\star \in \operatorname*{arg\,min}_{c \in \mathbb{R}^r : \mathbf{1}^\top c = 1} \|b_r(c)\|_\nu^2. \tag{13}$$

Thus $c^\star$ minimizes the squared conditional bias in Eq. (12), not the full mean-squared recurrence error, whose noise term also depends on $c$.

**Theorem 4.2** (Noisy value recurrence under mixing)**.** *Under the geometric-mixing assumptions above, let $s_t \sim \nu$. For every $r \geq 1$, the value sequence admits the order-$r$ linear recurrence in Eq. (11), with conditionally mean-zero noise and a conditional approximation error controlled by mixing. Specifically, an optimizer $c^\star$ in Eq. (13) exists, and every such optimizer satisfies*

$$\|b_r(c^\star)\|_\nu \leq \frac{2C r_{\max}(1+\rho)}{1-\gamma\rho} \rho^{r-1}, \tag{14}$$

$$\mathbb{E}_\pi[\xi_{t,r}(c^\star)^2] \leq W(c^\star)^2 \, \mathbb{E}_\pi[(V(s_{t+1}) - V(s_t))^2], \tag{15}$$

*where $r_{\max}$ is the reward bound from § 3 and $W(c) = \sum_{j=0}^{r-1} |\sum_{i=j+1}^r c_i|$. Thus the conditional approximation error decays exponentially with $r$ at the mixing rate $\rho$, while the noise is controlled separately.*

The bias bound follows by comparison with $c = (1, 0, \ldots, 0)$. For this choice, $\epsilon_{t,r} = V(s_{t+r}) - V(s_{t+r-1})$ has the same distribution for every $r$ under stationarity: the decaying bias reflects conditioning on an increasingly distant starting state, and does not imply a decreasing sampled error. The noise bound is small when successive value changes are small and $W(c^\star)$ is moderate; conditional-bias optimality alone does not control $W(c^\star)$. Constraining $\|c\|_1 \leq B$ for fixed $B \geq 1$ gives a noise bound uniform in $r$ for the resulting optimizer (Corollary B.5). Proofs appear in Appendix B.

**State-action values and expected Hankel rank.** For finite state and action spaces, the same analysis applies to $Q^\pi$ through the state-action chain $(s_t, a_t)$, which inherits geometric mixing with constants $C_Q, \rho$. For $q_t = Q^\pi(s_{t-1}, a_{t-1})$, let $\overline{H} = \mathbb{E}_\pi[\mathcal{H}_{m,n}(q) \mid s_0, a_0]$. Theorem 4.1(ii) gives, for $1 \leq r < \min(m, n)$,

$$\frac{1}{\sqrt{mn}} \min_{\operatorname{rank} A \leq r} \|\overline{H} - A\|_{\mathrm{F}} \leq \frac{2C_Q r_{\max}(1+\rho)}{1-\gamma\rho} \rho^{r-1}. \tag{16}$$

Thus mixing yields an approximately low-rank conditional expected Hankel matrix for sufficiently large admissible $r$. Approximation errors for sampled matrices additionally depend on stochastic fluctuations. The construction and proof appear in Appendix B.1.

# 5 Q-HANKELING

Theorem 4.1 motivates our choice of a recurrence penalty to encourage low Hankel rank in critic value sequences. Building on the empirical observations in § 4.2, Q-Hankeling adds this penalty to the critic loss on short trajectory windows and learns the recurrence coefficients jointly with the critic. This avoids forming Hankel matrices or computing their singular values.

## 5.1 RECURRENCE-BASED REGULARIZATION

We seek Q-functions whose value sequences along trajectories have low Hankel rank. One approach is to penalize the nuclear norm $\|H\|_* = \sum_j \sigma_j(H)$ of the corresponding Hankel matrix, a standard surrogate for rank (Fazel et al., 2001; Recht et al., 2010; Fazel et al., 2013). Direct evaluation forms the matrix and computes its singular values, typically through an $O(n^3)$ decomposition for a dense $n \times n$ matrix, adding this cost to each regularized update.

We instead use the recurrence characterization in Theorem 4.1: under its nondegeneracy assumption, approximate Hankel rank $r$ yields an approximate order-$r$ linear recurrence, and conversely, small recurrence residuals yield a rank-$r$ matrix approximation, with conditioning factors controlling the error bounds. This motivates penalizing recurrence residuals directly, avoiding Hankel-matrix construction and decomposition. For a transition sampled at time $t_b$, define the order-$r$ residual

$$\epsilon_b(\theta, c) = Q_\theta(s_{t_b}, a_{t_b}) - \sum_{i=1}^{r} c_i Q_\theta(s_{t_b-i}, a_{t_b-i}), \tag{17}$$

where $c \in \mathbb{R}^r$ is a trainable coefficient vector shared across replay windows, initialized randomly with $\mathbf{1}^\top c = 1$. Values are evaluated at the stored actions, so each window follows a trajectory generated by the behaviour policy. Among the $B$ transitions in the base algorithm's minibatch, let $\mathcal{B}_{\mathrm{H}}$ contain those whose $r$ consecutive predecessors are still available in replay and belong to the same episode. The regularizer and total critic loss are

$$\mathcal{L}_{\mathrm{Hankel}}(\theta, c) = \frac{1}{|\mathcal{B}_{\mathrm{H}}|} \sum_{b \in \mathcal{B}_{\mathrm{H}}} \mathrm{Huber}\big(\epsilon_b(\theta, c)\big), \tag{18}$$

$$\mathcal{L}(\theta, c) = \mathcal{L}_{\mathrm{base}}(\theta) + \lambda_g \mathcal{L}_{\mathrm{Hankel}}(\theta, c), \tag{19}$$

with $\mathcal{L}_{\mathrm{Hankel}}(\theta, c) = 0$ when $\mathcal{B}_{\mathrm{H}}$ is empty. Here Huber has threshold one, $\mathcal{L}_{\mathrm{base}}(\theta)$ is the base critic objective, and $\lambda_g \geq 0$ controls regularization at gradient step $g$. The base loss depends only on the critic parameters $\theta$, while the Hankel loss depends on both $\theta$ and $c$. The penalty encourages a shared approximate recurrence across windows; the base loss anchors the critic to its Bellman targets. The Huber definition and optimization details are in Appendix C.1.

## 5.2 TRAINING WITH TRAJECTORY REPLAY

The replay buffer retains episode identities and temporal order so that each sampled transition can be paired with its predecessors. Sampling follows the base algorithm's rule. Transitions without a complete window still contribute to $\mathcal{L}_{\mathrm{base}}(\theta)$, but not to $\mathcal{L}_{\mathrm{Hankel}}(\theta, c)$; no additional environment interaction is required.

All $r + 1$ values in Eq. (17) are computed with the online critic, and gradients flow through the entire window. The critic parameters and recurrence coefficients are updated jointly, with $c$ receiving gradients only from the penalty. We activate regularization after a warm-up of $K_0$ critic updates: $\lambda_g = \lambda \mathbb{I}[g \geq K_0]$, counting $g$ from the start of learning after replay prefill. Algorithm 1 summarizes the update; replay implementation details and warm-up settings appear in Appendix C.

## 5.3 INTEGRATION WITH BASELINE METHODS

Q-Hankeling augments the critic objectives of DQN, SAC, TD3, and DER while retaining their target and actor update rules. For twin critics, we combine the penalties using shared recurrence coefficients; for distributional critics, we regularize expected Q-values. Each window requires $r + 1$ critic evaluations and $O(r)$ residual arithmetic. Baseline-specific details and runtime measurements appear in Appendices C.3 and D.7.

**Algorithm 1** One critic update with Q-Hankeling

**Require:** online critic $Q_\theta$, coefficients $c \in \mathbb{R}^r$, weight $\lambda$, warm-up $K_0$, step $g$

Sample $B$ transitions using the base replay rule; compute $\mathcal{L}_{\text{base}}(\theta)$
$\mathcal{B}_{\text{H}} \leftarrow$ indices with all $r$ predecessors available in the same episode
$\mathcal{L}_{\text{Hankel}}(\theta, c) \leftarrow 0$
**if** $\lambda > 0$, $g \geq K_0$, and $\mathcal{B}_{\text{H}} \neq \varnothing$ **then**
    Fetch the valid windows and evaluate their $r+1$ values with $Q_\theta$
    Compute $\epsilon_b(\theta, c)$ and $\mathcal{L}_{\text{Hankel}}(\theta, c)$ using Eqs. (17)–(18)
**end if**
$\mathcal{L}(\theta, c) \leftarrow \mathcal{L}_{\text{base}}(\theta) + \lambda \mathbb{I}[g \geq K_0] \mathcal{L}_{\text{Hankel}}(\theta, c)$
Update $\theta$ with the base optimizer; update $c$ when the penalty is active and nonempty
Follow the base algorithm's target and actor update schedules

## 6 Experimental Results

We evaluate whether Q-Hankeling improves policy quality and sample efficiency at fixed interaction budgets across Classical Control, Atari-100k, and MuJoCo. Tables 1 and 2 summarize the main results.

**Experimental setup.** We use Stable-Baselines3 DQN, SAC, and TD3 with RL Zoo configurations (Raffin et al., 2021; Raffin, 2020), and Data-Efficient Rainbow (DER) with DrQ augmentation and an IQN head for Atari (van Hasselt et al., 2019; Yarats et al., 2021; Dabney et al., 2018). Baseline and Q-Hankeling runs use paired seeds and matched interaction and update budgets. We report final evaluation scores and steps to return thresholds, including the number of seeds reaching each threshold (see Table 10). Implementation details, configuration exceptions, tuning, evaluation protocols, and runtime are in Appendix D.

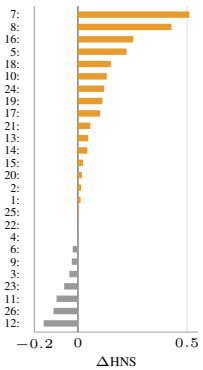
**Classical Control.** On CartPole, MountainCar, and Acrobot, the final-score gains are mixed (Table 1): Q-Hankeling improves CartPole and MountainCar by approximately 8% and 4%, respectively, but reduces Acrobot performance by 6%. On CartPole, the mean steps to return 200 fall from 17k to 5k, with all five seeds reaching the threshold in both variants. Higher thresholds do not improve consistently (Table 10), so faster early learning need not imply faster convergence to a near-optimal policy.

Table 1: Final evaluation performance. Arrows show baseline → Q-Hankeling; algorithm subscripts give $(\lambda, r)$. Score subscripts report ± standard deviation over seeds (five per comparison, except Swimmer–SAC: two). Atari reports mean HNS over 26 games, with standard deviations of the five per-seed game averages. Bold suite colors mark the better result; gray marks the other.

| Environment | Algorithm ⟶ Q-Hankeling | | | | |
|---|---|---|---|---|---|
| | $(\lambda, r)$ | Final Score Mean (± Std) | | | Change |
| Ant | SAC-QH$_{(0.1,8)}$ | 2361.3$_{(\pm 1960.0)}$ | → | **3589.4**$_{(\pm 1312.2)}$ | +52% |
| | TD3-QH$_{(1,2)}$ | 3437.2$_{(\pm 1801.8)}$ | → | **4178.6**$_{(\pm 1299.9)}$ | +22% |
| HalfCheetah | SAC-QH$_{(1,2)}$ | **10912.8**$_{(\pm 586.1)}$ | → | 10659.8$_{(\pm 511.9)}$ | -2% |
| | TD3-QH$_{(0.1,2)}$ | 10138.3$_{(\pm 794.5)}$ | → | **10987.0**$_{(\pm 473.9)}$ | +8% |
| Swimmer | SAC-QH$_{(1,8)}$ | 342.9$_{(\pm 0.9)}$ | → | **344.4**$_{(\pm 2.8)}$ | +0% |
| | TD3-QH$_{(10,8)}$ | 104.9$_{(\pm 140.9)}$ | → | **314.7**$_{(\pm 24.4)}$ | +200% |
| Humanoid | SAC-QH$_{(20,8)}$ | 4693.0$_{(\pm 703.1)}$ | → | **5091.4**$_{(\pm 423.9)}$ | +8% |
| HumanoidStandup | SAC-QH$_{(10,2)}$ | 150042.1$_{(\pm 7248.4)}$ | → | **155632.2**$_{(\pm 6612.8)}$ | +4% |
| MountainCar | DQN-QH$_{(0.1,2)}$ | -131.8$_{(\pm 15.0)}$ | → | **-126.1**$_{(\pm 42.3)}$ | +4% |
| CartPole | DQN-QH$_{(1,2)}$ | 407.0$_{(\pm 172.9)}$ | → | **440.8**$_{(\pm 132.1)}$ | +8% |
| Acrobot | DQN-QH$_{(0.5,2)}$ | **-73.6**$_{(\pm 2.4)}$ | → | -77.9$_{(\pm 2.8)}$ | -6% |
| Atari-100k | DER-QH$_{(2,8)}$ | HNS 0.463$_{(\pm 0.039)}$ | → | HNS **0.530**$_{(\pm 0.067)}$ | +14% |

Table 2: Atari-100k final performance. Entries show baseline → Q-Hankeling raw scores, with HNS in subscripts (not standard deviations). ΔHNS is Q-Hankeling minus baseline HNS. Bold orange marks the better result; gray marks the other. The right-hand plot shows sorted ΔHNS, labeled by game number (1–26); orange denotes gains and gray losses. The summary strip covers all 26 games; parentheses give changes in HNS.

| | Game | DER with Q-Hankeling Raw Scores (HNS) | ΔHNS | | Game | DER with Q-Hankeling Raw Scores (HNS) | ΔHNS |
|---|---|---|---|---|---|---|---|
| 1: | Alien | $817.3_{(0.085)}$ → **$903.4_{(0.098)}$** | +0.012 | 14: | Gopher | $406.8_{(0.069)}$ → **$498.4_{(0.112)}$** | +0.043 |
| 2: | Amidar | $125.1_{(0.070)}$ → **$148.8_{(0.083)}$** | +0.014 | 15: | Hero | $3881_{(0.096)}$ → **$4572_{(0.119)}$** | +0.023 |
| 3: | Assault | **$641.5_{(0.807)}$** → $621.5_{(0.768)}$ | -0.039 | 16: | Jamesbond | $349.1_{(1.169)}$ → **$418.4_{(1.422)}$** | +0.253 |
| 4: | Asterix | **$939.7_{(0.088)}$** → $910.6_{(0.084)}$ | -0.004 | 17: | Kangaroo | $723.8_{(0.225)}$ → **$1027_{(0.327)}$** | +0.102 |
| 5: | BankHeist | $53.9_{(0.054)}$ → **$219.0_{(0.277)}$** | +0.223 | 18: | Krull | $4012_{(2.262)}$ → **$4174_{(2.413)}$** | +0.151 |
| 6: | BattleZone | **$15638_{(0.381)}$** → $14813_{(0.358)}$ | -0.024 | 19: | KungFuMaster | $8879_{(0.384)}$ → **$11393_{(0.495)}$** | +0.112 |
| 7: | Boxing | $11.5_{(0.946)}$ → **$17.6_{(1.456)}$** | +0.510 | 20: | MsPacman | $1147_{(0.126)}$ → **$1264_{(0.144)}$** | +0.018 |
| 8: | Breakout | $15.3_{(0.473)}$ → **$27.6_{(0.900)}$** | +0.428 | 21: | Pong | $-16.6_{(0.117)}$ → **$-14.6_{(0.174)}$** | +0.057 |
| 9: | ChopperCommand | **$1430_{(0.094)}$** → $1248_{(0.066)}$ | -0.028 | 22: | PrivateEye | **$86.2_{(0.001)}$** → $21.2_{(-0.000)}$ | -0.001 |
| 10: | CrazyClimber | $19735_{(0.357)}$ → **$23046_{(0.490)}$** | +0.132 | 23: | Qbert | **$2756_{(0.195)}$** → $1924_{(0.132)}$ | -0.063 |
| 11: | DemonAttack | **$1600_{(0.796)}$** → $1423_{(0.698)}$ | -0.098 | 24: | RoadRunner | $12489_{(1.593)}$ → **$13425_{(1.712)}$** | +0.120 |
| 12: | Freeway | **$27.4_{(0.924)}$** → $22.7_{(0.767)}$ | -0.157 | 25: | Seaquest | $363.9_{(0.007)}$ → **$477.6_{(0.010)}$** | +0.003 |
| 13: | Frostbite | $1642_{(0.369)}$ → **$1843_{(0.416)}$** | +0.047 | 26: | UpNDown | **$4504_{(0.356)}$** → $3252_{(0.244)}$ | -0.112 |




| Mean HNS | Median HNS | IQM HNS | Games with better mean score |
|---|---|---|---|
| 0.463 → **0.530** (+0.066) | 0.290 → **0.302** (+0.011) | 0.277 → **0.308** (+0.031) | 9 → **17** |

**Atari-100k** With 100k interactions per game (Kaiser et al., 2020), a single setting $(\lambda, r) = (2, 8)$ selected on a five-game tuning subset improves mean human-normalized score from 0.463 to 0.530 (approximately 14%), and IQM from 0.277 to 0.308. Q-Hankeling improves the mean score on 17 of 26 games (Table 2). The largest gains occur on Boxing and Breakout, while Freeway and UpNDown deteriorate, indicating broad but nonuniform benefits. Per-game learning curves and evaluation details appear in Appendix E.4.

