LIFTED BELLMAN LINEAR PROGRAMMING FOR OFFLINE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline reinforcement learning (RL) typically trains a critic by minimizing a regression loss against bootstrapped value targets, which are stabilized by target networks updated through exponential moving averages (EMA). Furthermore, multi-step targets incorporate behavior-policy actions and therefore require off-policy correction. We instead impose in-sample Bellman optimality on the critic through inequality constraints. We formulate the Lifted Bellman Linear Program (LBLP), which lifts the linear programming characterization of Bellman optimality to the joint (Q, V) space so that every constraint involves only state-action pairs in the dataset. Its unique minimizer is the in-sample optimal pair, and constraints along K-step segments of dataset trajectories leave this minimizer unchanged for any rollout policy and any horizon. Under deterministic dynamics, this minimizer lies between the best return of the dataset and the optimal value. Relaxing the constraints into hinge penalties recovers the same solution above a finite penalty coefficient in the tabular case. Approximate Lifted Bellman Unconstrained Minimization (ALBUM) implements this relaxation with neural networks and detaches the K-step rollout targets by stop gradient. Its objective contains no squared regression onto bootstrapped targets, so it trains without target networks or EMA updates. Under deterministic dynamics, the K-step constraints allow discounted returns along dataset trajectories to serve as lower bounds without off-policy correction or action chunking, providing a direct mechanism for long-horizon value propagation. On OGBench, ALBUM uses a single critic with a Gaussian policy, matches the average performance of FQL, and substantially outperforms ReBRAC, while using the fewest parameters and the least peak GPU memory among all measured methods.

1 INTRODUCTION

Offline reinforcement learning (RL) learns a policy solely from a pre-collected dataset (Levine et al., 2020), and its central obstacle is value overestimation, since the Bellman maximization selects values extrapolated to actions outside the dataset and each backup propagates them to preceding states (Mao et al., 2024). Prior work either keeps policy improvement close to the behavior policy (Fujimoto et al., 2019; Kumar et al., 2019; Fujimoto & Gu, 2021), or modifies the Bellman update by penalizing the values assigned to actions the dataset does not contain (Kumar et al., 2020), or by replacing the maximization with an in-sample regression (Kostrikov et al., 2022; Garg et al., 2023; Xu et al., 2023). This regression brings two costs. The critic is regressed onto a target computed from its own estimate, so a target network with EMA updates is needed to keep the target stable (Mnih et al., 2015; Lillicrap et al., 2015). The same loss also limits how far a single backup reaches. When a multi-step return enters the target, the intermediate actions come from the behavior policy, and the resulting target is biased for the learned policy unless it is corrected (Li et al., 2025). This raises a more basic question. *Can in-sample Bellman optimality be imposed on the critic entirely through inequality constraints over state-action pairs in the dataset, so that the critic is the solution of a constrained program rather than a regression onto bootstrapped targets?*

We approach this question starting from the linear program (LP) characterization of Bellman optimality, in which the optimal value function is the smallest function satisfying a family of Bellman inequalities (De Farias & Van Roy, 2003; Puterman, 2014). However, neither of the two standard LP formulations carries over to offline RL directly. The action-value formulation imposes a Bellman

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.

inequality for every action at the next state, requiring evaluation at actions that the dataset need not contain (Lee et al., 2026). The state-value formulation avoids this issue by eliminating the next action, but identifies only the optimal state-value function, so a policy must be recovered from the dual occupancy measure (Nachum & Dai, 2020). Offline work building on this formulation, therefore, turns to the dual, either by learning a density ratio (Lee et al., 2021) or by solving a saddle-point problem (Zhan et al., 2022; Kamoutsi et al., 2021; Ozdaglar et al., 2023; Gabbianelli et al., 2024).

In this work, we resolve both difficulties with the Lifted Bellman Linear Program (LBLP), which stays on the primal side and lifts the program to the joint (Q, V) space. An epigraph variable V expresses $V(s) \geq \max_a Q(s, a)$ as the family of inequalities $Q(s, a) \leq V(s)$, one for each action, so the Bellman inequality relates Q to V alone and involves no next action. LBLP places positive weight on both Q and V, so its unique minimizer directly determines both Q and V without requiring a dual step. Unlike prior formulations that also use this lifting (Jeong et al., 2025; Mehta & Meyn, 2020; Bas-Serrano et al., 2021; Lu & Meyn, 2023), in which either V is not a decision variable or only one of Q and V carries weight in the objective (Section 2), LBLP uniquely determines the pair through its objective.

In the offline setting, we instantiate LBLP on the empirical Markov decision process (MDP) induced by the dataset. Every constraint is indexed by a dataset state-action pair, including the K-step constraints along dataset trajectories. These additional constraints leave the feasible set and the unique minimizer unchanged. Under deterministic dynamics, this minimizer lies between the best dataset return and the optimal value, and both bounds hold without the coefficient of a value penalty (Kumar et al., 2020). For neural function approximation, we relax the constraints into hinge penalties (Ryu et al., 2019; Sikchi et al., 2024), and the relaxation is exact above a finite penalty coefficient in the tabular case. Approximate Lifted Bellman Unconstrained Minimization (ALBUM) implements this relaxation with neural networks and detaches the K-step targets by stop gradient. Since the ALBUM objective contains no squared regression onto bootstrapped targets, it requires no target networks or EMA updates, and because it imposes maximization only through constraints, it trains with a single critic. Under deterministic dynamics, its multi-step penalties provide lower bounds along trajectories without off-policy correction or action chunking, which gives a direct mechanism for long-horizon value propagation. Using a single Q, a single V, and a Gaussian actor, ALBUM matches the average performance of FQL (Park et al., 2025b) on OGBench (Park et al., 2025a) and substantially outperforms ReBRAC (Tarasov et al., 2023) while using half as many parameters.

Our contributions are as follows.

- We formulate the Lifted Bellman Linear Program (LBLP), a primal LP over the joint (Q, V) space whose constraints involve only state-action pairs in the dataset. With positive objective weights on both Q and V, its unique minimizer is the Bellman-optimal pair of the empirical MDP. Constraints along K-step rollouts are implied by the one-step constraints, so they can be added for any rollout policy and horizon without off-policy correction or action chunking.
- Under deterministic dynamics, we show that this minimizer lies between the best dataset return and the optimal value, and both bounds hold without a penalty coefficient. In an idealized coordinate-wise iteration, the K-step term never slows and can accelerate value propagation (Theorem 4.5).
- We relax LBLP through hinge penalties and show that the relaxation is exact above a finite penalty coefficient in the tabular case. ALBUM implements this relaxation with neural networks. Since its objective contains no squared regression onto bootstrapped targets and imposes maximization only through constraints, ALBUM trains without target networks or EMA updates and with a single critic (Table 2).
- We analyze the update obtained by detaching the K-step targets with stop gradient. The detachment removes γ and K from the rollout term of the necessary condition for bounded updates (Proposition 5.1), and under additional conditions on the dataset, the LBLP solution is a stationary point of this update (Appendix I).
- On OGBench (Park et al., 2025a), ALBUM with single Q and V networks and a Gaussian actor matches the average performance of FQL (Park et al., 2025b) and substantially outperforms ReBRAC (Tarasov et al., 2023), while using the fewest parameters and the least peak GPU memory among all measured methods.