**MuJoCo.** At one million interactions, Q-Hankeling improves final return in seven of the eight SAC/TD3 task–algorithm combinations in Table 1. The largest gains are on Ant (52% with SAC, 22% with TD3) and Swimmer (200% with TD3); HalfCheetah–SAC declines by 2%. Sample-efficiency gains depend on the target: HalfCheetah–TD3 reaches return 8000 in 336k rather than 418k steps, with five of five seeds succeeding in both variants, whereas Humanoid–SAC takes longer to reach 5000 (Table 10). These results support recurrence regularization as a useful inductive bias, with benefits that vary across tasks and learning stages. Full threshold results and additional plots are in Appendix E; Hankel and gradient diagnostics are described in Appendix D.2.

## 7 Conclusion

We investigated the hypothesis that learned Q-values along trajectories of high-performing policies form approximately low-rank Hankel matrices. Our experiments provide empirical support for this hypothesis across Classical Control, MuJoCo, and Atari. Motivated by this evidence, Q-Hankeling encourages learned Q-functions to exhibit this Hankel structure by penalizing deviations from a linear recurrence whose coefficients are learned jointly with the critic. This promotes temporal structure without explicitly constructing or decomposing Hankel matrices. The observed gains in policy quality and sample efficiency support this approach, although benefits vary across tasks and learning stages.

The hypothesis and regularizer require evaluation beyond the current benchmark suites. Geometric mixing provides one sufficient explanation for low rank in conditional expected Hankel matrices; weaker assumptions and nonmixing settings deserve study. Our analysis does not establish low rank along individual noisy trajectories or characterize learned critics under changing policies. A broader theory, including guarantees for estimation accuracy and sample efficiency, remains open. Finally, extending recurrence regularization to actor learning through temporal structure in policy outputs or representations raises questions about its effects on policy improvement and exploration.

### AI USE STATEMENT

Generative AI tools assisted manuscript drafting and editing, the formulation and presentation of mathematical results, code writing, and figure preparation. The authors take responsibility for the final manuscript, including its mathematical claims, experimental results, code, and figures.

### ETHICS STATEMENT

The authors have considered the ethical implications of their work and have ensured that the research adheres to the relevant ethical guidelines and standards.

### REPRODUCIBILITY STATEMENT

Appendices A and B give the mathematical assumptions and proofs. Appendix C describes the objective, replay handling, baseline adaptations, and update pseudocode. Appendix D specifies benchmark configurations, seeds, evaluation protocols, rank measurements, and compute settings. Appendix E provides the supplementary results and diagnostics, including sample sizes and uncertainty conventions. Implementation code will be provided as supplementary material.

## REFERENCES

Alekh Agarwal, Sham Kakade, Akshay Krishnamurthy, and Wen Sun. Flambe: Structural complexity and representation learning of low rank mdps. *Advances in neural information processing systems*, 33:20095–20107, 2020.

Anish Agarwal, Muhammad Jehangir Amjad, Devavrat Shah, and Dennis Shen. Model agnostic time series analysis via matrix estimation. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2(3):1–39, 2018. doi: 10.1145/3287319.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems 34*, 2021.

Yu. A. Al'pin. The Hankel matrix rank theorem revisited. *Linear Algebra and its Applications*, 534: 97–101, 2017. doi: 10.1016/j.laa.2017.08.010.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013. doi: 10.1613/jair.3912. URL https://arxiv.org/abs/1207.4708.

James A. Cadzow. Signal enhancement—a composite property mapping algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(1):49–62, 1988. doi: 10.1109/29.1488.

Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural computation*, 5(4):613–624, 1993.

Simon Du, Akshay Krishnamurthy, Nan Jiang, Alekh Agarwal, Miroslav Dudik, and John Langford. Provably efficient rl with rich observations via latent state decoding. In *International Conference on Machine Learning*, pp. 1665–1674. PMLR, 2019.

Bastien Dubail, Stefan Stojanovic, and Alexandre Proutiere. Shift before you learn: Enabling low-rank representations in reinforcement learning. *Advances in Neural Information Processing Systems*, 38:72243–72305, 2025.

Maryam Fazel, Haitham Hindi, and Stephen P. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *Proceedings of the American Control Conference*, volume 6, pp. 4734–4739, Arlington, Virginia, USA, June 2001. IEEE.

Maryam Fazel, Ting Kei Pong, Defeng Sun, and Paul Tseng. Hankel matrix rank minimization with applications to system identification and realization. *SIAM Journal on Matrix Analysis and Applications*, 34(3):946–977, 2013. doi: 10.1137/110853996.

Nina Golyandina, Vladimir Nekrutkin, and Anatoly Zhigljavsky. *Analysis of Time Series Structure: SSA and Related Techniques*. Chapman & Hall/CRC, 2001.

BL Ho and Rudolf E Kálmán. Effective construction of linear state-variable models from input/output functions: Die konstruktion von linearen modeilen in der darstellung durch zustandsvariable aus den beziehungen für ein-und ausgangsgrößen. *at-Automatisierungstechnik*, 14 (1-12):545–548, 1966.

Yassir Jedra, Junghyun Lee, Alexandre Proutiere, and Se-Young Yun. Nearly optimal latent state decoding in block mdps. In *International Conference on Artificial Intelligence and Statistics*, pp. 2805–2904. PMLR, 2023.

Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual decision processes with low bellman rank are pac-learnable. In *International Conference on Machine Learning*, pp. 1704–1713. PMLR, 2017.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on learning theory*, pp. 2137–2143. PMLR, 2020.

Łukasz Kaiser, Mohammad Babaeizadeh, Piotr Miłos, Błażej Osiński, Roy H. Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model-based reinforcement learning for Atari. In *International Conference on Learning Representations*, 2020.

Boris Kramer and Alex A. Gorodetsky. System identification via CUR-factored Hankel approximation. *SIAM Journal on Scientific Computing*, 40(2):A848–A866, 2018. doi: 10.1137/17M1137632.

Leopold Kronecker. *Zur Theorie der Elimination einer Variabeln aus zwei algebraischen Gleichungen*. Königliche Akad. der Wissenschaften, 1881.

Cassidy Laidlaw, Stuart J Russell, and Anca Dragan. Bridging rl theory and practice with the effective horizon. *Advances in Neural Information Processing Systems*, 36:58953–59007, 2023.

David A. Levin and Yuval Peres. *Markov Chains and Mixing Times*. American Mathematical Society, 2 edition, 2017. URL `https://pages.uoregon.edu/dlevin/MARKOV/mcmt2e.pdf`.

Qiyang Li, Zhiyuan Paul Zhou, and Sergey Levine. Reinforcement learning with action chunking. *Advances in Neural Information Processing Systems*, 38:55518–55553, 2025.

Jungseul Ok, Alexandre Proutiere, and Damianos Tranos. Exploration in structured reinforcement learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, 2019.

Vladimir V. Peller. An excursion into the theory of Hankel operators. In *Holomorphic Spaces*, volume 33 of *Mathematical Sciences Research Institute Publications*, pp. 65–120. Cambridge University Press, 1998. URL `https://library.slmath.org/books/Book33/files/peller.pdf`.

Vladimir V. Peller. *Hankel operators and their applications*, volume 15. Springer, 2003.

Antonin Raffin. RL Baselines3 Zoo. `https://github.com/DLR-RM/rl-baselines3-zoo`, 2020. GitHub repository.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010. doi: 10.1137/070697835.

Tyler Sam, Yudong Chen, and Christina Lee Yu. Overcoming the long horizon barrier for sample-efficient reinforcement learning with latent low-rank structure. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 7(2):1–60, 2023.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

Paul Schwerdtner, Jules Berman, and Benjamin Peherstorfer. Hankel singular value regularization for highly compressible state space models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 134344–134369. Curran Associates, Inc., 2025. doi: 10.52202/085713-4481. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/c3070c3388552a08a3326f0d28dc2af9-Paper-Conference.pdf.

Younggyo Seo and Pieter Abbeel. Coarse-to-fine q-network with action sequence for data-efficient reinforcement learning. *Advances in Neural Information Processing Systems*, 38:109915–109940, 2025.

Devavrat Shah, Dogyoon Song, Zhi Xu, and Yuzhe Yang. Sample efficient reinforcement learning via low-rank matrix estimation. *Advances in Neural Information Processing Systems*, 33:12092–12103, 2020.

Stefan Stojanovic, Yassir Jedra, and Alexandre Proutiere. Model-free low-rank reinforcement learning via leveraged entry-wise matrix estimation. *Advances in Neural Information Processing Systems*, 37:30886–30924, 2024.

Wen Sun, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Model-based rl in contextual decision processes: Pac bounds and exponential improvements over model-free approaches. In *Conference on learning theory*, pp. 2898–2933. PMLR, 2019.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Dong Tian, Onur Celik, and Gerhard Neumann. Chunking the critic: A transformer-based soft actor-critic with n-step returns. In *International Conference on Learning Representations*, 2026.

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

Ahmed Touati, Jérémy Rapin, and Yann Ollivier. Does zero-shot reinforcement learning exist? *arXiv preprint arXiv:2209.14935*, 2022.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A standard interface for reinforcement learning environments. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2025. URL https://arxiv.org/abs/2407.17032.

Hado van Hasselt, Matteo Hessel, and John Aslanides. When to use parametric models in reinforcement learning? In *Advances in Neural Information Processing Systems 32*, 2019.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, 2016.

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=GY6-6sTvGaf.

Thomas T Zhang, Daniel Pfrommer, Chaoyi Pan, Nikolai Matni, and Max Simchowitz. Action chunking and data augmentation yield exponential improvements in behavior cloning for continuous spaces. In *International Conference on Learning Representations*, 2026.

Dongruo Zhou, Quanquan Gu, and Csaba Szepesvari. Nearly minimax optimal reinforcement learning for linear mixture markov decision processes. In *Conference on Learning Theory*, pp. 4532–4576. PMLR, 2021.

# A HANKEL RANK AND RECURRENCES

The forward direction uses a CUR (skeleton) approximation based on the first $r$ columns and a selected set of $r$ rows. Related CUR-based Hankel approximations are used in system identification (Kramer & Gorodetsky, 2018); we give the recurrence error argument in full.

*Proof of Theorem 4.1. Part (i).* Set $C = H_{:,1:r}$. Since $C$ has full column rank, there exists a row set $I \subseteq \{1, \dots, m\}$, $|I| = r$, such that $B = C_{I,:} = H_{I,1:r}$ is nonsingular. Choose $I$ to maximize $|\det B|$, breaking ties lexicographically, and order its indices increasingly. Define

$$R = H_{I,:}, \qquad X = B^{-1}R, \qquad E = H - CX,$$

and set

$$M = \max_{1 \le j \le n} \|X_{:,j}\|_1, \qquad \kappa_B = \frac{\prod_{i=1}^{r} \sigma_i(H)}{|\det B|}, \qquad K_r(H) = (1 + 2M)\kappa_B. \tag{20}$$

These quantities are finite because $B$ is nonsingular. Write $\delta = \|E\|_{\max} := \max_{i,j} |E_{ij}|$. Since $C_{I,:} = B$ and $X_{:,1:r} = I_r$, the rows of $E$ indexed by $I$ and its first $r$ columns vanish. Define $c_i = X_{r+1-i,r+1}$ and $\epsilon_t = q_t - \sum_{i=1}^{r} c_i q_{t-i}$ for $r < t \le T$. Clearly $\|c\|_1 = \|X_{:,r+1}\|_1 \le M$.

*Step 1: transfer the CUR error to a shared recurrence.* The Hankel structure gives

$$E_{s,r+1} = q_{s+r} - \sum_{k=1}^{r} q_{s+k-1} X_{k,r+1} = \epsilon_{s+r}, \qquad 1 \le s \le m. \tag{21}$$

Define the row-recurrence operator $L_c \in \mathbb{R}^{(m-r)\times m}$ by $(L_c Z)_{u,:} = Z_{u+r,:} - \sum_{i=1}^{r} c_i Z_{u+r-i,:}$. For $1 \le u \le m - r$ and $1 \le k \le r$,

$$(L_c C)_{u,k} = \epsilon_{u+r+k-1} = E_{u+k-1,r+1}.$$

Consequently, $\|L_c C\|_{\max} \le \delta$. Using $L_c H = (L_c C)X + L_c E$, we obtain

$$\begin{aligned} |(L_c H)_{u,j}| &\le \delta \|X_{:,j}\|_1 + (1 + \|c\|_1)\delta \\ &\le (1 + 2M)\delta. \end{aligned}$$

But $(L_c H)_{u,j} = \epsilon_{u+r+j-1}$, and these indices cover every $t \in \{r+1, \dots, T\}$. Thus

$$\max_{r < t \le T} |\epsilon_t| \le (1 + 2M)\|H - CB^{-1}R\|_{\max}. \tag{22}$$

*Step 2: bound the CUR error by a neglected singular value.* For $i \notin I$ and $j > r$, form the bordered submatrix

$$D_{ij} = \begin{pmatrix} B & H_{I,j} \\ H_{i,1:r} & H_{ij} \end{pmatrix}.$$

The Schur complement identity gives $\det D_{ij} = (\det B)E_{ij}$. Since selecting rows and columns cannot increase any singular value, $\sigma_k(D_{ij}) \le \sigma_k(H)$ for $1 \le k \le r + 1$. Therefore

$$|E_{ij}| = \frac{\prod_{k=1}^{r+1} \sigma_k(D_{ij})}{|\det B|} \le \frac{\prod_{k=1}^{r} \sigma_k(H)}{|\det B|} \sigma_{r+1}(H) = \kappa_B \sigma_{r+1}(H).$$

The remaining entries of $E$ are zero, so combining this bound with Eq. (22) and the definition of $K_r(H)$ proves part (i).

*Part (ii).* Fix any $c \in \mathbb{R}^r$, and define $D_c \in \mathbb{R}^{(m-r)\times m}$ by

$$(D_c z)_u = z_{u+r} - \sum_{i=1}^{r} c_i z_{u+r-i}, \qquad 1 \le u \le m - r.$$

Its submatrix in columns $r + 1, \dots, m$ is lower triangular with unit diagonal. Hence $D_c$ has full row rank and $\dim \ker D_c = r$. Let $P_c$ be the orthogonal projection onto $\ker D_c$, and set $A = P_c H$. Then $\operatorname{rank} A \le r$. Writing $D_c^{\dagger}$ for the Moore–Penrose pseudoinverse, we have $I - P_c = D_c^{\dagger} D_c$ and therefore

$$\|H - A\|_{\mathrm{F}} = \|D_c^{\dagger} D_c H\|_{\mathrm{F}} \le \frac{\|D_c H\|_{\mathrm{F}}}{\sigma_{\min}(D_c)}.$$

The Hankel structure gives $(D_c H)_{u,j} = \epsilon_{u+r+j-1}$, so

$$\|D_c H\|_{\mathrm{F}} \le \sqrt{(m-r)n}\, \max_{r<t\le T} |\epsilon_t|.$$

Consequently, part (ii) holds with the finite factor

$$J_{m,n,r}(c) = \frac{\sqrt{(m-r)n}}{\sigma_{\min}(D_c)}. \tag{23}$$

Here $\sigma_{\min}(D_c)$ is the smallest of its $m-r$ positive singular values. The approximation $A$ need not itself be Hankel; its rank bound establishes the claimed matrix approximation error. □

**Approximate rank and conditioning.** If $\|H-A\|_{\mathrm{F}} \le \varepsilon \|H\|_{\mathrm{F}}$ for some matrix $A$ of rank at most $r$, then $\sigma_{r+1}(H) \le \|H-A\|_2 \le \varepsilon \|H\|_{\mathrm{F}}$. Substitution yields the relative-error version stated in the main text. The quantities $M$ and $\kappa_B$ depend on the selected block; this result does not assert that they are uniformly bounded for all approximately low-rank Hankel matrices. Likewise, the converse factor $J_{m,n,r}(c)$ can grow with the dimensions or change with the coefficients as $\sigma_{\min}(D_c)$ becomes small.

**Why a nondegeneracy condition is necessary.** Take $q = (0,0,1)$. Then

$$\mathcal{H}_{2,2}(q) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

has rank one, but $q_3 - c_1 q_2 = 1$ for every $c_1$. Its first column is zero, so the full-column-rank assumption fails. Thus finite Hankel rank alone does not ensure a forward recurrence of the same order.

**Recurrence error from the AAK approximation.** For the infinite-operator statement in § 4.1, AAK gives $\|\mathcal{H}_\infty(q) - \mathcal{H}_\infty(\bar{q})\|_{\mathrm{op}} = \sigma_{r+1}(\mathcal{H}_\infty(q))$. Applying the operator difference to the first coordinate vector bounds $d = q - \bar{q}$ in $\ell^2$ by the same quantity. Let $c$ be the recurrence coefficients of $\bar{q}$, padded with zeros if its order is less than $r$. Then $\epsilon_t = d_t - \sum_{i=1}^{r} c_i d_{t-i}$ for $t > r$. The triangle inequality on these shifted tails yields $\|\epsilon\|_{\ell^2} \le (1 + \|c\|_1)\|d\|_{\ell^2}$, which proves the stated bound.

## B Mixing and Value Recurrences

We give the mixing lemma, supporting propositions, and proofs for § 4.3.

Fix a stationary policy $\pi$, and let $P^\pi(s' \mid s) = \mathbb{E}_{a\sim\pi(\cdot|s)}[p(s' \mid s,a)]$ denote its induced state transition kernel. We use the following standard finite-state convergence result (Levin & Peres, 2017, Theorem 4.9).