2 Related Works

Offline RL. Offline RL aims to learn a policy from a pre-collected dataset, where the maximization can amplify erroneous value estimates at out-of-distribution actions (Fujimoto et al., 2019; Kumar et al., 2019; Kostrikov et al., 2022; Mao et al., 2024). Existing methods mitigate this problem through policy constraints (Fujimoto et al., 2019; Kumar et al., 2019; Fujimoto & Gu, 2021), pessimistic value regularization (Kumar et al., 2020; Lyu et al., 2022; Mao et al., 2024), in-sample regression (Kostrikov et al., 2022; Garg et al., 2023; Xu et al., 2023), hybrid regularization (Tarasov et al., 2023), and generative actor-critic methods (Wang et al., 2022; Park et al., 2025b).

Multi-step Values. Multi-step and Monte Carlo returns have been used as lower bounds for critic values to facilitate value propagation, as auxiliary terms in a regression objective (He et al., 2017; Oh et al., 2018). Iterated Bellman inequalities have been used in LP-based value-function approximation to obtain tighter underestimators (Wang et al., 2015). In offline RL, multi-step targets are biased for the learned policy unless corrected (Li et al., 2025), and a prevailing approach is action chunking, which conditions the critic on the action sequence (Li et al., 2025; Kim et al., 2026; Li et al., 2026; Song et al., 2026). ALBUM instead imposes K-step returns as inequality constraints of the program that defines a single-action critic, rather than as auxiliary terms alongside a regression loss. Under deterministic dynamics, these constraints require no off-policy correction and leave the solution of the program unchanged, while enabling long-horizon value propagation without action chunking.

Linear Programs for Bellman Optimality. The LP characterization replaces the maximization with Bellman inequalities (De Farias & Van Roy, 2003; Puterman, 2014; Lakshminarayanan et al., 2018). Off-policy and offline work building on this formulation mostly passes to the dual, either estimating density ratios for evaluation or policy optimization (Nachum et al., 2019a; Zhang et al., 2020; Nachum et al., 2019b; Lee et al., 2021; Nachum & Dai, 2020) or solving saddle-point problems (Zhan et al., 2022; Kamoutsi et al., 2021; Ozdaglar et al., 2023; Gabbianelli et al., 2024). Other work introduces variables alongside Q (Mehta & Meyn, 2020; Lu & Meyn, 2023; Jeong et al., 2025), but in these programs V is not a decision variable or carries no weight in the objective. Lee & He (2019); Bas-Serrano et al. (2021) carry Q and V together, but only V is weighted, and Q is fixed by the Bellman equality rather than determined by the objective. Lee et al. (2026) relax the inequalities with a log barrier, but their program involves Q alone and therefore evaluates Q at next actions. Hinge relaxations appear in Ryu et al. (2019), whose hinge retains a maximization in its argument, and in Sikchi et al. (2024), whose Q is fit by regression outside the program. To our knowledge, LBLP is the first primal program for Bellman optimality whose constraints involve only dataset pairs and whose objective determines both Q and V. Two properties follow from this structure. The critic is the solution of a program rather than a regression onto bootstrapped targets, and multi-step constraints along dataset trajectories are implied by the one-step constraints, so they leave the solution unchanged without off-policy correction.

3 Preliminaries

3.1 Markov Decision Processes and Offline Reinforcement Learning

We consider discounted MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ with finite state space $\mathcal{S}$, finite action space $\mathcal{A}$, transition kernel $P(\cdot \mid s, a)$, expected reward $R(s, a)$, and discount factor $0 \le \gamma < 1$. We identify an action-value function Q with a vector in $\mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ indexed by (s, a) and a state-value function V with a vector in $\mathbb{R}^{|\mathcal{S}|}$, and we write $\langle \cdot, \cdot \rangle$ for the Euclidean inner product on either space. The same symbol P denotes the operator this kernel induces, $(PV)(s, a) := \sum_{s' \in \mathcal{S}} P(s' \mid s, a) V(s')$. The realized reward $r(s, a, s')$, whose average over successors is $R(s, a)$, satisfies $r_{\min} \le r(s, a, s') \le r_{\max}$. For a scalar x we write $[x]_+ := \max\{x, 0\}$, and both this operation and inequalities between vectors are applied componentwise. Let $\mathbb{E}_{\mathcal{M},\pi}$ denote the expectation over trajectories $(s_0, a_0, s_1, a_1, \ldots)$ generated by $a_k \sim \pi(\cdot \mid s_k)$ and $s_{k+1} \sim P(\cdot \mid s_k, a_k)$, with $r_k := r(s_k, a_k, s_{k+1})$. The value functions of π are $V^\pi(s) = \mathbb{E}_{\mathcal{M},\pi}[\sum_{k=0}^{\infty} \gamma^k r_k \mid s]$ and $Q^\pi(s, a) = \mathbb{E}_{\mathcal{M},\pi}[\sum_{k=0}^{\infty} \gamma^k r_k \mid s, a]$. Their optima Q^* and V^* are linked by $V^*(s) = \max_{a \in \mathcal{A}} Q^*(s, a)$. In the offline setting the agent is given a fixed dataset $\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^m$ of m transitions collected by a behavior policy β, where $r_i = r(s_i, a_i, s_i')$. Since our algorithm draws contiguous subsequences rather than isolated transitions, we assume $\mathcal{D}$ retains the trajectory structure of the collected data. The dataset also

induces an empirical Markov decision process (Fujimoto et al., 2019), whose definition is given in Appendix C.

3.2 Linear Programming Characterization of Bellman Optimality

For any ρ with strictly positive entries, V^* is the unique solution of the linear program (De Farias & Van Roy, 2003; Puterman, 2014)

$$\min_V \langle \rho, V \rangle \quad \text{subject to} \quad V(s) \geq R(s,a) + \gamma(PV)(s,a), \qquad (s,a) \in \mathcal{S} \times \mathcal{A}. \tag{1}$$

The LP stated over Q involves $Q(s', a')$ for every action a' at the next state s' (Puterman, 2014; Lee et al., 2026).

4 Lifted Bellman Linear Programming

We introduce the *Lifted Bellman Linear Program* (LBLP), which lifts the LP of Section 3.2 from V or Q alone to the pair (Q, V). The lifting is the classical epigraph reformulation (Boyd & Vandenberghe, 2004), under which $V(s) \geq \max_{a \in \mathcal{A}} Q(s,a)$ is equivalent to $Q(s,a) - V(s) \leq 0$ for all $a \in \mathcal{A}$. We state the formulation for an arbitrary finite MDP, since its exactness does not rely on the offline setting, and Section 4.2 instantiates it on the empirical MDP induced by the dataset.

Definition 4.1 (Lifted Bellman Linear Program (LBLP))**.** *Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ be a finite discounted MDP with $0 \leq \gamma < 1$, let β be a behavior policy on $\mathcal{M}$, and let $K > 1$ be a finite integer. The decision variables are Q and V, the weights satisfy $c_Q(s,a) > 0$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ and $c_V(s) > 0$ for all $s \in \mathcal{S}$, and the objective is $F(Q,V) := \langle c_Q, Q \rangle + \langle c_V, V \rangle$. Let us define*

$$y_K(s,a) := \mathbb{E}_{\mathcal{M},\beta}\left[\sum_{t=0}^{K-1} \gamma^t R(s_t, a_t) + \gamma^K V(s_K) \;\middle|\; s, a \right],$$

where the rollout follows $a_t \sim \beta(\cdot \mid s_t)$ for $t \geq 1$. The constraint functions are

$$g_{\mathrm{B}}(Q,V)(s,a) := R(s,a) + \gamma(PV)(s,a) - Q(s,a), \qquad g_{\mathrm{E}}(Q,V)(s,a) := Q(s,a) - V(s),$$
$$g_{\mathrm{KQ}}(Q,V)(s,a) := y_K(s,a) - Q(s,a), \qquad g_{\mathrm{KV}}(Q,V)(s,a) := y_K(s,a) - V(s).$$

The LBLP associated with $(\mathcal{M}, \beta, K)$ is defined as

$$\min_{Q,V} F(Q,V) \quad \textit{s.t.} \quad g_{\mathrm{B}} \leq 0, \quad g_{\mathrm{E}} \leq 0, \quad g_{\mathrm{KQ}} \leq 0, \quad g_{\mathrm{KV}} \leq 0. \tag{2}$$

The four constraints play two distinct roles. The one-step constraints g_{B} and g_{E} together determine the Bellman-optimal pair, whereas g_{KQ} and g_{KV} leave both the feasible set and the solution unchanged, as Section 4.1 shows. We retain them because each directly relates the value at (s,a) to a value K transitions later through a single inequality, and Section 4.3 shows that this can accelerate convergence toward the solution.

4.1 Exactness of LBLP

Lemma 4.2 (Rollout constraints are implied by the one-step constraint)**.** *Let (Q,V) satisfy $g_{\mathrm{B}}(Q,V) \leq 0$ and $g_{\mathrm{E}}(Q,V) \leq 0$. Then $g_{\mathrm{KQ}}(Q,V) \leq 0$ and $g_{\mathrm{KV}}(Q,V) \leq 0$.*

The proof is in Appendix D.1. Note that the implication holds in one direction only. A point satisfying the K-step constraint $(g_{\mathrm{KQ}}(Q,V), g_{\mathrm{KV}}(Q,V))$ need not satisfy the one-step constraint $(g_B(Q,V), g_E(Q,V))$, since the latter demands an inequality at every transition of the rollout while the former demands only one on their aggregate. By Lemma 4.2, the rollout constraints never exclude the Bellman-optimal pair. It remains to show that the one-step constraint forces $Q \geq Q^*$ and $V \geq V^*$ at every feasible point, so that minimizing the objective returns the pair itself.

Proposition 4.3 (Exactness of the LBLP)**.** *Let $\mathcal{M}$ be a finite discounted MDP, β be a policy on $\mathcal{M}$, and $K > 1$ be a finite integer. Then the LBLP of Definition 4.1 has the unique minimizer $(Q,V) = (Q^*, V^*)$.*

The proof is in Appendix D.2. Both results hold for any finite MDP and any rollout policy, so the rollout constraints require no off-policy correction, and they carry over to the offline setting, where $\mathcal{M}$ is unavailable and only the dataset $\mathcal{D}$ is given.

4.2 LBLP ON THE EMPIRICAL MDP OF AN OFFLINE DATASET

The dataset $\mathcal{D}$ induces an empirical MDP $\mathcal{M}_\mathcal{D}$ with pair set $\mathcal{X}_\mathcal{D} = \{(s,a) : (s,a,r,s') \in \mathcal{D}\}$ and action sets $\mathcal{A}_\mathcal{D}(s) = \{a : (s,a) \in \mathcal{X}_\mathcal{D}\}$. Appendix C gives the full construction. Since $\mathcal{M}_\mathcal{D}$ is a finite discounted MDP, Definition 4.1 applies directly. We write $(Q^*_\mathcal{D}, V^*_\mathcal{D})$ for its minimizer, which Proposition 4.3 identifies as the Bellman-optimal pair of $\mathcal{M}_\mathcal{D}$, so that $V^*_\mathcal{D}(s) = \max_{a\in\mathcal{A}_\mathcal{D}(s)} Q^*_\mathcal{D}(s,a)$ with the maximum taken over the actions the data contains at s. Here β is a policy on $\mathcal{M}_\mathcal{D}$, since it selects only actions the data contains, and the program never evaluates an action outside the data, as its constraints live on $\mathcal{X}_\mathcal{D}$ and the maximization ranges over $\mathcal{A}_\mathcal{D}$. For $(s,a) \in \mathcal{X}_\mathcal{D}$, let $\mathcal{H}_\mathcal{D}(s,a)$ be the set of paths obtained by taking a trajectory of $\mathcal{D}$ in which (s,a) appears and following it from that point to its end. Writing $G(\tau) = \sum_{t\geq 0} \gamma^t R_\mathcal{D}(s_t,a_t)$ for the discounted return along $\tau = \big((s_0,a_0),(s_1,a_1),\dots\big)$, we define $G_\mathcal{D}(s,a) := \max_{\tau\in\mathcal{H}_\mathcal{D}(s,a)} G(\tau)$.