**Lemma B.1** (Convergence to stationarity)**.** *Suppose the state space $\mathcal{S}$ is finite and $P^\pi$ is irreducible and aperiodic. Then it has a unique stationary distribution $\nu^\pi$, satisfying $\nu^\pi P^\pi = \nu^\pi$, and there exist constants $C > 0$ and $\rho \in (0,1)$ such that, for every integer $t \ge 0$,*

$$\sup_{s\in\mathcal{S}} \left\| (P^\pi)^t(s,\cdot) - \nu^\pi \right\|_{\mathrm{TV}} \le C\rho^t. \tag{24}$$

*Here $\|\mu-\nu\|_{\mathrm{TV}} = \frac{1}{2}\sum_{s\in\mathcal{S}} |\mu(s) - \nu(s)|$.*

Write $P = P^\pi$, $\nu = \nu^\pi$, and let $R(s) = R^\pi(s)$ be the expected immediate reward. For $0 \le \gamma < 1$, the Bellman equation gives $V = V^\pi = (I - \gamma P)^{-1} R$. Define the conditional mean sequence

$$M_j(s) := \mathbb{E}_\pi[V(s_j) \mid s_0 = s] = (P^j V)(s), \qquad j \ge 0, \tag{25}$$

and write $\bar{R} = \nu R$. Recall that $|R(s)| \le r_{\max}$.

**Proposition B.2** (Value recurrence with mixing-controlled bias)**.** *Under the assumptions of Lemma B.1, fix an integer $r \ge 1$ and real coefficients $c_1, \dots, c_r$. Define*

$$b_r(s) := M_r(s) - \sum_{i=1}^{r} c_i M_{r-i}(s). \tag{26}$$

*Along a trajectory generated by $\pi$, for every $t \geq 0$,*

$$V(s_{t+r}) = \sum_{i=1}^{r} c_i V(s_{t+r-i}) + b_r(s_t) + \xi_{t,r}, \tag{27}$$

*where*

$$\xi_{t,r} := V(s_{t+r}) - M_r(s_t) - \sum_{i=1}^{r} c_i \big[V(s_{t+r-i}) - M_{r-i}(s_t)\big], \qquad \mathbb{E}_\pi[\xi_{t,r} \mid s_t] = 0. \tag{28}$$

*The bias satisfies*

$$\left\| b_r - \left(1 - \sum_{i=1}^{r} c_i\right) \frac{\bar{R}}{1-\gamma} \mathbf{1} \right\|_\infty \leq \frac{2Cr_{\max}}{1-\gamma\rho} \left(\rho^r + \sum_{i=1}^{r} |c_i| \rho^{r-i}\right). \tag{29}$$

*No reversibility assumption is required.*

*Proof.* The Markov property gives $\mathbb{E}_\pi[V(s_{t+j}) \mid s_t] = M_j(s_t)$. Adding and subtracting these conditional means proves Eqs. (27) and (28). For any $j \geq 0$, the total-variation bound in Lemma B.1 implies

$$\|P^j R - \bar{R}\mathbf{1}\|_\infty \leq 2Cr_{\max}\rho^j.$$

The convergent Bellman series yields $M_j = \sum_{\ell=0}^{\infty} \gamma^\ell P^{j+\ell} R$, and hence

$$\left\| M_j - \frac{\bar{R}}{1-\gamma}\mathbf{1} \right\|_\infty \leq 2Cr_{\max} \sum_{\ell=0}^{\infty} \gamma^\ell \rho^{j+\ell} = \frac{2Cr_{\max}}{1-\gamma\rho}\rho^j. \tag{30}$$

Substituting this bound into Eq. (26) and applying the triangle inequality proves Eq. (29). □

When $\sum_{i=1}^{r} c_i = 1$, the stationary term vanishes. For example, $c_1 = 1$ and $c_i = 0$ for $i \geq 2$ give

$$\|b_r\|_\infty \leq \frac{2Cr_{\max}(1+\rho)}{1-\gamma\rho}\rho^{r-1}. \tag{31}$$

This establishes the existence of a recurrence with exponentially small bias conditional on the start of the length-$r$ trajectory window. For general coefficients, decay requires control of both the stationary term and the weighted sum $\sum_{i=1}^{r} |c_i|\rho^{r-i}$; it does not follow merely by increasing $r$.

For the conditional mean sequence itself, Eq. (30) also gives

$$\left\| M_{t+r} - \sum_{i=1}^{r} c_i M_{t+r-i} - \left(1 - \sum_{i=1}^{r} c_i\right) \frac{\bar{R}}{1-\gamma}\mathbf{1} \right\|_\infty \leq \frac{2Cr_{\max}}{1-\gamma\rho}\rho^t \left(\rho^r + \sum_{i=1}^{r} |c_i|\rho^{r-i}\right). \tag{32}$$

These bounds control the conditional bias, not the magnitude or variance of $\xi_{t,r}$. In particular, in the example above the sampled residual is $V(s_{t+r}) - V(s_{t+r-1})$, which can remain large despite a small conditional bias. Mixing therefore explains approximate recurrences in conditional mean values; additional control of the stochastic residual is needed to explain low Hankel rank along individual trajectories.

**Choosing the recurrence coefficients.** The choice $(1, 0, \ldots, 0)$ establishes an upper bound on the achievable bias; it need not minimise that bias. To allow the coefficients to capture the value dynamics, write $b_r(c)$ for the bias in Eq. (26) and define $\|f\|_\nu^2 := \sum_{s\in\mathcal{S}} \nu(s) f(s)^2$. For this analysis, impose $\mathbf{1}^\top c = 1$ so that the recurrence preserves constant sequences.

**Proposition B.3** (Optimal conditional-mean recurrence)**.** *Under the assumptions of Proposition B.2, for every $r \geq 1$ the optimisation problem*

$$c^\star \in \operatorname*{arg\,min}_{c\in\mathbb{R}^r:\, \mathbf{1}^\top c = 1} \left\| P^r V - \sum_{i=1}^{r} c_i P^{r-i} V \right\|_\nu^2 \tag{33}$$

*has a solution, and every solution satisfies*

$$\|b_r(c^\star)\|_\nu \le \frac{2Cr_{\max}(1+\rho)}{1-\gamma\rho}\rho^{r-1}. \tag{34}$$

*The predicted vector $\sum_{i=1}^r c_i^\star P^{r-i}V$ is unique, although its coefficient representation need not be. The optimal bias is zero if and only if $P^rV$ lies in the affine hull of $V, PV, \dots, P^{r-1}V$.*

*If $s_t \sim \nu$, define*

$$W(c) := \sum_{j=0}^{r-1}\left|\sum_{i=j+1}^{r} c_i\right|. \tag{35}$$

*For the same optimiser $c^\star$, the stochastic residual in Eq. (28) satisfies*

$$\mathbb{E}_\pi[\xi_{t,r}(c^\star)^2] \le W(c^\star)^2\left(\mathbb{E}_\pi[\mathrm{Var}_\pi(V(s_{t+1}) \mid s_t)] + \|PV - V\|_\nu^2\right). \tag{36}$$

*This variance bound depends on the optimiser through $W(c^\star)$; conditional-bias optimality alone does not bound this quantity.*

*Proof.* The feasible predicted vectors form the nonempty affine subspace $\{\sum_{i=1}^r c_i P^{r-i}V : \mathbf{1}^\top c = 1\}$. Since $\nu(s) > 0$ for every state, projection onto this finite-dimensional closed affine subspace in the $\nu$-weighted inner product exists and is unique. This proves existence, uniqueness of the predicted vector, and the characterisation of zero bias. The feasible choice $c = (1, 0, \dots, 0)$, together with Eq. (31) and $\|f\|_\nu \le \|f\|_\infty$, proves Eq. (34). Finally, every optimiser has coefficients summing to one, so the general variance bound in Proposition B.4 applies to $c^\star$ and proves Eq. (36). □

This is a constrained least-squares problem. With $A = [P^{r-1}V, \dots, V]$, $y = P^rV$, and $D_\nu = \mathrm{diag}(\nu)$, its solutions are characterised by

$$A^\top D_\nu(Ac^\star - y) + \eta\mathbf{1} = 0, \qquad \mathbf{1}^\top c^\star = 1, \tag{37}$$

for some scalar multiplier $\eta$. Thus the coefficients depend on the value dynamics; the comparison with $(1, 0, \dots, 0)$ does not identify the optimiser or require any coefficient to be nonzero.

For example, suppose $V = a\mathbf{1} + u$, where $u \neq 0$ and $Pu = \lambda u$ for $0 < \lambda < 1$. At order $r = 2$, the feasible coefficients $(1+\lambda, -\lambda)$ give

$$P^2V - (1+\lambda)PV + \lambda V = 0, \tag{38}$$

whereas $(1, 0)$ leaves bias $\lambda(\lambda - 1)u \neq 0$. The optimiser therefore captures the nonconstant mode and strictly improves on the single-lag choice. This example does not require reversibility, and zero conditional bias does not imply zero sampled recurrence error.

**Conditional bias and stochastic fluctuations.** For fixed coefficients, define the sampled residual $\epsilon_{t,r}(c) := V(s_{t+r}) - \sum_{i=1}^r c_i V(s_{t+r-i})$. Proposition B.2 gives

$$b_r(c)(s_t) = \mathbb{E}_\pi[\epsilon_{t,r}(c) \mid s_t], \qquad \xi_{t,r}(c) = \epsilon_{t,r}(c) - b_r(c)(s_t). \tag{39}$$

The bias is the conditional approximation error of the recurrence; it does not arise from the conditionally mean-zero fluctuation $\xi_{t,r}(c)$. If $s_t \sim \nu$, their orthogonality gives the exact decomposition

$$\mathbb{E}_\pi[\epsilon_{t,r}(c)^2] = \|b_r(c)\|_\nu^2 + \mathbb{E}_\pi[\xi_{t,r}(c)^2]. \tag{40}$$

Indeed, the cross term vanishes after conditioning on $s_t$. Eq. (33) minimises the first term only. A population loss based on sampled recurrence residuals also includes the second term, which depends on the coefficients, so its minimiser need not coincide with $c^\star$. In particular, the exponential bound on the optimal conditional bias does not by itself bound the sampled recurrence loss.

**Proposition B.4** (General variance bound)**.** *Suppose $s_t \sim \nu$, where $\nu$ is stationary for $P$, and fix $c \in \mathbb{R}^r$ with $\sum_{i=1}^r c_i = 1$. Let $W(c)$ be the cumulative coefficient weight defined in Eq. (35). Then the stochastic residual in Eq. (39) satisfies*

$$\begin{aligned}\mathbb{E}_\pi[\xi_{t,r}(c)^2] &\le \mathbb{E}_\pi[\epsilon_{t,r}(c)^2] \\ &\le W(c)^2\left(\mathbb{E}_\pi[\mathrm{Var}_\pi(V(s_{t+1}) \mid s_t)] + \|PV - V\|_\nu^2\right).\end{aligned} \tag{41}$$

*This bound requires neither reversibility nor geometric mixing and allows signed coefficients.*

*Proof.* Since the coefficients sum to one, telescoping gives

$$\epsilon_{t,r}(c) = \sum_{j=0}^{r-1} \left( \sum_{i=j+1}^{r} c_i \right) \left[ V(s_{t+r-j}) - V(s_{t+r-j-1}) \right]. \tag{42}$$

Conditioning on $s_t$ gives

$$\mathbb{E}_\pi \left[ (V(s_{t+1}) - V(s_t))^2 \right] = \mathbb{E}_\pi [\mathrm{Var}_\pi(V(s_{t+1}) \mid s_t)] + \|PV - V\|_\nu^2. \tag{43}$$

By stationarity, each value increment in Eq. (42) has this same second moment. The triangle inequality in $L^2$ therefore yields

$$\left( \mathbb{E}_\pi[\epsilon_{t,r}(c)^2] \right)^{1/2} \le W(c) \left( \mathbb{E}_\pi [\mathrm{Var}_\pi(V(s_{t+1}) \mid s_t)] + \|PV - V\|_\nu^2 \right)^{1/2}.$$

Squaring proves the second inequality. The first follows from the nonnegative squared-bias term in Eq. (40). □

For nonnegative coefficients, $W(c) = \sum_{i=1}^{r} i\, c_i$ is the average lag of the recurrence. For signed coefficients, the cumulative sums retain cancellations between weights. Thus the variance is controlled when the one-step conditional variance and predictable change $\|PV - V\|_\nu^2$ are small and $W(c)$ remains moderate. The predictable-change term is needed because conditional variance alone does not equal the mean squared value increment. The bound applies to the optimised coefficients $c^\star$ as well, but their conditional-bias optimality alone does not control $W(c^\star)$. No decay of this variance bound with recurrence order follows from mixing alone.

**Corollary B.5** (Decaying bias and uniformly bounded variance)**.** *Under the assumptions of Proposition B.3, suppose $s_t \sim \nu$ and fix a finite $B \ge 1$, independent of $r$. For every $r \ge 1$, the constrained problem*

$$c_B^\star \in \operatorname*{arg\,min}_{\substack{c \in \mathbb{R}^r : \mathbf{1}^\top c = 1 \\ \|c\|_1 \le B}} \|b_r(c)\|_\nu^2 \tag{44}$$

*has a solution. Every solution satisfies*

$$\|b_r(c_B^\star)\|_\nu \le \frac{2C r_{\max}(1+\rho)}{1-\gamma\rho} \rho^{r-1}, \tag{45}$$

$$\mathbb{E}_\pi[\xi_{t,r}(c_B^\star)^2] \le (1+B)^2 \,\mathrm{Var}_\nu(V), \tag{46}$$

*where $\mathrm{Var}_\nu(V) = \nu(V^2) - (\nu V)^2$. Thus the conditional bias decays exponentially while the stochastic variance is bounded uniformly in the recurrence order.*

*Proof.* The feasible set is compact and contains $(1, 0, \ldots, 0)$, so continuity of the objective gives existence. Comparing with this feasible choice proves Eq. (45), as in Proposition B.3.

For any fixed coefficients $c$, the term $c_r V(s_t)$ is known when conditioning on $s_t$. Consequently,

$$\begin{aligned}
\mathbb{E}_\pi[\xi_{t,r}(c)^2] &= \mathbb{E}_\pi \left[ \mathrm{Var}_\pi \left( V(s_{t+r}) - \sum_{i=1}^{r-1} c_i V(s_{t+r-i}) \,\middle|\, s_t \right) \right] \\
&\le \mathrm{Var}_\pi \left( V(s_{t+r}) - \sum_{i=1}^{r-1} c_i V(s_{t+r-i}) \right) \\
&\le \left( 1 + \sum_{i=1}^{r-1} |c_i| \right)^2 \mathrm{Var}_\nu(V).
\end{aligned}$$

The first inequality is the law of total variance. The second follows by centring each value by $\nu V$, applying the triangle inequality in $L^2$, and using stationarity. Empty sums are zero when $r = 1$. Substituting $c = c_B^\star$ and using $\|c_B^\star\|_1 \le B$ proves Eq. (46). □

The coefficient constraint ensures that increasing the recurrence order does not amplify the noise without bound. It does not assert that the noise vanishes, and the unconstrained optimiser in Proposition B.3 need not satisfy this additional constraint.

**Proof of Theorem 4.2.** Proposition B.2 establishes the recurrence decomposition and its conditional bias. Proposition B.3 guarantees that an optimizer exists and proves the bias bound in Eq. (14). Applying Proposition B.4 to this same optimizer and using Eq. (43) gives Eq. (15). No reversibility assumption is used. □

### B.1 State-action values and expected Hankel rank

For finite state and action spaces, the state-action process $(s_t, a_t)$ under $\pi$ has transition kernel

$$K^\pi((s,a),(s',a')) = p(s' \mid s,a)\pi(a' \mid s'), \qquad \mu^\pi(s,a) = \nu^\pi(s)\pi(a \mid s). \tag{47}$$

The distribution $\mu^\pi$ is stationary. For $j \geq 1$, the state marginal after $j$ transitions from $(s,a)$ is $p(\cdot \mid s,a)(P^\pi)^{j-1}$; attaching the conditional action distribution $\pi$ preserves total-variation distance. Consequently,

$$\sup_{s,a} \|(K^\pi)^j((s,a),\cdot) - \mu^\pi\|_{\mathrm{TV}} \leq C_Q \rho^j, \qquad j \geq 0, \qquad C_Q = \max\{1, C/\rho\}. \tag{48}$$

Writing $Q = Q^\pi$ and $K = K^\pi$, the Bellman equation is $Q = r + \gamma K Q$, where $|r(s,a)| \leq r_{\max}$. The preceding recurrence analysis therefore applies with $(P, V, R, \nu, C)$ replaced by $(K, Q, r, \mu^\pi, C_Q)$. For stationary statements, we restrict the chain to the support of $\mu^\pi$. The same conditional-bias optimization and noise bounds then hold for $Q^\pi$.