Proposition 4.4 (Sandwich property of the LBLP solution)**.** *Suppose the transition dynamics of $\mathcal{M}$ are deterministic, so that $P_\mathcal{D}(\cdot \mid s,a) = P(\cdot \mid s,a)$ and $R_\mathcal{D}(s,a) = R(s,a)$ for every $(s,a) \in \mathcal{X}_\mathcal{D}$, and suppose every trajectory of $\mathcal{D}$ ends at a state s with $\mathcal{A}_\mathcal{D}(s) = \emptyset$ that is terminal in $\mathcal{M}$. Then for every $(s,a) \in \mathcal{X}_\mathcal{D}$, $G_\mathcal{D}(s,a) \leq Q^*_\mathcal{D}(s,a) \leq Q^*(s,a)$, where Q^* denotes the optimal value in $\mathcal{M}$.*

The proof is provided in Appendix D.3. Under full behavioral coverage the left inequality also holds against Q^β, and both bounds carry over to the state-value function (Appendix D.4). Conservative methods such as CQL (Kumar et al., 2020) obtain this upper bound on their fixed point through a penalty coefficient that controls the degree of conservatism, and excessive conservatism can degrade performance (Lyu et al., 2022; Cen et al., 2024). For the LBLP minimizer, both bounds of Proposition 4.4 hold without a coefficient, and the lower bound $G_\mathcal{D}$ depends on $\mathcal{D}$ alone. Figure 1 verifies this on a tabular gridworld where Q^* is available in closed form, solving the LBLP of Definition 4.1 as an exact LP. The minimizer stays inside the admissible interval at every pair of $\mathcal{X}_\mathcal{D}$. Appendix F gives the construction and repeats the measurement across eight seeds.

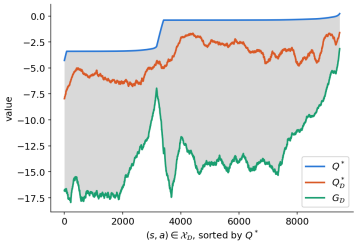

Figure 1: Proposition 4.4 on a tabular gridworld. Each point is a sorted pair in $\mathcal{X}_\mathcal{D}$.

4.3 ACCELERATION THROUGH ROLLOUT CONSTRAINTS

By Lemma 4.2 and Proposition 4.3, the rollout constraints and their horizon can be varied without changing the solution, and we show that such a choice can accelerate convergence toward it. To characterize this effect, we consider an idealized scheme that approaches $(Q^*_\mathcal{D}, V^*_\mathcal{D})$ from below. Assume deterministic dynamics and fix, for each $(s,a) \in \mathcal{X}_\mathcal{D}$, a dataset trajectory starting from (s,a), along which $s^{(k)}(s,a)$ is the state reached after k transitions and $G_k(s,a)$ collects the discounted rewards on the way. For $k \in \{1,K\}$, let $L_k[V](s,a) := G_k(s,a) + \gamma^k V(s^{(k)}(s,a))$ denote the k-step target. Starting from an underestimate $V_0 \leq V^*_\mathcal{D}$, the scheme updates

$$Q_{n+1}(s,a) = \max_{k\in\{1,K\}} L_k[V_n](s,a), \qquad V_{n+1}(s) = \max_{a\in\mathcal{A}_\mathcal{D}(s)} Q_{n+1}(s,a), \tag{3}$$

so that each state settles at the largest $Q_{n+1}(s,a)$ the data offers there. With $K = 1$, equation 3 is value iteration on $\mathcal{M}_\mathcal{D}$, so comparing it with $K > 1$ isolates how the rollout horizon affects value propagation. We define the suboptimality of the realized k-step path and the current value gap as $\delta_k(s,a) := Q^*_\mathcal{D}(s,a) - L_k[V^*_\mathcal{D}](s,a)$ and $e_n := V^*_\mathcal{D} - V_n$, respectively.

Theorem 4.5 (Multi-horizon speedup)**.** *Under the hypotheses of Lemma D.4, the following hold.*

1. *$\|e_{n+1}\|_\infty \leq \gamma\|e_n\|_\infty$ for any rollout length K, so the error decays geometrically at rate γ and the iterates converge to $(Q^*_\mathcal{D}, V^*_\mathcal{D})$. This is the minimizer of LBLP on $\mathcal{M}_\mathcal{D}$ identified in Proposition 4.3.*
2. *The error $\big(Q^*_\mathcal{D} - Q_{n+1}\big)(s,a)$ is a minimum over $k \in \{1,K\}$ and is therefore no larger than its $k = 1$ term, so adding rollout horizon never slows the iteration. At coordinates where the K-step term attains the minimum, the error is at most $\delta_K(s,a) + \gamma^K\|e_n\|_\infty$ rather than $\gamma\|e_n\|_\infty$.*

The proof is provided in Appendix D.6. To verify this acceleration, we ran a numerical simulation on the same gridworld environment used in Section 4.2, with the detailed setting described in Appendix E. The results are reported in Appendix G.

4.4 Exactness of the Relaxed LBLP

LBLP is a constrained program, and enforcing its constraints directly becomes impractical once Q and V are parameterized by neural networks. Replacing the constraints with penalties on their violation turns LBLP into an unconstrained problem, and with hinge penalties this replacement keeps the solution of LBLP once the penalty weights are large enough. We state the relaxation over the same variables as Definition 4.1, with the rollout value given by the expectation y_K, thereby maintaining a direct correspondence with the original program.

Definition 4.6 (Relaxed LBLP)**.** *Let $\mathcal{M}$, β, and K be as in Definition 4.1, with the objective and constraint functions defined there. For penalty weights $\lambda_{\mathrm{B}}, \lambda_{\mathrm{E}}, \lambda_{\mathrm{K}} > 0$, the relaxed LBLP minimizes*

$$\begin{aligned}\mathcal{L}_{\mathrm{R}}(Q,V) := F(Q,V) &+ \lambda_{\mathrm{B}} \left\| [g_{\mathrm{B}}(Q,V)]_+ \right\|_1 + \lambda_{\mathrm{E}} \left\| [g_{\mathrm{E}}(Q,V)]_+ \right\|_1 \\ &+ \lambda_{\mathrm{K}} \left(\left\| [g_{\mathrm{KQ}}(Q,V)]_+ \right\|_1 + \left\| [g_{\mathrm{KV}}(Q,V)]_+ \right\|_1 \right),\end{aligned} \tag{4}$$

Proposition 4.7 (Exactness of the hinge relaxation)**.** *Let $(\lambda^*_{\mathrm{B}}, \lambda^*_{\mathrm{E}}) \geq 0$ be optimal dual variables of the program on $\mathcal{M}_{\mathcal{D}}$ that keeps only the constraints $g_{\mathrm{B}} \leq 0$ and $g_{\mathrm{E}} \leq 0$, and suppose $\lambda_{\mathrm{B}} > \max_{(s,a)\in\mathcal{X}_{\mathcal{D}}} \lambda^*_{\mathrm{B}}(s,a)$ and $\lambda_{\mathrm{E}} > \max_{(s,a)\in\mathcal{X}_{\mathcal{D}}} \lambda^*_{\mathrm{E}}(s,a)$. Then $(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$ is the unique minimizer of equation 4, for every $\lambda_{\mathrm{K}} \geq 0$ and every rollout length K.*

The proof is provided in Appendix D.7. The threshold is finite, so the solution is recovered at a finite setting of the coefficients rather than in a limit. The threshold involves g_{B} and g_{E} alone, so exactness places no condition on the rollout coefficient λ_{K}. The acceleration of Section 4.3 also appears when equation 4 is minimized by gradient descent, as Appendix G shows. Proposition 4.7 relates LBLP to its tabular hinge relaxation and does not enter the analysis of ALBUM.

5 Approximate Lifted Bellman Unconstrained Minimization

ALBUM implements the relaxed LBLP of Definition 4.6 with neural networks. Its critic update detaches the K-step rollout targets by `stop_gradient`. The lifted variables are represented by separate networks Q_θ and V_ϕ. Given a transition sequence $(s_0, a_0, r_0, \ldots, r_{K-1}, s_K)$ from the dataset with $(s,a) = (s_0, a_0)$, let us define $\hat{y}_k(s,a) := \sum_{t=0}^{k-1} \gamma^t r_t + \gamma^k V_\phi(s_k)$ for $k \in \{1, K\}$. The ALBUM critic loss is

$$\begin{aligned}\mathcal{L}(\theta,\phi) = \omega_Q \, &\mathbb{E}_{\mathcal{D}}[Q_\theta(s,a)] + \omega_V \, \mathbb{E}_{\mathcal{D}}[V_\phi(s)] \\ &+ \lambda_{\mathrm{B}} \, \mathbb{E}_{\mathcal{D}}\left[[\hat{y}_1(s,a) - Q_\theta(s,a)]_+ \right] + \lambda_{\mathrm{E}} \, \mathbb{E}_{\mathcal{D}}\left[[Q_\theta(s,a) - V_\phi(s)]_+ \right] \\ &+ \lambda_{\mathrm{K}} \left(\mathbb{E}_{\mathcal{D}}\left[\left[\mathrm{sg}\left[\hat{y}_K(s,a)\right] - Q_\theta(s,a) \right]_+ \right] + \mathbb{E}_{\mathcal{D}}\left[\left[\mathrm{sg}\left[\hat{y}_K(s,a)\right] - V_\phi(s) \right]_+ \right] \right)\end{aligned} \tag{5}$$

where $\mathrm{sg}[\cdot]$ denotes `stop_gradient`, so the rollout target is treated as a constant at each update. The two objective terms are the empirical counterparts of F, with the sampling distribution of $\mathcal{D}$ playing the role of the weights c_Q and c_V, which are positive on every pair of $\mathcal{M}_{\mathcal{D}}$. The sample-based $\hat{y}_K$ replaces the expectation in Definition 4.1 with a single trajectory of $\mathcal{D}$, and under deterministic dynamics the one-step constraints still imply this sampled constraint (Remark D.1).

Unlike a target network in temporal difference regression, which stabilizes a bootstrapped target by delaying its update, the detached target $\mathrm{sg}[\hat{y}_K]$ enters only as a lower bound inside a hinge. Treating $\hat{y}_K$ as a constant removes a gradient term that shared parameters would otherwise introduce into the update of $V_\phi(s)$ and that is absent from the coordinatewise formulation in equation 4 (Appendix H). Thus, the implemented critic update is distinct from the gradient of the scalar objective in equation 5. Under deterministic dynamics and additional conditions on the dataset, Appendix I further shows that $(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$ is a stationary point of the detached update (Proposition I.2).

Since our contribution concerns critic optimization, we extract the policy with a standard objective from the offline RL literature so that the reported gains are attributable to the critic rather than to

policy extraction. We use the DDPG+BC objective (Fujimoto & Gu, 2021; Tarasov et al., 2023; Park et al., 2024),

$$\mathcal{L}_{\text{actor}}(\psi) = -\,\mathbb{E}_{s\sim\mathcal{D}}[Q_\theta(s, \pi_\psi(s))] + \alpha_{\text{BC}}\,\mathbb{E}_{(s,a)\sim\mathcal{D}}\Big[\|\pi_\psi(s) - a\|_2^2\Big]\,. \tag{6}$$

As in other in-sample methods (Kostrikov et al., 2022), the critic is never trained at actions outside the dataset, so its values there rest on function approximation, and the behavioral cloning term keeps the actor's queries near the data.

5.1 A Necessary Condition for Bounded Updates

Proposition 5.1 (Necessary conditions for bounded updates)**.** *Let Q and V in equation 5 be free variables, and consider lowering every entry of Q and V by the same amount.*

(i) The ALBUM update, which treats $\text{sg}[\hat{y}_K]$ as a constant, does not lower Q and V without bound along this direction only if

$$\lambda_{\text{B}}(1-\gamma) + 2\lambda_{\text{K}} \;\geq\; \omega_{\text{Q}} + \omega_{\text{V}}. \tag{7}$$

(ii) The gradient update of equation 5 without stop gradient does not lower Q and V without bound along this direction only if

$$\lambda_{\text{B}}(1-\gamma) + 2\lambda_{\text{K}}(1-\gamma^K) \;\geq\; \omega_{\text{Q}} + \omega_{\text{V}}. \tag{8}$$

The proof is provided in Appendix D.8. ALBUM follows the update in (i), so equation 7 governs the downward drift of its critic, and equation 8 applies to an update that ALBUM does not use. The detached rollout target removes γ and K from the condition. The same condition extends to neural parameterizations whose outputs can be lowered by the same constant at every input, and Appendix L.4 shows that the critic leaves the feasible range when it is violated. The default coefficients in Table 3 give $\lambda_{\text{B}}(1-\gamma) + 2\lambda_{\text{K}} = 2.0125$ against $\omega_{\text{Q}} + \omega_{\text{V}} = 0.15$.

6 Experiments

We evaluate ALBUM on the state-based tasks of OGBench (Park et al., 2025a) and compare it with different policy classes. We report the success rate after 1M gradient steps over 8 seeds, with 50 evaluation episodes per seed. All five critic coefficients in equation 5 are fixed across all domains at the values in Table 3 and are not tuned per environment. Since equation 7 involves neither the reward nor the dataset, these values satisfy it in every domain. Although the coefficients can affect performance, as shown in Appendix L, we select them once on the ablation tasks. For the actor, we follow the tuning protocol of Park et al. (2025b) and tune only the behavioral cloning coefficient α_{BC} in equation 6 on the default task of each environment. This per-environment tuning budget is no larger than that of any baseline, as ALBUM performs no per-environment critic tuning whereas ReBRAC (Tarasov et al., 2023) tunes its critic regularization coefficient per environment (Park et al., 2025b). The ablations rerun FQL and ReBRAC with the hyperparameters in Table 6 of Park et al. (2025b).