**Proposition B.6** (Conditional-mean Hankel approximation)**.** *Under the above assumptions, condition on an initial state-action pair $(s_0, a_0)$, let $q_t = Q^\pi(s_{t-1}, a_{t-1})$ for $1 \leq t \leq T = m + n - 1$, and set $\overline{H} = \mathbb{E}_\pi[\mathcal{H}_{m,n}(q) \mid s_0, a_0]$. For every $1 \leq r < \min(m,n)$, the bound in Eq. (16) holds. The approximating rank-$r$ matrix may itself be chosen Hankel.*

*Proof.* Let $\bar{q}_t = (K^{t-1}Q)(s_0,a_0)$ and $\bar{Q} = \mu^\pi Q = (\mu^\pi r)/(1-\gamma)$. Applying the Bellman-series argument of Eq. (30) to $K$ gives

$$|\bar{q}_t - \bar{Q}| \leq \frac{2C_Q r_{\max}}{1-\gamma\rho}\rho^{t-1}. \tag{49}$$

Define $\widehat{q}_t = \bar{q}_t$ for $t \leq r$ and $\widehat{q}_t = \bar{q}_r$ for $t > r$. This sequence satisfies the exact order-$r$ recurrence with $c = (1, 0, \ldots, 0)$. Theorem 4.1(ii), applied with zero residual, implies $\operatorname{rank} \mathcal{H}_{m,n}(\widehat{q}) \leq r$. For $t > r$, Eq. (49) yields

$$|\bar{q}_t - \widehat{q}_t| \leq \frac{2C_Q r_{\max}}{1-\gamma\rho}(\rho^{t-1} + \rho^{r-1}) \leq \frac{2C_Q r_{\max}(1+\rho)}{1-\gamma\rho}\rho^{r-1}.$$

The first $r$ entries agree exactly. Since $\overline{H} = \mathcal{H}_{m,n}(\bar{q})$, bounding each of its $mn$ entrywise errors proves Eq. (16). □

**Expected matrix versus expected approximation error.** Write $H = \mathcal{H}_{m,n}(q)$ and $\widehat{H} = \mathcal{H}_{m,n}(\widehat{q})$. Conditional on $(s_0,a_0)$, $\widehat{H}$ is deterministic and $\mathbb{E}_\pi[H - \overline{H} \mid s_0, a_0] = 0$. Thus

$$\begin{aligned}\mathbb{E}_\pi\left[\min_{\operatorname{rank} A \leq r} \|H - A\|_{\mathrm{F}}^2 \,\middle|\, s_0, a_0\right] &\leq \|\overline{H} - \widehat{H}\|_{\mathrm{F}}^2 \\ &\quad + \mathbb{E}_\pi[\|H - \overline{H}\|_{\mathrm{F}}^2 \mid s_0, a_0].\end{aligned} \tag{50}$$

Mixing controls the first term through Eq. (16); it does not force the second term to vanish. Consequently, approximate low rank of the expected Hankel matrix alone does not establish small expected approximation error for individual trajectories. If the chain starts in stationarity and we average without conditioning on $(s_0,a_0)$, the mean sequence is constant and its Hankel matrix already has rank at most one.

# C Q-HANKELING IMPLEMENTATION DETAILS

## C.1 OBJECTIVE AND COEFFICIENT OPTIMIZATION

The penalty in Eq. (18) uses the unit-threshold Huber loss

$$\mathrm{Huber}(\epsilon) = \begin{cases} \frac{1}{2}\epsilon^2, & |\epsilon| \le 1, \\ |\epsilon| - \frac{1}{2}, & |\epsilon| > 1. \end{cases} \tag{51}$$

The coefficient vector is initialized randomly with $\mathbf{1}^\top c = 1$ and is shared across replay windows. Its optimizer group has a separate learning rate and no weight decay. The critic receives gradients from both loss terms, while $c$ receives only the recurrence gradient:

$$\begin{aligned} \nabla_\theta \mathcal{L}(\theta, c) &= \nabla_\theta \mathcal{L}_{\mathrm{base}}(\theta) + \lambda_g \nabla_\theta \mathcal{L}_{\mathrm{Hankel}}(\theta, c), \\ \nabla_c \mathcal{L}(\theta, c) &= \lambda_g \nabla_c \mathcal{L}_{\mathrm{Hankel}}(\theta, c). \end{aligned} \tag{52}$$

Both the current and lagged values are evaluated with the online critic, and gradients propagate through every evaluation. For DQN, the coefficients are excluded from the gradient-norm clipping applied to the critic parameters.

**Alternative gradient paths.** Using the target network for lagged values or detaching those values from the gradient are possible alternatives. Preliminary comparisons found poorer performance with target-network lags and similar performance with detached lags on most tested tasks. The reported method differentiates through all online-critic values; a systematic comparison of these variants remains future work.

## C.2 REPLAY WINDOWS AND EPISODE BOUNDARIES

Replay stores episode identities and temporal order alongside the transitions. Its capacity is measured in transitions, and minibatch sampling follows the base algorithm's rule. After sampling the base minibatch, the implementation retrieves each transition's $r$ predecessors. A window is valid only when every predecessor is still stored, temporally consecutive, and in the same episode. Invalid windows contribute no recurrence penalty; their endpoint transitions remain in the base critic update.

For a ring buffer, predecessors are located by stepping backward through the stored indices, checking episode identities and whether any required transition has been evicted. For episode-organized replay, the predecessors are read from the corresponding episode. Complete episodes are stored together, and the oldest episodes are evicted until the transition-count capacity is met. Uniform transition sampling draws a global transition index and maps it to an episode and position, so longer episodes are sampled in proportion to their length. These checks enforce the definition of $\mathcal{B}_{\mathrm{H}}$ in § 5.1; the condition $t_b \ge r$ alone is insufficient after replay eviction.

## C.3 BASELINE-SPECIFIC ADAPTATIONS

**DQN.** The recurrence term is added to the Huber TD objective before the optimizer step. The base target construction, target-network schedule, and replay sampling rule are retained. With $\lambda = 0$, the augmented objective reduces to the baseline objective; the implementation checks this equivalence with unit tests.

**SAC and TD3.** For online critics $Q_{\theta_1}$ and $Q_{\theta_2}$, define $\epsilon_{b,k} = Q_{\theta_k}(s_{t_b}, a_{t_b}) - \sum_{i=1}^{r} c_i Q_{\theta_k}(s_{t_b-i}, a_{t_b-i})$. Writing $\theta = (\theta_1, \theta_2)$, the coefficient vector $c$ is shared, and the recurrence loss is

$$\mathcal{L}_{\mathrm{Hankel}}(\theta, c) = \frac{\kappa}{|\mathcal{B}_{\mathrm{H}}|} \sum_{k=1}^{2} \sum_{b \in \mathcal{B}_{\mathrm{H}}} \mathrm{Huber}(\epsilon_{b,k}), \tag{53}$$

with value zero for an empty valid set. The scale matches the base critic objective: $\kappa = 1/2$ for SAC and $\kappa = 1$ for TD3. The penalty is added with weight $\lambda_g$. The actor objectives retain their baseline forms and depend on the regularized critics through $\min_k Q_{\theta_k}(s, \widetilde{a})$ in SAC and $Q_{\theta_1}(s, \pi_\phi(s))$ in

TD3. The recurrence uses stored behaviour-policy actions, rather than actions recomputed by the current actor.

For SAC, the recurrence is applied directly to the soft critic output. The entropy term remains in the base target, whose discounted next-action contribution can be written as $\widetilde{r}_t = r_t - \gamma\alpha \log \pi_\phi(a_{t+1} \mid s_{t+1})$, with the next action sampled by the target computation. Both the policy and temperature can change during training. The penalty therefore serves as a learned structural regularizer on these critic outputs; no exact recurrence along stored trajectories is assumed.

**Efficient Rainbow.** The Atari baseline combines Data-Efficient Rainbow (van Hasselt et al., 2019), the DrQ configuration (Yarats et al., 2021), and an IQN head (Dabney et al., 2018). Transitions are retained one step at a time so that recurrence windows remain consecutive; the $n$-step return is assembled separately for the base target. For the recurrence evaluations, scalar Q-values are means of the quantile outputs at a fixed midpoint grid $\tau_i = (2i-1)/(2N)$, $i = 1, \dots, N$. The same fraction grid is used for every value in the window. A random image transformation is sampled per window and shared across its $r+1$ frame stacks, while base TD evaluations retain their usual augmentation procedure. This avoids independent sampling noise in the recurrence from quantile fractions or image transformations. DER importance-sampling weights multiply the base TD term only; the recurrence penalty is averaged uniformly over valid sampled windows.

### C.4 Update pseudocode

Algorithms 2 and 3 expand the critic update of Algorithm 1. Collection, target updates, and actor updates follow the selected baseline recipe. In particular, the DQN listing does not require Double DQN or a particular target-update rule. The initialization below follows § 5.1.

**Algorithm 2** DQN/DER critic update with Q-Hankeling

**Require:** Online critic $Q_\theta$, coefficients $c \in \mathbb{R}^r$ initialized randomly with $\mathbf{1}^\top c = 1$, update count $g$, warm-up $K_0$

1: Sample the base replay minibatch and construct its usual TD targets and $\mathcal{L}_{\text{base}}(\theta)$
2: Fetch each endpoint's $r$ predecessors; keep valid windows in $\mathcal{B}_{\text{H}}$ using episode and time-step checks
3: Evaluate all $r+1$ values per valid window with the online critic
4: $\epsilon_b \leftarrow Q_\theta(s_{t_b}, a_{t_b}) - \sum_{i=1}^{r} c_i Q_\theta(s_{t_b-i}, a_{t_b-i})$
5: $\mathcal{L}_{\text{Hankel}}(\theta, c) \leftarrow |\mathcal{B}_{\text{H}}|^{-1} \sum_{b \in \mathcal{B}_{\text{H}}} \text{Huber}(\epsilon_b)$; use zero if the set is empty
6: $\lambda_g \leftarrow \lambda \mathbb{I}[g \geq K_0]$; $\mathcal{L}(\theta, c) \leftarrow \mathcal{L}_{\text{base}}(\theta) + \lambda_g \mathcal{L}_{\text{Hankel}}(\theta, c)$
7: Backpropagate through current and lagged online values
8: Update $\theta$ and $c$ in separate optimizer groups; apply the baseline's network-gradient clipping to $\theta$ only
9: Apply the baseline target-update schedule; increment $g$

For DER, the base update retains its distributional targets, prioritized replay weights, and augmentations. The recurrence uses the fixed quantile grid and shared window augmentation described above.

The actor receives no direct recurrence penalty. Retaining the baseline counter convention also preserves the parity of TD3's delayed actor updates when a nonfinite critic update is skipped.

### C.5 Warm-up schedule

The regularizer is enabled after $K_0$ critic updates:

$$\lambda_g = \lambda \mathbb{I}[g \geq K_0], \tag{54}$$

where $g$ counts gradient steps after learning starts, following replay prefill. Table 5 gives the benchmark settings. Gradient-step and environment-step counts differ when the baseline performs multiple critic updates per environment interaction.

**Algorithm 3** SAC/TD3 update with Q-Hankeling

**Require:** Twin critics $Q_{\theta_1}, Q_{\theta_2}$, actor $\pi_\phi$, shared random $c$ with $\mathbf{1}^\top c = 1$, critic scale $\kappa$, warm-up $K_0$

1: Advance the baseline update counter and sample a replay minibatch
2: Construct the usual SAC entropy-regularized targets or TD3 smoothed targets $y_b$
3: $\mathcal{L}_{\text{base}}(\theta) \leftarrow \kappa \sum_{k=1}^{2} B^{-1} \sum_b (Q_{\theta_k}(s_b, a_b) - y_b)^2$
4: Retrieve valid predecessor windows $\mathcal{B}_{\text{H}}$ with episode and time-step checks
5: Compute $\mathcal{L}_{\text{Hankel}}(\theta, c)$ from Eq. (53), using stored actions and online critics
6: $\mathcal{L}(\theta, c) \leftarrow \mathcal{L}_{\text{base}}(\theta) + \lambda \mathbb{I}[g \geq K_0] \mathcal{L}_{\text{Hankel}}(\theta, c)$
7: If the critic loss is nonfinite, skip this update while retaining the advanced counter
8: Update the critics and $c$ jointly, with separate learning rates and no added gradient clipping
9: Apply the baseline actor, temperature (SAC), and target updates on their original schedules

### C.6 Computational complexity

For at most $B$ valid windows, the recurrence uses $B(r+1)$ critic values, $O(Br)$ scalar arithmetic, and $r$ learned coefficients. When the current-transition value is reused from the base update, only $Br$ additional critic evaluations are needed. Evaluations are batched across windows and time steps. For a dense $n \times n$ Hankel matrix formed from $2n-1$ values, explicit nuclear-norm regularization requires those value evaluations and typically an $O(n^3)$ singular-value decomposition, together with its backward pass (Fazel et al., 2001; Recht et al., 2010; Fazel et al., 2013). These costs separate network evaluation from matrix arithmetic; the recurrence and matrix objectives need not use the same horizon.

Schwerdtner et al. (2025) regularize Hankel singular values through a state-space parametrization. Here the sequence is produced by a learned critic evaluated on replay trajectories, so the method instead operates directly on scalar value windows.

Measured runtime ratios are reported in Appendix D.7. The comparisons in § 6 hold environment interactions and gradient-step budgets fixed; the regularizer adds computation without collecting additional environment samples.

# D EXPERIMENTAL PROTOCOL

## D.1 BENCHMARKS AND EVALUATION PROTOCOL

Each comparison starts from one baseline configuration and adds the Q-Hankeling block, with paired seeds, equal environment-interaction and gradient-step budgets, and the same hardware. Figure 4 summarizes the workflow; Table 3 lists the suites. The five-seed MuJoCo and Classical Control runs use seeds 22, 44, 66, 88, and 100; Atari uses seeds 0–4. Some supplementary MuJoCo comparisons contain two seeds, as detailed in Appendix E.2. Training returns are logged throughout; evaluation schedules and final-score protocols are specified below.

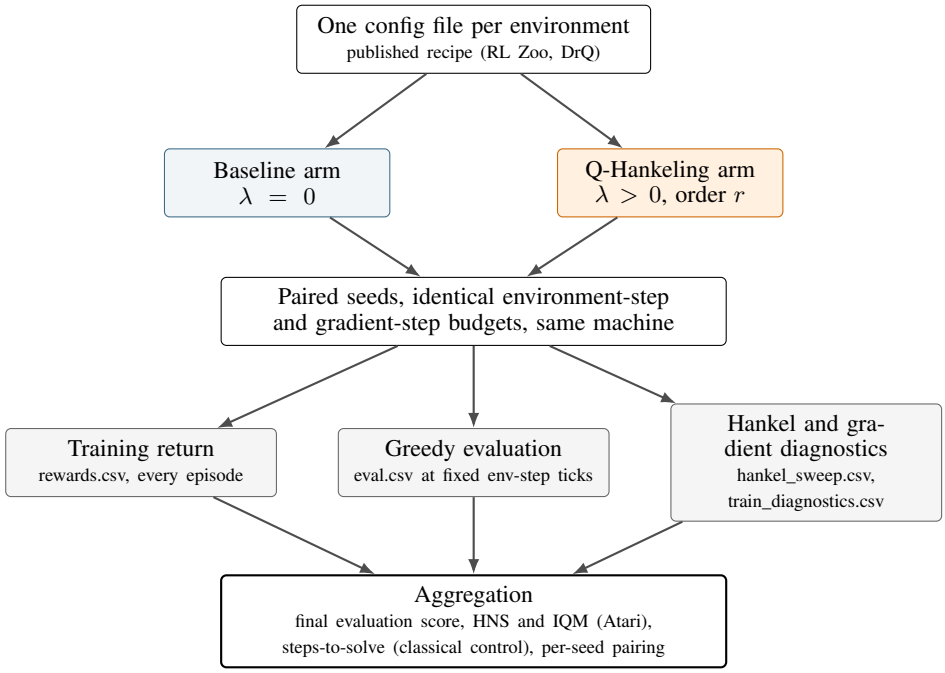



Figure 4: The evaluation pipeline shared by every experiment. The two arms of a comparison are produced from one configuration file and differ only in the Q-Hankeling block.

| Suite | Environments | Base algorithm | Seeds | Budget | Reported in |
|---|---|---|---|---|---|
| Atari-100k | 26 suite games | DER, DrQ configuration with IQN head | 5 | 100k steps | § 6, § E.4 |
| Atari-100k tuning | BankHeist, BattleZone, Boxing, DemonAttack, KungFuMaster | same | 3 | 50k steps | § E.5 |
| MuJoCo TD3 | HalfCheetah, Ant, Swimmer | SB3 TD3, RL Zoo configuration | 5 | 1M steps | § 6, § E.2 |
| MuJoCo SAC | HalfCheetah, Ant, Swimmer, Humanoid, HumanoidStandup | SB3 SAC, RL Zoo configuration | 2–5 | 1M steps | § 6, § E.2 |
| Classical Control | CartPole, MountainCar, Acrobot | SB3 DQN, RL Zoo configuration (MountainCar tuned, see text) | 5 | 50k, 120k, 100k steps | § 6, § E.1 |

Table 3: Summary of the experiment suites. Budgets count policy–environment interactions.

**Atari-100k.** The protocol follows Kaiser et al. (2020). The policy is trained using 100k environment interactions (400k frames with frame skip 4). The final policy of each seed is scored on 32 full games with $\varepsilon = 0.001$, the raw scores are recorded, and human normalized scores are computed with the random (a policy choosing uniformly among the 18 actions) and human references of Wang et al. (2016). The published methods shown alongside our arms are quoted from their papers and are not rerun. The tuning subset (BankHeist, BattleZone, Boxing, DemonAttack and KungFuMaster) was run for 50k steps with $\lambda \in \{1, 2, 4, 10, 100\}$ at $r = 8$ and $r \in \{4, 8, 16\}$ at $\lambda = 2$, and the setting $\lambda = 2$, $r = 8$ was selected on the mean human normalized score at 50k steps before any 100k run was made.

**MuJoCo.** We use the v5 environments with the RL Zoo defaults for SAC and TD3, a budget of one million environment steps, a deterministic greedy evaluation of five episodes every 20k steps, and a Q-Hankeling warm-up of 10k gradient steps after the 10k step prefill. The grid $\lambda \in \{0.1, 1, 10\} \times r \in \{2, 8\}$ was run per environment and base algorithm (with $\lambda = 20$ added for Humanoid), and settings are selected by final greedy return. Table 1 gives the main comparison; supplementary configurations and sample sizes are specified in Appendix E.2.

**Classical Control.** CartPole, MountainCar and Acrobot start from the RL Zoo DQN configurations, with a greedy evaluation of 50 episodes every 2k environment steps on CartPole and every 4k steps on MountainCar and Acrobot. The RL Zoo recipe for Acrobot is the quantile regression variant of DQN from SB3-Contrib, reported as DQN throughout since the penalty and the comparison are identical. For MountainCar, a tuning study compared 15 variants of the target-update rule, learning rate, network width, replay capacity, and exploration schedule. Both methods use the selected recipe, with Polyak target updates and learning rate $10^{-3}$.