Comparison across policy classes. Table 1 compares ALBUM with ten baselines, including Gaussian policies (BC, IQL (Kostrikov et al., 2022), ReBRAC (Tarasov et al., 2023)), diffusion policies (IDQL (Hansen-Estruch et al., 2023), SRPO (Chen et al., 2024), CAC (Ding & Jin, 2024)), and flow policies (FAWAC, FBRAC, IFQL, FQL (Park et al., 2025b)). Their scores are taken from Park et al. (2025b), who tuned every baseline under the fair tuning budget, which allows a direct comparison with ALBUM tuned under the same protocol. ALBUM and FQL achieve the two highest state-based averages in Table 1. ALBUM attains the highest score on `antmaze-large-navigate`, `antmaze-giant-navigate`, `scene-play`, and `puzzle-3x3-play`, by a large margin on the last two, and all four domains require long-horizon value propagation. ALBUM is weakest on `humanoidmaze-large-navigate`. It achieves these results with a Gaussian policy and without a critic ensemble or a target network. Compared with ReBRAC, which uses the same policy class, ALBUM improves the average by 12.4 points and leads on eight of ten domains.

	Gaussian Policies			Diffusion Policies			Flow Policies				Gaussian (Ours)
Task Category	BC	IQL	ReBRAC	IDQL	SRPO	CAC	FAWAC	FBRAC	IFQL	FQL	ALBUM (Ours)
antmaze-large-navigate	10.6	53.4	<u>80.8</u>	20.8	10.6	32.8	6.4	60.2	28.0	78.6	**89.8**±1.0
antmaze-giant-navigate	0.2	4.0	<u>26.2</u>	0.0	0.0	0.0	0.0	3.8	2.6	8.6	**28.1**±5.1
humanoidmaze-medium-navigate	2.0	32.8	21.8	0.8	1.4	52.8	19.4	38.4	**60.4**	<u>57.4</u>	35.5±1.8
humanoidmaze-large-navigate	0.4	2.4	2.6	0.6	0.2	0.6	0.2	2.2	**11.0**	<u>4.2</u>	1.6±0.5
antsoccer-arena-navigate	1.0	8.4	0.0	11.8	1.0	1.8	12.4	16.0	33.2	**60.2**	<u>36.1</u>±3.8
cube-single-play	5.4	83.0	90.6	<u>94.6</u>	79.6	85.2	81.2	78.6	79.2	**95.8**	83.2±1.7
cube-double-play	1.6	6.4	12.2	14.6	1.4	5.8	5.2	<u>15.0</u>	14.0	**28.6**	13.3±2.8
scene-play	4.6	27.6	40.6	46.2	20.0	39.8	29.8	44.8	30.4	<u>55.8</u>	**64.4**±5.5
puzzle-3x3-play	1.8	9.0	21.6	10.4	17.8	19.4	6.4	14.0	19.0	<u>29.6</u>	**61.2**±6.4
puzzle-4x4-play	0.2	7.4	14.0	**29.2**	10.6	14.8	0.4	13.2	<u>25.2</u>	17.2	20.6±2.5
Average (state-based)	2.8	23.4	31.0	22.9	14.3	25.3	16.1	28.6	30.3	**43.6**	**43.4**

Table 1: Success rate (%) on OGBench state-based tasks. ALBUM: mean over 5 tasks and 8 seeds at 1M steps, ± std across seeds. Baselines from Park et al. (2025b). Best in bold, second underlined.

6.1 Ablations

In this section, we examine the design properties induced by the LBLP formulation and test whether the empirical behavior is consistent with the theoretical analysis. Unless otherwise specified, hyperparameters follow Table 3, with task, seed, and evaluation settings specified in Appendix L.

Why does ALBUM remain stable without target networks or critic ensembles? Averaged over the three tasks in Table 2, removing target networks and EMA updates degrades ReBRAC by 16% and FQL by 5%, using a single critic degrades them by 87% and 34%, and removing both degrades them by 90% and 49%. These results are consistent with the LP-derived critic objective, which contains neither moving-target regression nor action maximization. See Appendix L.1 for detailed comparisons.

	antmaze-large-navigate-task1		scene-play-task2		puzzle-3x3-play-task4	
	Success (%)	Δ	Success (%)	Δ	Success (%)	Δ
ALBUM(w/ ensemble & w/o target)	91.8±4.3	–	93.0±3.0	–	93.2±8.4	–
ALBUM(w/o ensemble & target, Ours)	92.5±3.1	+0.7	93.2±3.2	+0.2	93.0±7.9	−0.2
FQL	88.0±3.3	–	84.8±11.5	–	10.8±2.0	–
FQL(w/o ensemble)	54.0±41.5	−34.0	58.5±14.7	−26.2	8.2±2.1	−2.5
FQL(w/o target)	83.5±7.5	−4.5	77.0±8.5	−7.8	14.0±6.6	+3.2
FQL(w/o ensemble & target)	45.8±11.5	−42.2	35.2±6.6	−49.5	12.0±3.9	+1.2
ReBRAC	95.8±3.4	–	52.0±15.5	–	6.5±2.4	–
ReBRAC(w/o ensemble)	7.0±18.5	−88.8	7.8±10.3	−44.2	5.5±5.0	−1.0
ReBRAC(w/o target)	56.0±38.2	−39.8	66.8±11.1	+14.8	7.2±3.0	+0.8
ReBRAC(w/o ensemble & target)	0.0±0.0	−95.8	10.8±10.9	−41.2	4.2±4.4	−2.2

Table 2: Success rate (%) on OGBench state-based default tasks. Mean 8 seeds at 1M steps, ± std across seeds, Δ the gap to the un-ablated method.

How does ALBUM compare with action-chunking methods? Table 6 compares ALBUM with QC-FQL (Li et al., 2025), DEAS (Kim et al., 2026), DQC (Li et al., 2026), CGQ (Song et al., 2026), and NFQL, an n-step variant of FQL evaluated in CGQ. ALBUM attains the highest score on `antmaze-large-navigate` and `antmaze-giant-navigate`, exceeding the best baseline by 10.8 and 16.4 points, respectively. Its average of 37.5 is higher than those of NFQL and QC-FQL and lower than those of FQL, DEAS, CGQ, and DQC. Full results are reported in Appendix L.2.

How much does ALBUM reduce computational cost? We compare the training cost per gradient step of ALBUM with ReBRAC, IQL, FQL, NFQL, QC-FQL, DQC, and CGQ, each in its default configuration (Table 7). These methods hold four to eight value networks, including critic ensembles and target copies, whereas ALBUM holds two. ALBUM also uses 2.43M parameters and 78 MiB of peak memory, the fewest among all measured methods. Compared with FQL, it reduces the

parameters by 50%, the peak memory by 39%, and the time per step by 44%. The reduction is larger against action-chunking methods, reaching 76%, 71%, and 74% against CGQ. With the actor updated at every step for all methods, ALBUM is the fastest, although ReBRAC with its default delayed actor update remains 15% faster. See Appendix L.3.

What happens when the coefficient condition is violated? Proposition 5.1 gives a necessary condition for the update to avoid an unbounded downward drift, and we enforce equation 7 when choosing the penalty coefficients. To verify its necessity in practice, we set $\lambda_{\mathrm{K}} = 0$ on `antmaze-large-navigate`, yielding $\lambda_{\mathrm{B}}(1-\gamma) + 2\lambda_{\mathrm{K}} = 0.013 < 0.15 = \omega_{\mathrm{Q}} + \omega_{\mathrm{V}}$. The resulting $V_{\min}$ is driven to the order of -10^6, far outside the physical range $[-1/(1-\gamma), 1/(1-\gamma)] = [-200, 200]$, confirming the unbounded regime predicted by Proposition 5.1. The task also has near-zero success and only 1% of samples satisfy $g_{\mathrm{B}} \leq 0$. Full results are reported in Appendix L.4.

Does the K-step term behave as predicted? Theorem 4.5 shows that adding the K-step term does not slow the idealized iteration and that, where the K-step term is selected, the error contracts at γ^K. Appendix G confirms both in a tabular setting. Under neural parameterization, a longer horizon accelerates learning on `antmaze-large-navigate`, with $K = 20$ reaching 50% success at 300k steps against 500k for $K = 5$. On `puzzle-3x3-play` a longer horizon instead lowers the final success rate, which the idealized iteration does not predict. Removing the K-step term entirely fails on every task. Full results are reported in Appendix L.5.

Does the coefficient condition alone recover performance? The condition can be satisfied without the K-step constraint by increasing the one-step penalties. We set $\lambda_{\mathrm{K}} = 0$ and increase $\lambda_{\mathrm{B}} = \lambda_{\mathrm{E}}$ well beyond the value $(\omega_Q + \omega_V)/(1-\gamma)$ required by equation 7. The values no longer diverge, but the best $\lambda_{\mathrm{K}} = 0$ setting achieves only 4.2%, 2.0%, and 0.5% success on `antmaze-large-navigate`, `scene-play`, and `puzzle-3x3-play`, respectively, compared with 92.5%, 93.2%, and 93.0% with $\lambda_{\mathrm{K}} = 1$. Thus, the K-step constraint contributes beyond satisfying the condition. Value heatmaps on `antmaze-large-navigate` (Figure 10) indicate faster value propagation with the K-step constraint (Appendix L.6).

Does the ALBUM critic improvement transfer to a generative policy? The main experiments fix a Gaussian policy so that the critic is the only change relative to the baselines of the same policy class in Table 1. To test the critic under an expressive policy, we replace the IQL critic of IFQL (Park et al., 2025b) with ALBUM without any tuning. The five-task average rises from 30.4 to 68.8 on `scene-play` and from 19.0 to 72.3 on `puzzle-3x3-play`, above both FQL and ALBUM in Table 1, and stays near its original value on `antmaze-large-navigate`. The gains concentrate on the tasks IFQL fails to solve, four of which move from 0 to between 54.0 and 97.0. (Appendix L.7).

7 Conclusion

In this work, we show that in-sample Bellman optimality can be imposed on an offline critic through inequality constraints, rather than through regression. LBLP realizes this principle over the joint (Q, V) space. With positive weight on both Q and V, its unique minimizer is the in-sample optimal pair, and K-step constraints along dataset trajectories leave this minimizer unchanged. Under deterministic dynamics, this minimizer lies between the best dataset return and the optimal value. The tabular hinge relaxation of LBLP recovers the same solution above a finite penalty coefficient. ALBUM implements this relaxed formulation with neural networks, and we derive a necessary condition on its coefficients for bounded updates that extends to the neural parameterization. The ALBUM critic contains no squared regression, so it trains without target networks or EMA updates, and under deterministic dynamics its K-step penalties use multi-step dataset returns as lower bounds without off-policy correction or action chunking. With a single Q and V network and a Gaussian actor, ALBUM matches the average performance of FQL on OGBench, substantially outperforms ReBRAC, with the fewest parameters and the least peak GPU memory among all measured methods. Establishing convergence of the neural update and extending the analysis to stochastic transitions remain future work.

AI USE STATEMENT

In this work, we used generative AI tools to assist in formulating and refining some mathematical claims, to suggest proof ingredients and proof sketches for some results, to implement the proposed methods, and to check the proofs as a supplement to the authors' own verification. The theoretical framework, the formulation of LBLP, and the design of the experiments were developed by the authors. We have not used generative AI tools to design research methodology or experiments or to interpret experimental results, and generating synthetic data, cleaning or reformatting datasets, and qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for translation and to edit parts of the manuscript for readability and organization. We have reviewed all AI-assisted work. Every proof was checked by at least two authors. We take responsibility for the final content of this work, including all mathematical claims, proofs, implementations, and experimental results.

REPRODUCIBILITY STATEMENT

The reference ALBUM implementation is provided as supplementary material with pinned dependency versions and is written in JAX. Algorithm 1 summarizes the training procedure, with the critic loss and actor objective given in equation 5 and equation 6, respectively. The hyperparameters shared across all environments are listed in Table 3, the environment-specific behavioral cloning coefficients are reported in Table 4, and the evaluation protocol, including the number of random seeds and evaluation episodes, is described in Section 6. The OGBench datasets are publicly available, and the baseline scores are taken from Park et al. (2025b) and Song et al. (2026). The theoretical results are stated with their assumptions and accompanied by complete proofs in Appendix D. For the Gridworld experiments, we provide the environment configuration, dataset collection procedure, exact solution, and computational procedure to facilitate reproduction in Appendix E.

REFERENCES

Joan Bas-Serrano, Sebastian Curi, Andreas Krause, and Gergely Neu. Logistic Q-learning. In *International conference on artificial intelligence and statistics*, pp. 3610–3618. PMLR, 2021.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Zhepeng Cen, Zuxin Liu, Zitong Wang, Yihang Yao, Henry Lam, and Ding Zhao. Learning from sparse offline datasets via conservative density estimation. *arXiv preprint arXiv:2401.08819*, 2024.