### D.2 Metrics

We report final policy quality and steps to return thresholds. For Atari aggregates, we additionally use the interquartile mean (IQM) and stratified bootstrap confidence intervals following Agarwal et al. (2021).

Steps to first reach a threshold measures sample efficiency. On MuJoCo, it is the first environment step at which the training return, smoothed over the last 10 completed episodes, is at or above a threshold. On Classical Control, thresholds are measured on the greedy evaluation return (50 episodes per tick). It is computed per seed, then reported as the mean and standard error over the seeds that reached it, always alongside the count $k/n$ of seeds that did, and a threshold no seed reaches is reported as not reached. Five thresholds are used per environment (Table 4) so that we can see when the sample efficiency benefit appears and how it evolves, and the main results table reports three selected thresholds. The training stream is generated by the behaviour policy together with its exploration noise, which is why a smoothed training return can sit below a level that the greedy evaluation reaches.

Final greedy evaluation return measures policy quality. It is the greedy or deterministic evaluation score of the final policy, reported as the mean and standard deviation over seeds. On Atari it is reported as the human normalized score, aggregated over the 26 games by the mean, the median and the IQM, with 95% confidence intervals over seeds within each game, together with the per game win probability $P(\text{DER-QH} > \text{DER})$ over all seed pairs. For Table 1, MuJoCo uses the last logged deterministic evaluation (5 episodes per seed), and Classical Control uses the last greedy evaluation (50 episodes per seed).

We also record Hankel and gradient diagnostics to examine how the penalty affects critic learning. The Hankel diagnostics are the effective rank and the normalized spectrum $\sigma_i/\sigma_1$ of the Hankel matrix of the critic trace $Q(s_t, \pi(s_t))$ along a deterministic rollout of a checkpoint, computed exactly as in § 4, where the effective rank is the smallest $k$ whose top $k$ singular values capture $99.9\%$ of the Frobenius energy. Alongside this we measure the same quantities on the penalised replay windows sampled during training, so that the rank of what the penalty acts on can be compared between the two arms. The gradient diagnostics are the gradient share $\omega = \|\lambda \nabla \mathcal{L}_{\text{Hankel}}\| / \|\nabla \mathcal{L}_{\text{base}}\|$ and the cosine between $\nabla \mathcal{L}_{\text{base}}$ and $\nabla \mathcal{L}_{\text{Hankel}}$ over the critic parameters, both logged every few hundred gradient steps. The share tells us how much of the update the penalty accounts for, and the cosine whether the two terms pull in the same direction.

### D.3 HANKEL MEASUREMENT AND CHECKPOINT SAMPLING

**Value sequences.** Along each evaluation rollout $(s_t, a_t)_{t=1}^{T}$, collected with the greedy action for value-based policies, the actor mean for SAC, and the noiseless actor output for TD3, we evaluate the critic to obtain two scalar sequences. The state-action sequence is

$$q_t = Q_\theta(s_t, a_t), \tag{55}$$

as in Eq. (8), and the state sequence is

$$v_t = \begin{cases} \max_a Q_\theta(s_t, a), & \text{value-based policies (DQN, DER),} \\ Q_\theta\big(s_t, \pi_\phi(s_t)\big), & \text{actor-critic policies (SAC, TD3),} \end{cases} \tag{56}$$

where $\pi_\phi(s_t)$ is the deterministic action used for evaluation. For DER, $Q_\theta$ is the mean over the fixed midpoint quantile grid of Appendix **??**; for twin critics, we use the first critic $Q_{\theta_1}$. Each sequence is arranged into the Hankel matrix of Eq. (8) and its effective rank is computed with Eq. (9). Because rollouts follow the evaluation rule, $a_t$ is the action that defines $v_t$, so $q_t$ and $v_t$ coincide up to ties in the greedy action; the $v_t$ curves in Figure 3 therefore serve as a consistency check rather than independent evidence about state values. In particular, $v_t$ for SAC is the critic evaluated at the actor mean, not the soft state value of the stochastic policy.

Rank of the rollout Hankel matrix relative to its size for the stock algorithms at the start, middle and end of training, one colour per suite, on the same environments as § 6. For every environment and seed, $W$ is the Hankel matrix of the critic along a greedy rollout ($Q(s_t, a_t)$ for discrete-action policies, $Q(s_t, \pi(s_t))$ with three rollouts stacked for continuous-action policies) and $\mathrm{rank}(W)$ is the smallest $k$ whose top-$k$ singular values hold $99.9\,\%$ of the Frobenius energy, reported over $\dim(W) = \min(\mathrm{rows}, \mathrm{cols})$, the attainable rank (dashed line). Start is the untrained network (first analysis tick), middle the tick nearest $50\,\%$ of the environment step budget, end the last tick (within the final $25\,\%$ of the budget). Every value is drawn as a dot, and the boxes pool the suite's environments and seeds (median, quartiles, $1.5$ IQR whiskers): Classical Control (CartPole, MountainCar, Acrobot) 3 SB3 DQN families $\times$ 5 seeds, with three rollouts each (45 values per box), MuJoCo (HalfCheetah, Ant, Swimmer, Humanoid, HumanoidStandup) 7 SAC/TD3 arms $\times$ 5 seeds (35 values), and Atari-100k, the standard 26-game split, $\times$ 4 seeds of DER (104 values at the start), whose analysis ticks fall every 200 episodes, so only the games with a tick inside the middle half of the budget (76 values) or its final quarter (45 values) contribute to the middle and end boxes.

### D.4 THRESHOLDS, BUDGETS AND WARM-UP

Table 4 gives the threshold ladders used in the supplementary plots. Table 10 reports three selected thresholds per task. Warm-up counts begin after replay prefill; the dormant percentages in Table 5 exclude that prefill.

| Environment | Base algorithm | Thresholds | Headline |
|---|---|---|---|
| HalfCheetah | SAC, TD3 | 2000, 4000, 6000, 8000, 10000 | RL Zoo 1M score |
| Ant | SAC, TD3 | 1000, 2000, 3000, 4000, 5000 | RL Zoo 1M score |
| Swimmer | SAC, TD3 | 100, 150, 200, 250, 300 | RL Zoo 1M score |
| Humanoid | SAC | 1000, 2000, 3000, 4000, 5000 | 5000 |
| HumanoidStandup | SAC | 60k, 90k, 120k, 150k, 180k | 180k |
| MountainCar | DQN | $-180$, $-160$, $-140$, $-120$, $-110$ | $-110$ |
| CartPole | DQN | 100, 200, 300, 400, 475 | 475 |
| Acrobot | DQN | $-300$, $-200$, $-150$, $-120$, $-100$ | $-100$ |
| Atari-100k | DER | 25, 50, 75 and 100% of the baseline's final training return, per game | 100% |

Table 4: Return thresholds used for the steps to threshold metric. The final column gives the top threshold of each plotted ladder. For HalfCheetah, Ant and Swimmer it is the RL Baselines3 Zoo score of the same recipe at 1M steps, the Zoo reports Humanoid only at 2M steps and has no HumanoidStandup entry, so those two use their top threshold.

| Base algorithm | Environments | Prefill (env steps) | Warm-up $K_0$ (grad steps) | Budget (env steps) | Dormant (%) |
|---|---|---|---|---|---|
| SAC, TD3 | HalfCheetah, Ant, Swimmer, Humanoid, HumanoidStandup | 10,000 | 10,000 | 1,000,000 | 1.0 |
| DQN | CartPole | 1,000 | 256 | 50,000 | 1.0 |
| DQN | MountainCar | 1,000 | 500 | 120,000 | 0.8 |
| DQN | Acrobot | 0 | 2,000 | 100,000 | 2.0 |
| DER | Atari-100k (26 games) | 1,600 | 10,000 | 100,000 | 5.0 |

Table 5: Warm-up of the Q-Hankeling penalty (Eq. (54)). $\lambda$ is held at zero for $K_0$ gradient steps once learning starts after the replay buffer prefill, then set to its full value. The last column is the share of the budget during which the penalty is dormant. On Atari, $K_0 = 10{,}000$ gradient steps corresponds to 5,000 environment steps at the DER replay ratio.

### D.5 Software

All experiments share one Python 3.12 codebase built on PyTorch (Paszke et al., 2019). The environments use Gymnasium 1.3 (Towers et al., 2025), with ALE-py 0.12 (Bellemare et al., 2013) for the Atari games and MuJoCo 3.1.6 (Todorov et al., 2012) for the locomotion tasks. We use Stable-Baselines3 2.9 with SB3-Contrib 2.9 (Raffin et al., 2021) for the reference DQN, QR-DQN, SAC and TD3 implementations onto which the Q-Hankeling penalty is added, with their RL Zoo configurations (Raffin, 2020), and our own implementation of DER with the DrQ configuration and an IQN head for Atari-100k. Optuna 4.9 was used for the Classical Control tuning study and NumPy, SciPy and Matplotlib for analysis. Dependencies are pinned with a `uv` lockfile on the workstation, the cloud instance and the laptop. On the cluster the same dependencies were installed into a Conda environment with PyTorch 2.5.1 built against CUDA 12.1, whereas the workstation used the aarch64 PyTorch 2.12 build with CUDA 13.0. Device selection is automatic (CUDA, then MPS, then CPU), so every configuration file runs unchanged on all machines. Setting $\lambda = 0$ removes the additional loss and recovers the selected baseline objective, including the recipe exceptions described in Appendix D.1.

### D.6 Hardware

Table 6 lists the compute platforms. Within each comparison, baseline and regularized runs use the same machine and budgets; the runtime ratios in Appendix D.7 are therefore within-machine comparisons.

| System | CPU | Accelerator | Memory | Experiments |
|---|---|---|---|---|
| Workstation (NVIDIA DGX Spark) | GB10, 20 core Arm | GB10 Blackwell GPU (CUDA 13.0) | 128 GB unified | Atari-100k seed 0, MuJoCo SAC sweeps, Classical Control |
| University HPC cluster | Intel Xeon Ice Lake, 8 cores per job | NVIDIA L40S 48 GB, 1 per job | 20 GB per job | Atari-100k seeds 1 to 4 and the tuning subset |
| Cloud CPU instance (64 vCPU) | Intel Xeon Platinum 8272CL | none | 504 GB | MuJoCo five seed SAC and TD3 runs |
| Laptop (Apple M1 Pro) | Apple M1 Pro | M1 Pro GPU (MPS) | 16 GB unified | Classical Control |

Table 6: Compute systems used in this paper and the experiments run on each.

### D.7 Measured runtime

Table 7 reports the wall-clock time per run relative to the baseline on the same machine, complementing the complexity analysis in Appendix C.6. The overhead grows with the order $r$, as expected from the $r$ additional critic evaluations per sample, and on Atari it comes almost entirely from passing $r + 1$ frame stacks through the encoder. These comparisons hold environment interactions and

gradient-step budgets fixed. An equal-wall-clock control with a higher baseline replay ratio has not been run, so the results do not establish an advantage at equal compute.

| Base algorithm | Environments | Machine | $r = 2$ | $r = 8$ |
|---|---|---|---|---|
| TD3 | HalfCheetah, Ant, Swimmer | cloud CPU instance | $1.2\times$ | $1.7$ to $1.8\times$ |
| SAC | HalfCheetah, Ant | cloud CPU instance | $1.05$ to $1.13\times$ | $1.3$ to $1.5\times$ |
| SAC | Swimmer, Humanoid, HumanoidStandup | workstation | — | up to $2.0\times$ |
| DER | Atari-100k | workstation, cluster | — | — |

Table 7: Wall-clock time per run with Q-Hankeling relative to the baseline, same machine and same budget. Dashes indicate measurements not reported here; Atari runtime is not reported.

# E Additional Results and Diagnostics

We group the supplementary results by benchmark suite, followed by hyperparameter and regularization diagnostics. The protocols are in Appendix D; Table 10 collects the threshold comparisons at the settings used in the main text.

## E.1 Classical Control

The Classical Control experiments use seeds 22, 44, 66, 88, and 100, with 50 greedy evaluation episodes every 2k environment steps on CartPole and every 4k on MountainCar and Acrobot. The main panels use the same $(\lambda, r)$ choices as Table 1. Figure 5 tracks policy quality, Figure 6 reports threshold crossing, and Figure 7 tracks the effective Hankel rank. The return comparisons remain mixed: the regularizer improves some learning stages and tasks, while Acrobot's final return is lower. Rank trajectories provide a complementary structural diagnostic and should not be read as a guarantee of improved returns.

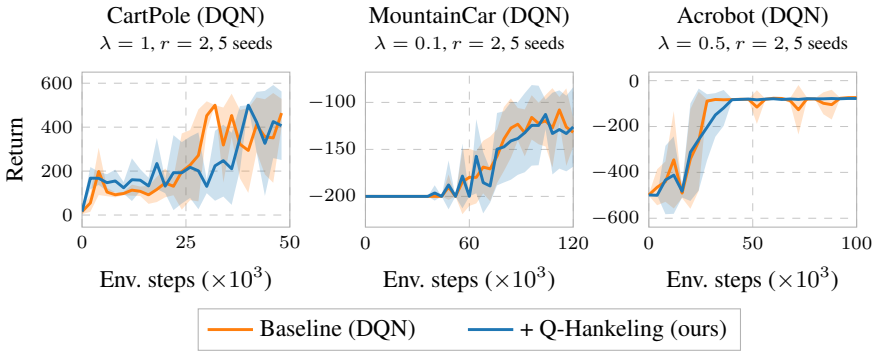



Figure 5: Classical Control evaluation returns at the settings in Table 1. Lines show the mean over five seeds; shading shows one standard deviation. Each evaluation averages 50 greedy episodes, every 2k steps on CartPole and every 4k on the other tasks.

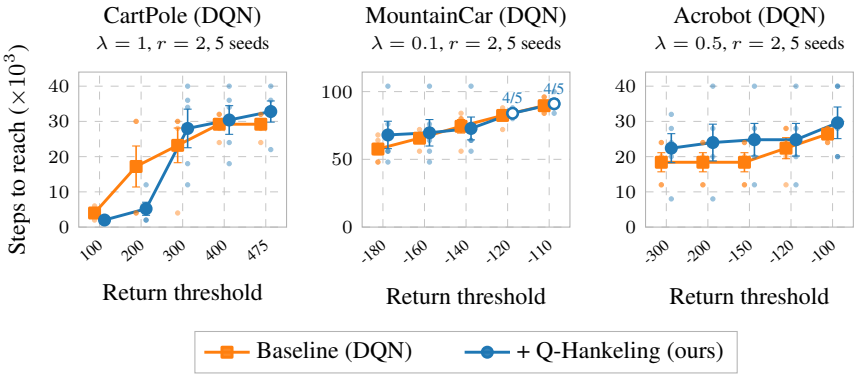



Figure 6: Classical Control steps to each threshold in Table 4, measured on greedy evaluation return. Dots show individual seeds; markers show the mean with one standard error over seeds that reach the threshold. Open markers and $k/n$ labels indicate that only $k$ of $n$ seeds reach it.

**Selection for sample efficiency.** Figures 8 and 9 show the alternative configurations selected to minimize total steps across the three-threshold ladder, counting unreached thresholds at the full budget. They use $(\lambda, r) = (0.5, 2)$ for CartPole and $(0.1, 8)$ for MountainCar and Acrobot. This selection criterion differs from final-return selection, so these panels are reported separately.

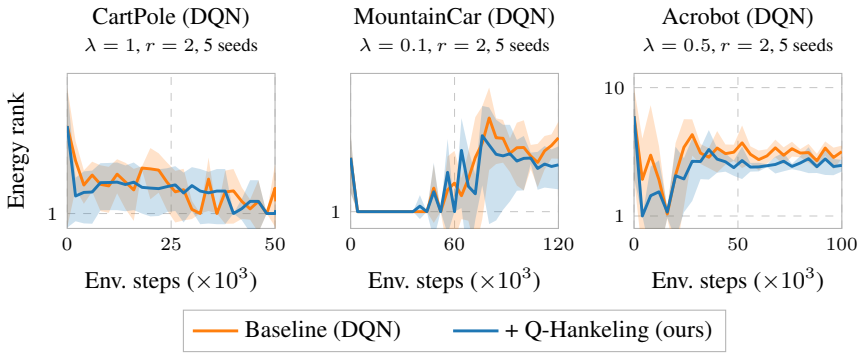



Figure 7: Effective rank of the rollout Hankel matrix of $Q(s_t, a_t)$ on Classical Control, using 99.9% of the Frobenius energy. Values are averaged over 50 greedy rollouts at each evaluation tick and then reported as the mean with one standard deviation over five seeds.

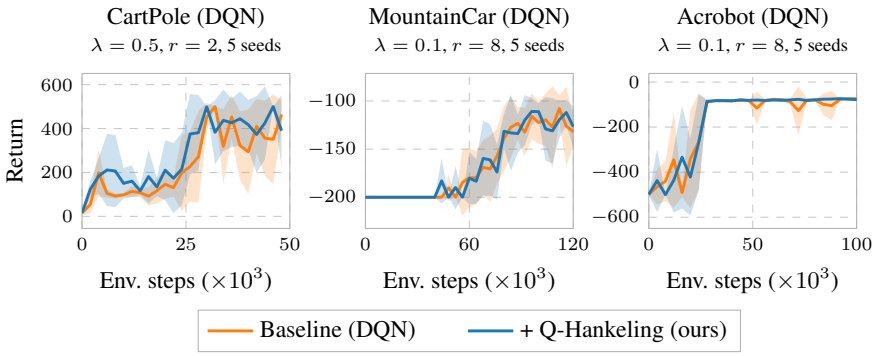



Figure 8: Classical Control evaluation curves for the configurations selected for sample efficiency. Evaluation and uncertainty conventions are as in Figure 5.

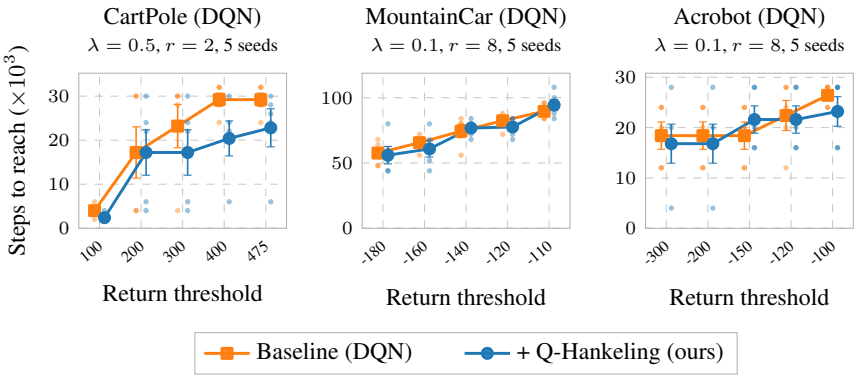



Figure 9: Threshold crossing for the configurations selected for sample efficiency. Marker and uncertainty conventions are as in Figure 6.

### E.2 MuJoCo

Figures 10 and 11 report the supplementary threshold and evaluation curves. Ant (SAC and TD3), HalfCheetah–TD3, and Swimmer–TD3 use five seeds; the HalfCheetah, Swimmer, Humanoid, and HumanoidStandup SAC panels use two. The two-seed comparisons use seeds 44 and 66. HalfCheetah–SAC uses $(\lambda, r) = (10, 8)$ here, compared with $(1, 2)$ in Table 1; HalfCheetah–TD3 uses $(1, 2)$ here and $(0.1, 2)$ in that table. The Humanoid and HumanoidStandup panels show the two-seed subsets of the corresponding five-seed main comparisons. Swimmer–SAC uses two seeds in both. These distinctions should be retained when comparing the curves with the main-table scores. All learning curves use five deterministic evaluation episodes per checkpoint, every 20k environment steps.

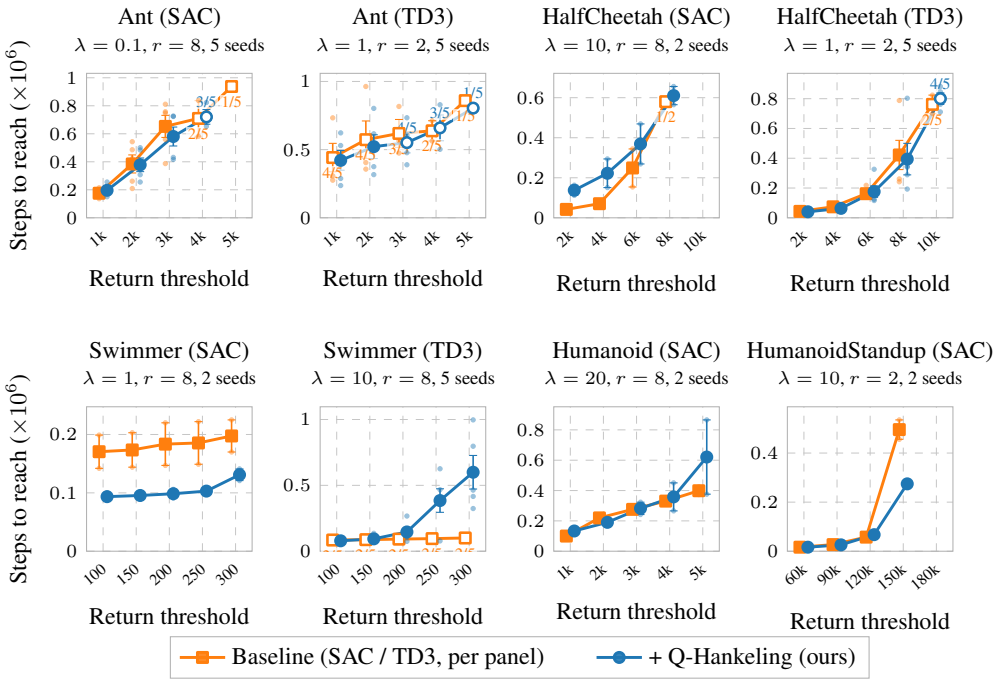



Figure 10: Sample efficiency on the MuJoCo tasks: environment steps at which the training return (rolling mean over 10 episodes) first reaches each threshold, for the baseline and its Q-Hankeling variant in each panel. Small dots: individual seeds; markers: mean over the seeds that reached the threshold, with one standard error; an open marker with a $k/n$ label means only $k$ of the $n$ seeds reached it within the budget, and a missing marker means none did.

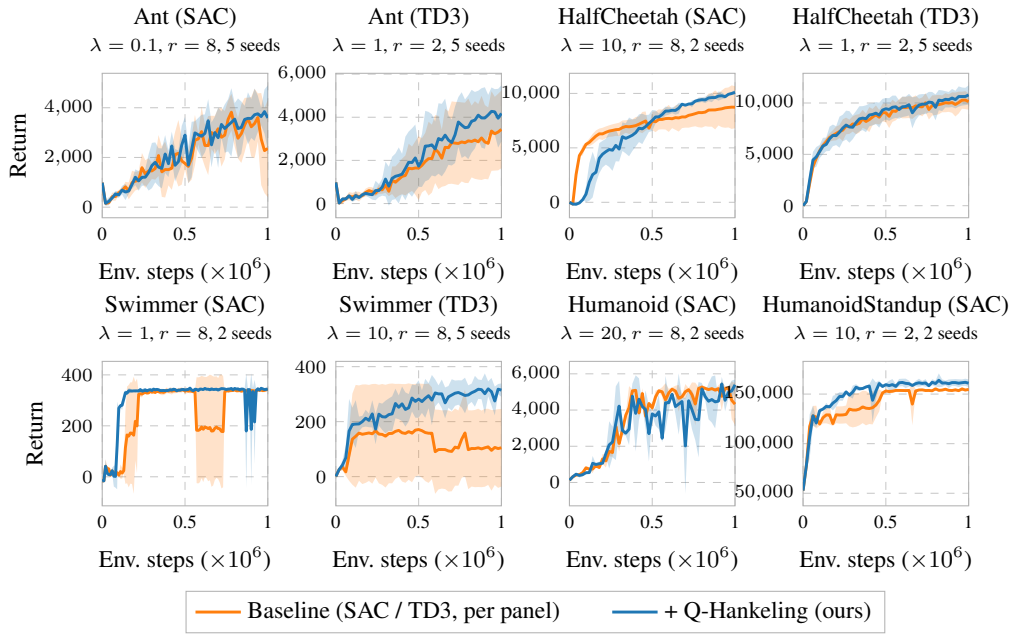



Figure 11: Learning curves on the MuJoCo tasks: the stock host algorithm (SAC or TD3, orange) and the same algorithm with the Q-Hankeling loss (blue) at the arm named in each panel title. Lines: mean over seeds of the evaluation return (deterministic policy, 5 episodes every 20k environment steps), shaded: $\pm$ one standard deviation across seeds. The seed count per panel is in the title.

### E.3 HANKEL SPECTRA OF THE CRITIC TRACE

This section collects the normalized Hankel singular values $\sigma_i/\sigma_1$ behind the effective ranks reported in Figure 1, Figure 7 and Appendix E.4.3, for every five-seed family on Classical Control (CartPole, MountainCar, Acrobot) and MuJoCo (Ant, HalfCheetah, and Swimmer with SAC and TD3 as available). Figure 12 extends the single-run MountainCar example of Figure 2 to the three tasks and five seeds: at every evaluation tick the 50 greedy rollouts each give one Hankel matrix of $Q(s_t, a_t)$, the rollout-mean spectrum is formed at the indices reached by at least half of the rollouts, and the curves show the mean over seeds. Figure 13 compares the end-of-training spectra of the baseline and of the Q-Hankeling arm of Table 1 with the untrained network. Figure 14 shows the same comparison for the MuJoCo families on the final checkpoint, with the three-rollout stacking of Eq. (8), and Figure 15 the spectra of the replay windows that the penalty acts on. Figure 16 overlays the baseline spectra of all tasks. The vertical rule at $i = r + 1$ in the comparison panels marks the singular value neglected by an order-$r$ recurrence (Theorem 4.1). In every family the trained critics decay by several orders of magnitude within the first few singular values, whereas the full-rank references decay slowly; the Q-Hankeling arms lower the tail beyond $r + 1$ on Ant and on the classical tasks and leave the rank-one HalfCheetah and Swimmer spectra essentially unchanged.

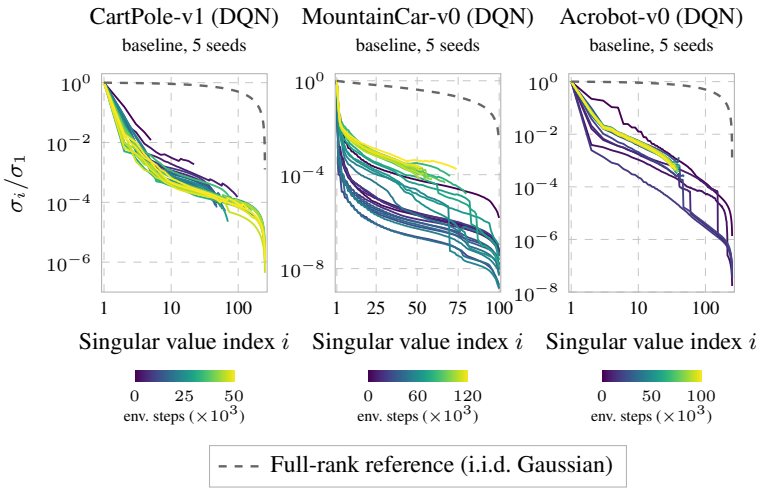



Figure 12: Normalized Hankel singular values $\sigma_i/\sigma_1$ of the greedy-rollout Hankel matrix of $Q(s_t, a_t)$ for baseline DQN on the Classical Control tasks at every evaluation tick (colour: environment steps; every 2k steps on CartPole, every 4k on MountainCar and Acrobot). Each curve is the mean over five seeds of the rollout-mean spectrum over 50 greedy evaluation rollouts, at the indices reached by at least half of the rollouts. Dashed: mean spectrum of i.i.d. Gaussian matrices of the largest Hankel shape on the panel (full rank). The index axis is logarithmic where the rollout length, and hence the matrix size, changes during training. Figure 2 shows one MountainCar seed at the analysis ticks, coloured by training episode.

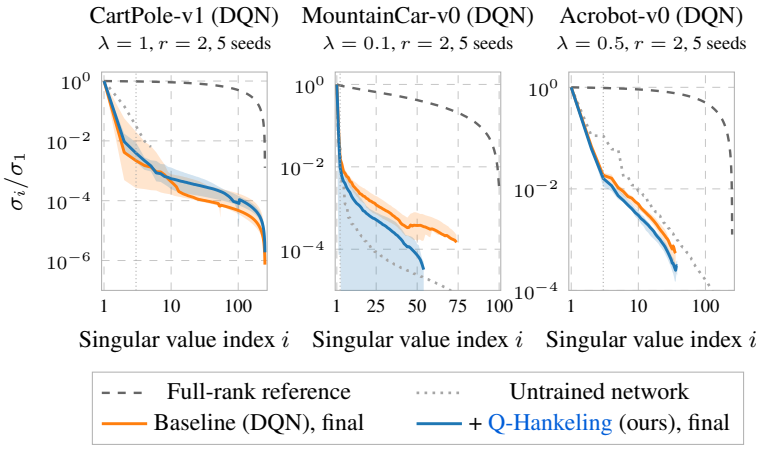



Figure 13: End-of-training spectra on Classical Control: the last evaluation tick of the baseline and of the Q-Hankeling arm of Table 1 (median over five seeds of the rollout-mean spectrum over 50 greedy rollouts; band: min–max over seeds). Dotted grey: the untrained network at the first evaluation tick (on CartPole its rollouts last under a dozen steps, so the spectrum is short). Dashed: full-rank reference as in Figure 12. The vertical rule marks $i = r + 1$ for the arm's recurrence order.

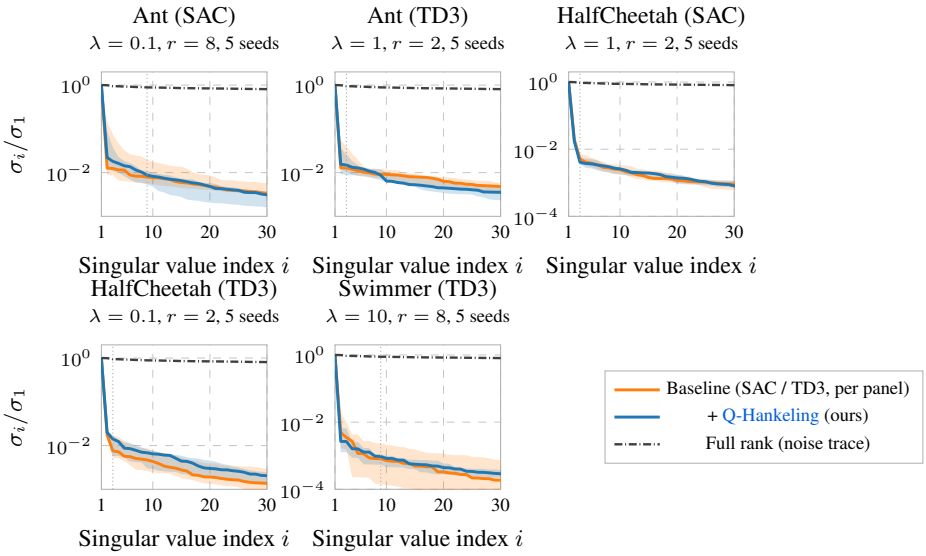



Figure 14: Final-checkpoint rollout-Hankel spectra on the five-seed MuJoCo families. Three deterministic rollouts of the final checkpoint (reset seeds 52–54; actor mean for SAC, noiseless actor for TD3) give the Hankel matrices of $Q(s_t, \pi(s_t))$, stacked vertically as in Eq. (8) ($1503 \times 500$ for 1000-step episodes). Lines: median over five seeds of $\sigma_i/\sigma_1$, band: min–max, baseline against the Q-Hankeling arm of Table 1; only the leading 30 indices are shown. Dash-dotted: Hankel matrix of a white-noise trace of the same length and stacking (full rank). The vertical rule marks $i = r + 1$.

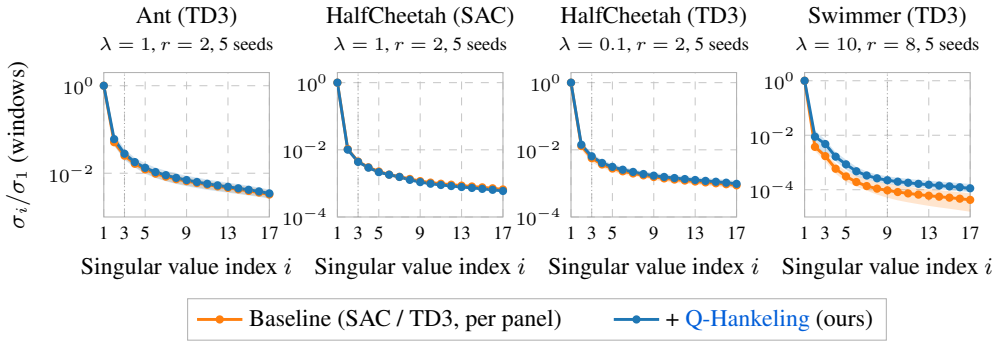



Figure 15: Spectra of the replay windows on which the penalty acts (Ant–TD3, HalfCheetah–SAC, HalfCheetah–TD3, Swimmer–TD3). Every 5000 gradient steps each run stacks the online $Q$-values along the 17-step trajectory windows ending at the transitions of the current replay batch into an $n \times 17$ matrix ($n \leq 256$, both critics) and records its singular values. Shown: $\sigma_i/\sigma_1$ averaged over the probe ticks in the final quarter of training and over the two critics, then mean over five seeds (band: min–max). The Q-Hankeling residual of Eq. (17) uses the trailing $r + 1$ columns of every window; the vertical rule marks $i = r + 1$. Ant–SAC was trained without the window probe and is omitted.

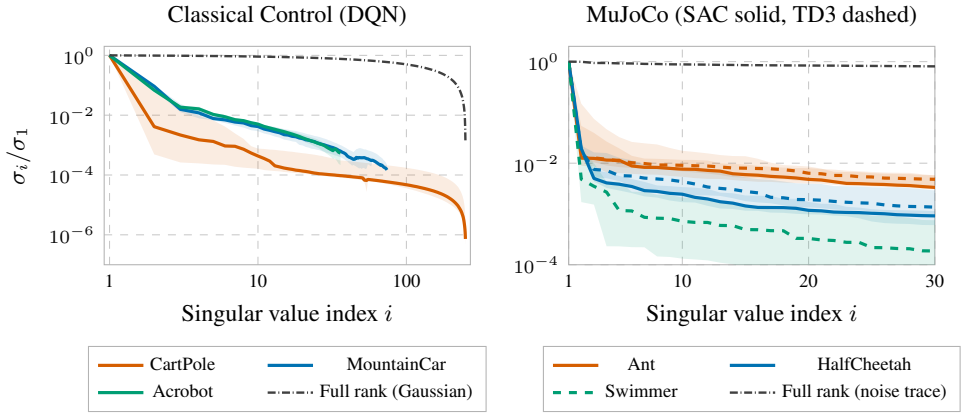



Figure 16: End-of-training spectra of every baseline critic on one axis per suite (colour: task; median over five seeds, band: min–max). Left: Classical Control DQN at the last evaluation tick, as in Figure 13. Right: MuJoCo SAC (solid) and TD3 (dashed) at the final checkpoint, as in Figure 14. Dash-dotted: the full-rank reference of each suite.

## E.4 ATARI-100K

### E.4.1 HUMAN-NORMALIZED SCORE COMPARISON

Figure 17 compares DER-QH with DER per game and places their aggregate scores alongside published model-free policies. DER-QH improves the mean on 17 of 26 games: mean HNS rises from 0.463 to 0.530, median HNS from 0.290 to 0.302, and IQM from 0.277 to 0.308. Gains and losses vary substantially by game, and many individual confidence intervals include zero. Published comparison scores are quoted from their papers rather than rerun.

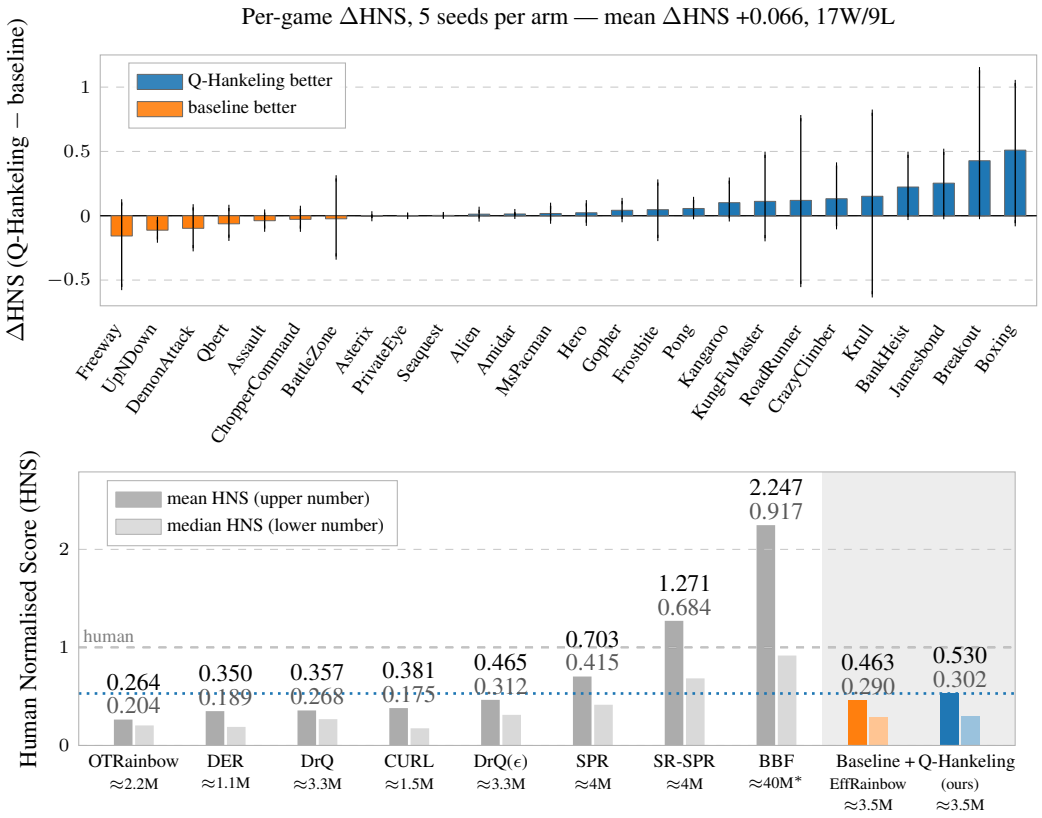



Figure 17: Per-game difference in human-normalized score between the Q-Hankeling variant and the baseline (bar: difference of the per-game mean scores over the seeds, whiskers show 95 % bootstrapped CI). Positive bars favour Q-Hankeling. Q-Hankeling outperforms the baseline on 17 of the 26 games. Atari-100k, 26 games: mean and median human-normalized score of published model-free policies and of our DER baseline with and without the Q-Hankeling loss (5 seeds). Published rows are quoted from(EfficientZero Table 1 and BBF Table A.1).

### E.4.2 TRAINING CURVES

The full training curves in Figures 18 and 19 show behavior-policy returns, including exploration. These differ from the near-greedy final policy evaluations. The trajectories expose both gains and regressions across training rather than only the final aggregate.

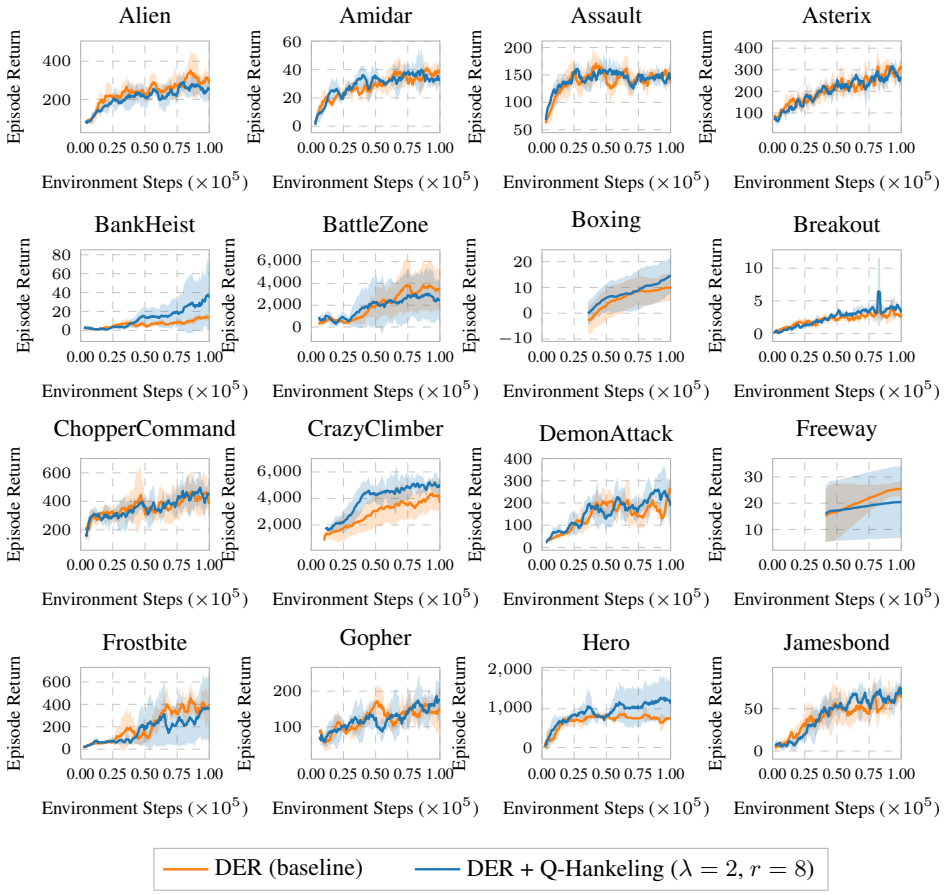



Figure 18: Training curves on the 26 Atari-100k games: DER baseline policies and policies trained with the Q-Hankeling loss. Lines: mean over 5 seeds of the raw episode return smoothed with a moving average of 20 episodes, on a common grid of 200 points over the 100k environment steps; shaded: ± one standard deviation across seeds. Panels whose Q-Hankeling arm has fewer seeds say so in the title. Games Alien–Jamesbond.

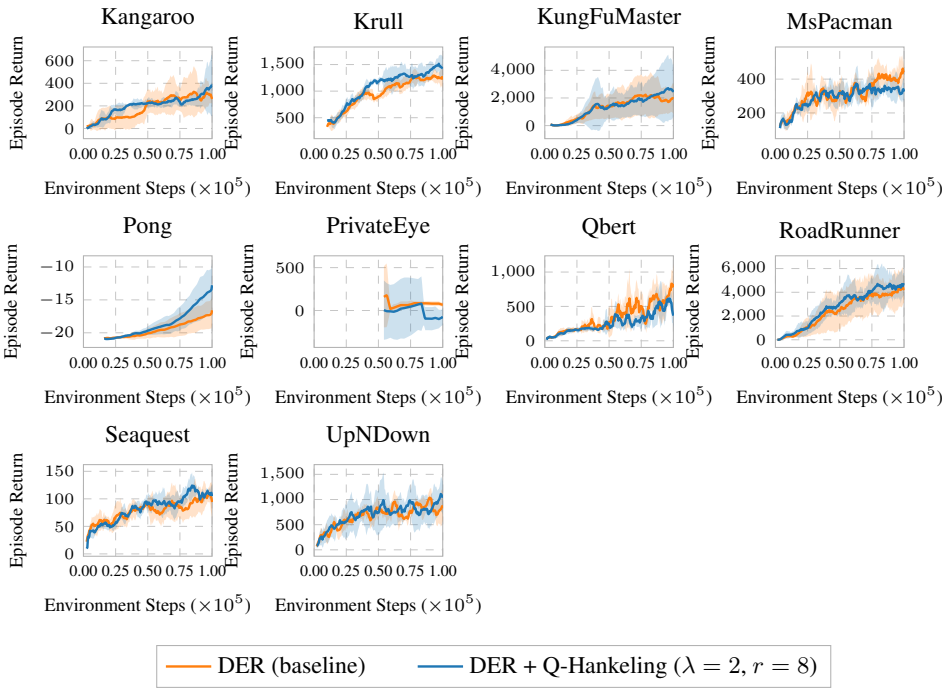



Figure 19: Training curves on the 26 Atari-100k games (continued): DER baseline policies and policies trained with the Q-Hankeling loss. Lines: mean over 5 seeds of the raw episode return smoothed with a moving average of 20 episodes, on a common grid of 200 points over the 100k environment steps; shaded: ± one standard deviation across seeds. Panels whose Q-Hankeling arm has fewer seeds say so in the title. Games Kangaroo–UpNDown.

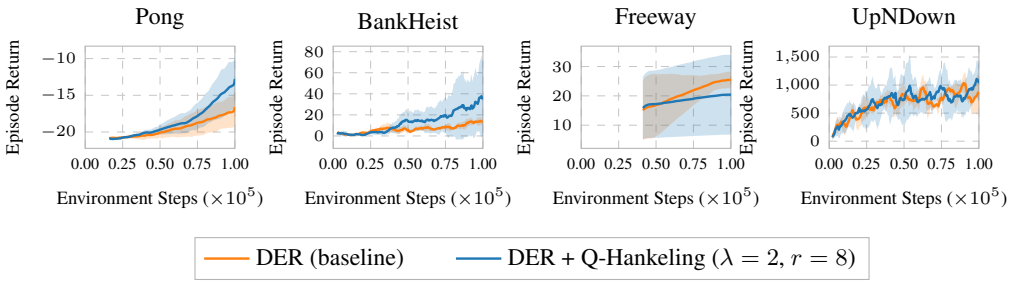



Figure 20: Training curves on Pong, BankHeist, Freeway and UpNDown (2 medium performers and 2 of the worst performers with Q-Hankeling based on Figure 17): DER baseline policies and policies trained with the Q-Hankeling loss. Lines: mean over 5 seeds of the raw episode return smoothed with a moving average of 20 episodes, on a common grid of 200 points over the 100k environment steps. Shaded: ± one standard deviation across seeds. All 26 games are in Figures 18–19.

### E.4.3 HANKEL DIAGNOSTICS

Table 2 gives the per-game final score comparison. The rank diagnostics measure two different objects: matrices of replay-window values during training and Hankel matrices of near-greedy rollouts at the final checkpoint. Replay diagnostics use 17-step windows, including when the recurrence order is $r = 8$; their full sampling protocol is in Figure 21. The centered variants subtract the scalar matrix mean before computing the spectrum, separating variation from the overall value level. DER-QH generally has lower replay-window effective rank, while final rollout Hankel ranks are similar between the two methods. These observations support an effect on training-time structure without establishing a causal link between rank reduction and return gains.

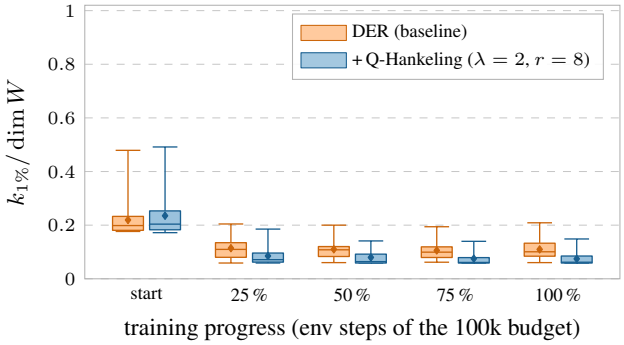



Figure 21: Effective rank of the replay-window $Q$ matrices along training on Atari-100k. Every 1000 gradient steps each run stacks the online $Q$-values along 17-step trajectory windows ending at the transitions of the current replay batch into an $n \times 17$ matrix $W$ ($n \leq 32$) and records its singular values; a shared order-$r$ recurrence on the windows would give $\operatorname{rank} W \leq r$. Reported: the smallest $k$ whose tail energy $\sum_{j>k} \sigma_j^2 / \sum_j \sigma_j^2 \leq \varepsilon$, divided by $\dim W = \min(n, 17)$, averaged over the ticks in the 5k environment steps preceding each stage mark and over 5 seeds; "start" is the first 5k steps, before the Q-Hankeling penalty is switched on. Each box: the 26 games (median, quartiles, whiskers = min/max, diamond = mean); $\varepsilon = 1\,\%$. At $\varepsilon \geq 5\,\%$ one constant mode already meets the threshold on nearly every game ($k = 1$); Appendix E.4.3 shows other thresholds and the centred variant.

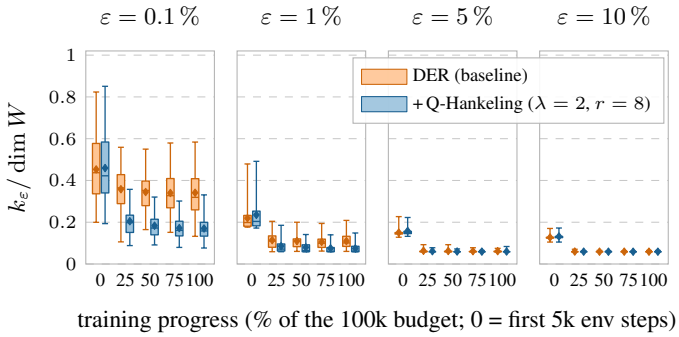



Figure 22: Effective rank of the raw replay-window $Q$ matrices (protocol of Figure 21) at tail-energy thresholds $\varepsilon \in \{0.1, 1, 5, 10\}\,\%$; the $x$ axis is the training progress in per cent of the 100k budget (0 = the first 5k environment steps).

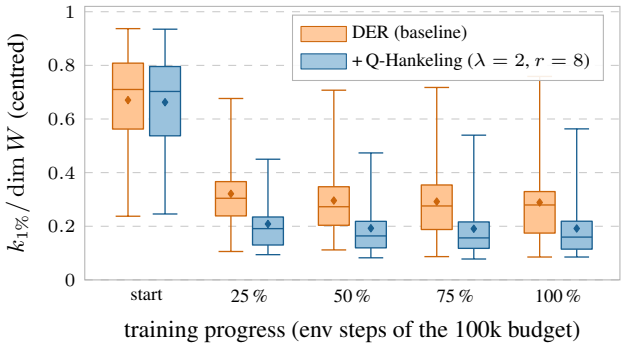



Figure 23: Centred variant: the spectrum of $W - \overline{W}$ (the matrix mean subtracted, which keeps a Hankel matrix Hankel), i.e. the number of modes needed by the variation of $Q$ around its level within the windows; $\varepsilon = 1\,\%$, otherwise as Figure 21.

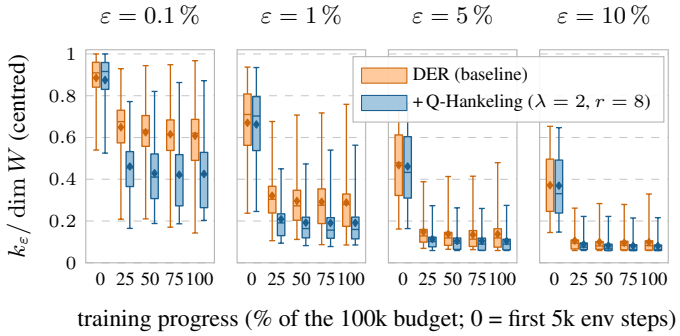



Figure 24: Centred variant at tail-energy thresholds $\varepsilon \in \{0.1, 1, 5, 10\}\,\%$ ($x$ axis as in Figure 22).

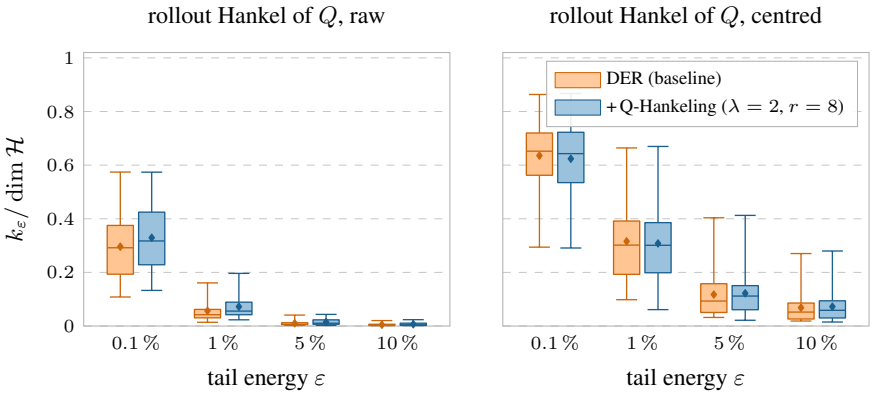
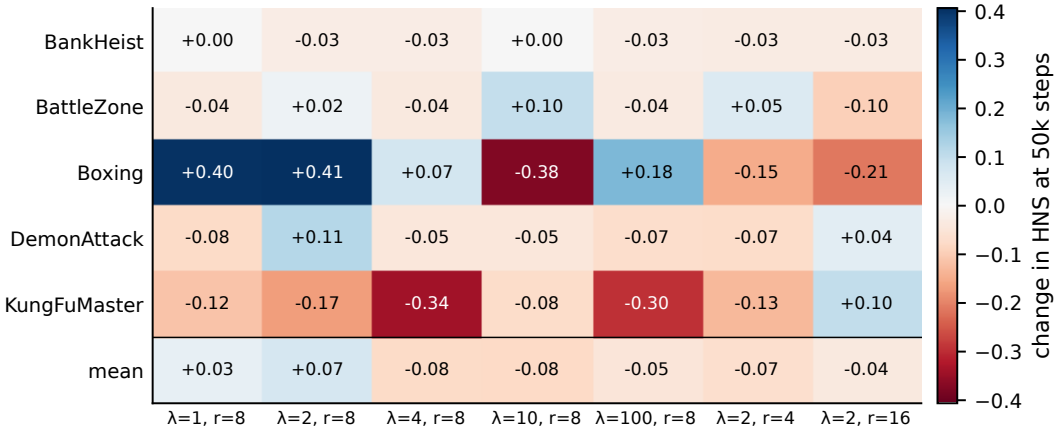



Figure 25: Effective rank of the rollout Hankel matrices at the end of training on Atari-100k: one near-greedy episode per final checkpoint, cut into 1024-step chunks, the $513 \times 512$ Hankel matrix of $Q(s_t, a_t)$ on each chunk, and the smallest $k$ whose tail energy $\sum_{j>k} \sigma_j^2 / \sum_j \sigma_j^2 \le \varepsilon$ divided by $\dim \mathcal{H} = 512$, averaged over the chunks and the available seeds. Left: raw matrices; right: matrix mean subtracted. Each box: the 26 games (median, quartiles, whiskers = min/max, diamond = mean).

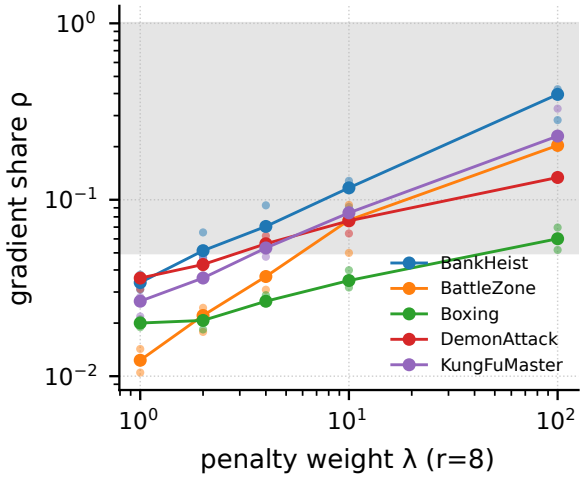
### E.5 Hyperparameter selection and regularization diagnostics

#### E.5.1 Atari tuning

Figure 26 shows the 50k-step tuning results for BankHeist, BattleZone, Boxing, DemonAttack, and KungFuMaster, with three seeds per setting. The mean change in HNS is highest at $(\lambda, r) = (2, 8)$ $(+0.07)$, followed by $(1, 8)$ $(+0.03)$; the other settings have negative mean changes. The selected setting is then used across all 26 games. The companion plot reports the gradient share, denoted $\rho$ in the export and $\omega$ in our notation. The order sweep is nonmonotone, so these results do not support a universal preference for longer recurrence windows.




(a) Change in HNS at 50k steps.




(b) Gradient share against $\lambda$, with $r = 8$.

Figure 26: Atari tuning on five games, three seeds per setting. The gradient-share plot uses $\rho$ for the quantity called $\omega$ in Appendix D.2.

#### E.5.2 Loss and gradient shares

**Regularization strength across environments.** The residual in Eq. (17) is measured in value units, whose scale varies across environments (roughly $10^2$ on CartPole, $10^3$ on HalfCheetah, and $10^5$ on HumanoidStandup). Consequently, the same $\lambda$ need not produce comparable critic updates. To compare the two loss terms, we log the unweighted gradient ratio $\|\nabla_\theta \mathcal{L}_{\text{Hankel}}(\theta, c)\| / \|\nabla_\theta \mathcal{L}_{\text{base}}(\theta)\|$ and its weighted counterpart $\omega$ defined in Appendix D.2. The selected $(\lambda, r)$ values are reported with each result in Tables 1 and 10.

Table 8 reports diagnostics from the selected runs. The weighted loss ratio is $\rho_{\text{loss}} = \lambda\mathcal{L}_{\text{Hankel}}(\theta, c)/\mathcal{L}_{\text{base}}(\theta)$; it need not track the gradient share $\omega$. The large variation across tasks illustrates why a common $\lambda$ does not imply comparable regularization strength. Rescaling $\lambda$ to a target loss ratio is only a linear diagnostic, since retraining can change both loss terms.

| Environment | Base algorithm | $(\lambda, r)$ | $\rho_{\text{loss}}$ | $\omega$ | $\lambda^\star$ for $\rho_{\text{loss}}$ of | |
|---|---|---|---|---|---|---|
| | | | | | 0.1 | 1 |
| Ant | SAC | 0.1, 8 | $0.014 \pm 0.001$ | | 0.71 | 7.1 |
| Ant | TD3 | 1, 2 | $0.11 \pm 0.04$ | $0.051 \pm 0.02$ | 0.89 | 8.9 |
| HalfCheetah | SAC | 1, 2 | $0.22 \pm 0.02$ | | 0.45 | 4.5 |
| HalfCheetah | TD3 | 0.1, 2 | $0.013 \pm 0.003$ | $0.0086 \pm 0.002$ | 0.78 | 7.8 |
| Swimmer | SAC | 1, 8 | $108 \pm 106$ | | 0.00093 | 0.0093 |
| Swimmer | TD3 | 10, 8 | $213 \pm 34$ | $0.77 \pm 0.08$ | 0.0047 | 0.047 |
| Humanoid | SAC | 20, 8 | $5.3 \pm 0.6$ | $19 \pm 2$ | 0.37 | 3.7 |
| HumanoidStandup | SAC | 10, 2 | $0.97 \pm 0.9$ | | 1.0 | 10 |
| MountainCar | DQN | 0.1, 2 | $0.12 \pm 0.02$ | $0.041 \pm 0.005$ | 0.081 | 0.81 |
| CartPole | DQN | 1, 2 | $0.92 \pm 0.1$ | $2.1 \pm 0.2$ | 0.11 | 1.1 |
| Acrobot | DQN | 0.5, 2 | $0.46 \pm 0.01$ | $0.48 \pm 0.007$ | 0.11 | 1.1 |
| Atari-100k | DER | 2, 8 | 0.0098 [0.0018, 0.027] | | 20 | 204 |

Table 8: Loss share $\rho_{\text{loss}}$ and gradient share $\omega$ of the Q-Hankeling penalty at the selected setting of every family, medians over the second half of training, mean $\pm$ standard deviation over available seeds. Empty cells are runs on which the gradient share was not logged. The Atari row is the median over the 26 games with the 25th and 75th percentiles across games in brackets. $\lambda^\star$ is the selected $\lambda$ rescaled to a loss share of 0.1 and of 1 under a linear extrapolation.

### E.5.3 Calibrating the penalty weight

The gradient share $\omega$ in Appendix D.2 compares the size of the recurrence and base-loss gradients over critic parameters. A probe that computes both gradients without applying the penalty can suggest an initial weight for a target share $\omega^\star$:

$$\lambda^\star = \omega^\star \frac{\text{median}_g \, \|\nabla_\theta \mathcal{L}_{\text{base}}\|}{\text{median}_g \, \|\nabla_\theta \mathcal{L}_{\text{Hankel}}\|}. \tag{57}$$

This is a calibration heuristic: the gradients and learned coefficients change when the regularizer is enabled. Figure 27 plots the gradient shares $\omega$ for the MuJoco and classical control environments. Table 9 reports the probe estimates; these are distinct from the selected-run diagnostics in Table 8.

| Environment | Baseline | $\omega$ at $\lambda = 0.5$ | $\lambda^\star$ ($\omega^\star = 0.1$) | $\lambda^\star$ ($\omega^\star = 1$) |
|---|---|---|---|---|
| CartPole | DQN | 0.63 | 0.08 | 0.8 |
| MountainCar | DQN | 0.44 | 0.11 | 1.1 |
| Acrobot | DQN | 0.014 | 3.6 | 36 |
| HalfCheetah | TD3 | — | 0.57 | 5.7 |
| Ant | TD3 | — | 1.1 | 11 |
| Swimmer | TD3 | — | 3.8 | 38 |

Table 9: Probe-based calibration from Eq. (57). The third column reports the realized share in the fixed-weight Classical Control runs. Dashes denote unreported measurements.

Gradient alignment provides a complementary local diagnostic. Write $u = \nabla_\theta \mathcal{L}_{\text{base}}$ and $v = \nabla_\theta \mathcal{L}_{\text{Hankel}}$. For a plain gradient step in direction $-(u + \lambda v)$, the directional derivative of the base loss is $-\|u\|^2 - \lambda\langle u, v\rangle$. When $\langle u, v\rangle < 0$, it is negative for

$$0 \le \lambda < \lambda^{\max} = \frac{\|u\|^2}{-\langle u, v\rangle}. \tag{58}$$

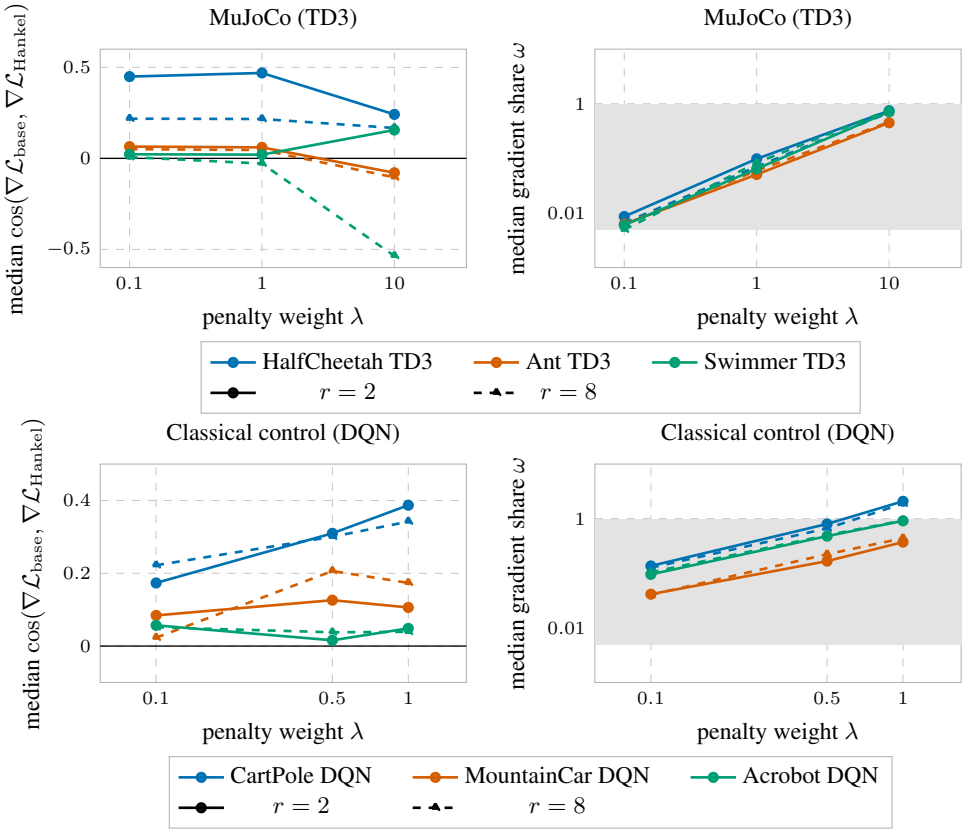



Figure 27: Gradient probe on the MuJoCo (TD3) and classical-control (DQN) families. Left column: the median cosine between the TD-loss and Q-Hankeling-loss gradients over the critic parameters, taken over the second half of training (per-seed medians, mean over seeds). Right column: the median gradient share $\omega = \lambda \|\nabla \mathcal{L}_{\text{Hankel}}\| / \|\nabla \mathcal{L}_{\text{base}}\|$ on the same window, with the band 0.005 to 1 shaded. One point per learned-$c$ arm of the $r \times \lambda$ screen; solid lines: order $r = 2$, dashed: $r = 8$; one colour per task. All plots use 5 seeds

This is a pointwise first-order condition, not a guarantee for a finite optimizer step or for a weight chosen from median probe statistics. The benchmark weights are selected by the sweeps in Appendix D.1, rather than an adaptive application of Eq. (58).

### E.6 THRESHOLD RESULTS AT THE MAIN COMPARISON SETTINGS

Table 10: Sample efficiency at the selected return thresholds. Arrows show baseline → Q-Hankeling; algorithm subscripts give $(\lambda, r)$. Each cell gives the return threshold above the mean steps to reach it, with reached/total seeds in subscripts; k denotes thousands. Bold suite colors favor more successful seeds, then fewer steps when counts match; ties are black. MuJoCo thresholds use rolling 10-episode training returns; Classical Control thresholds use greedy evaluation returns. Settings match Table 1; Atari has no steps-to-threshold results.

| Environment | Algo. → QH $(\lambda, r)$ | Target 1<br>Steps to target<br>(seed success) | Target 2<br>Steps to target<br>(seed success) | Target 3<br>Steps to target<br>(seed success) |
|---|---|---|---|---|
| Ant | SAC-QH$_{(0.1,8)}$ | 1000<br>**175k**$_{(5/5)}$ → 196k$_{(5/5)}$ | 2000<br>385k$_{(5/5)}$ → **378k**$_{(5/5)}$ | 3000<br>653k$_{(5/5)}$ → **580k**$_{(5/5)}$ |
| | TD3-QH$_{(1,2)}$ | 1000<br>438k$_{(4/5)}$ → **417k**$_{(5/5)}$ | 2000<br>569k$_{(4/5)}$ → **518k**$_{(5/5)}$ | 3000<br>615k$_{(3/5)}$ → **547k**$_{(4/5)}$ |
| HalfCheetah | SAC-QH$_{(1,2)}$ | 4000<br>**72k**$_{(5/5)}$ → 76k$_{(5/5)}$ | 8000<br>428k$_{(5/5)}$ → **372k**$_{(5/5)}$ | 10000<br>**774k**$_{(4/5)}$ → 782k$_{(4/5)}$ |
| | TD3-QH$_{(0.1,2)}$ | 4000<br>70k$_{(5/5)}$ → **68k**$_{(5/5)}$ | 8000<br>418k$_{(5/5)}$ → **336k**$_{(5/5)}$ | 10000<br>758k$_{(2/5)}$ → **784k**$_{(4/5)}$ |
| Swimmer | SAC-QH$_{(1,8)}$ | 100<br>166k$_{(2/2)}$ → **90k**$_{(2/2)}$ | 200<br>180k$_{(2/2)}$ → **94k**$_{(2/2)}$ | 300<br>194k$_{(2/2)}$ → **127k**$_{(2/2)}$ |
| | TD3-QH$_{(10,8)}$ | 100<br>81k$_{(2/5)}$ → **75k**$_{(5/5)}$ | 200<br>88k$_{(2/5)}$ → **142k**$_{(5/5)}$ | 300<br>96k$_{(2/5)}$ → **596k**$_{(5/5)}$ |
| Humanoid | SAC-QH$_{(20,8)}$ | 1000<br>**113k**$_{(5/5)}$ → 131k$_{(5/5)}$ | 3000<br>268k$_{(5/5)}$ → **251k**$_{(5/5)}$ | 5000<br>**474k**$_{(5/5)}$ → 518k$_{(5/5)}$ |
| HumanoidStandup | SAC-QH$_{(10,2)}$ | 60000<br>17k$_{(5/5)}$ → 17k$_{(5/5)}$ | 120000<br>**47k**$_{(5/5)}$ → 51k$_{(5/5)}$ | 150000<br>534k$_{(4/5)}$ → **303k**$_{(4/5)}$ |
| MountainCar | DQN-QH$_{(0.1,2)}$ | -150<br>70k$_{(5/5)}$ → 70k$_{(5/5)}$ | -130<br>**74k**$_{(5/5)}$ → 65k$_{(4/5)}$ | -110<br>**90k**$_{(5/5)}$ → 91k$_{(4/5)}$ |
| CartPole | DQN-QH$_{(1,2)}$ | 200<br>17k$_{(5/5)}$ → **5k**$_{(5/5)}$ | 400<br>**29k**$_{(5/5)}$ → 30k$_{(5/5)}$ | 475<br>**29k**$_{(5/5)}$ → 33k$_{(5/5)}$ |
| Acrobot | DQN-QH$_{(0.5,2)}$ | -200<br>**18k**$_{(5/5)}$ → 24k$_{(5/5)}$ | -150<br>**18k**$_{(5/5)}$ → 25k$_{(5/5)}$ | -100<br>**26k**$_{(5/5)}$ → 30k$_{(5/5)}$ |