Huayu Chen, Cheng Lu, Zhengyi Wang, Hang Su, and Jun Zhu. Score regularized policy optimization through diffusion behavior. In *International Conference on Learning Representations*, volume 2024, pp. 10211–10230, 2024.

Daniela Pucci De Farias and Benjamin Van Roy. The linear programming approach to approximate dynamic programming. *Operations research*, 51(6):850–865, 2003.

Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 53047–53066, 2024.

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

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1587–1596. PMLR, 2018.

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

Germano Gabbianelli, Gergely Neu, Matteo Papini, and Nneka M Okolo. Offline primal-dual reinforcement learning for linear MDPS. In *International Conference on Artificial Intelligence and Statistics*, pp. 3169–3177. PMLR, 2024.

Divyansh Garg, Joey Hejna, Matthieu Geist, and Stefano Ermon. Extreme Q-learning: Maxent RL without entropy. *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=SJ0Lde3tRL.

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

Hado Hasselt. Double Q-learning. *Advances in neural information processing systems*, 23, 2010.

Frank S He, Yang Liu, Alexander G Schwing, and Jian Peng. Learning to play in a day: Faster deep reinforcement learning by optimality tightening. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rJ8Je4clg.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Narim Jeong, Donghwan Lee, and Niao He. Stochastic primal-dual Q-learning. *arXiv preprint arXiv:1810.08298*, 2025.

Angeliki Kamoutsi, Goran Banjac, and John Lygeros. Efficient performance bounds for primal-dual reinforcement learning from demonstrations. In *International Conference on Machine Learning*, pp. 5257–5268. PMLR, 2021.

Changyeon Kim, Haeone Lee, Younggyo Seo, Kimin Lee, and Yuke Zhu. DEAS: Detached value learning with action sequence for scalable offline RL. In *International Conference on Learning Representations*, volume 2026, pp. 64035–64058, 2026.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy Q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Chandrashekar Lakshminarayanan, Shalabh Bhatnagar, and Csaba Szepesvári. A linearly relaxed approximate linear program for Markov decision processes. *IEEE Transactions on Automatic Control*, 63(4):1185–1191, 2018.

Donghwan Lee and Niao He. Stochastic primal-dual Q-learning algorithm for discounted mdps. In *2019 american control conference (acc)*, pp. 4897–4902. IEEE, 2019.

Donghwan Lee, Hyukjun Yang, and Bum Geun Park. Analysis of approximate linear programming solution to Markov decision problem with log barrier function. In *International Conference on Learning Representations*, 2026.

Jongmin Lee, Wonseok Jeon, Byungjun Lee, Joelle Pineau, and Kee-Eung Kim. OptiDICE: Offline policy optimization via stationary distribution correction estimation. In *International Conference on Machine Learning*, pp. 6120–6130. PMLR, 2021.

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

Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement learning with action chunking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=XUks1Y96NR.

Qiyang Li, Seohong Park, and Sergey Levine. Decoupled Q-chunking. In *International Conference on Learning Representations*, volume 2026, pp. 145315–145389, 2026.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Fan Lu and Sean P Meyn. Convex Q learning in a stochastic environment. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pp. 776–781. IEEE, 2023.

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

Yixiu Mao, Qi Wang, Yun Qu, Yuhang Jiang, and Xiangyang Ji. Doubly mild generalization for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:51436–51473, 2024.

Prashant G Mehta and Sean P Meyn. Convex Q-learning, part 1: Deterministic optimal control. *arXiv preprint arXiv:2008.03559*, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Ofir Nachum and Bo Dai. Reinforcement learning via Fenchel-Rockafellar duality. *arXiv preprint arXiv:2001.01866*, 2020.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. DualDICE: Behavior-agnostic estimation of discounted stationary distribution corrections. *Advances in neural information processing systems*, 32, 2019a.

Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. AlgaeDICE: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019b.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *International conference on machine learning*, pp. 3878–3887. PMLR, 2018.

Asuman E Ozdaglar, Sarath Pattathil, Jiawei Zhang, and Kaiqing Zhang. Revisiting the linear-programming framework for offline RL with general function approximation. In *International Conference on Machine Learning*, pp. 26769–26791. PMLR, 2023.

Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline RL? *Advances in Neural Information Processing Systems*, 37:79029–79056, 2024.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/ecd92623ac899357312aaa8915853699-Paper-Conference.pdf.

Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-learning. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=KVf2SFL1pi&noteId=Iyc15gj8hQ.

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

R Tyrrell Rockafellar. *Convex analysis*, volume 28. Princeton university press, 1997.

Moonkyung Ryu, Yinlam Chow, Ross Anderson, Christian Tjandraatmadja, and Craig Boutilier. CAQL: Continuous action Q-learning. *arXiv preprint arXiv:1909.12397*, 2019.

Harshit Sikchi, Qinqing Zheng, Amy Zhang, and Scott Niekum. Dual RL: Unification and new methods for reinforcement and imitation learning. In *International Conference on Learning Representations*, volume 2024, pp. 9305–9352, 2024.

Gwanwoo Song, Kwanyoung Park, and Youngwoon Lee. Chunk-guided Q-learning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=yojuz84NLb.

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

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Yang Wang, Brendan O'Donoghue, and Stephen Boyd. Approximate dynamic programming via iterated Bellman inequalities. *International Journal of Robust and Nonlinear Control*, 25(10): 1472–1496, 2015.

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

Chenjun Xiao, Han Wang, Yangchen Pan, Adam White, and Martha White. The in-sample softmax for offline reinforcement learning. *arXiv preprint arXiv:2302.14372*, 2023.

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline RL with no OOD actions: In-sample learning via implicit value regularization. *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ueYYgo2pSSU.

Wenhao Zhan, Baihe Huang, Audrey Huang, Nan Jiang, and Jason Lee. Offline reinforcement learning with realizability and single-policy concentrability. In *Conference on Learning Theory*, pp. 2730–2775. PMLR, 2022.

Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. GenDICE: Generalized offline estimation of stationary values. *arXiv preprint arXiv:2002.09072*, 2020.

A LIMITATIONS

Lemma 4.2, Proposition 4.3, Proposition 4.7, and Proposition 5.1 hold for any finite discounted MDP, and the remaining theoretical results assume deterministic transitions. The assumption enters at three points. Proposition 4.4 identifies $\mathcal{M}_\mathcal{D}$ with $\mathcal{M}$ on $\mathcal{X}_\mathcal{D}$, Lemma D.4 fixes one dataset trajectory per pair, and Remark D.1 replaces the expectation y_K by a single trajectory. The last one carries the sharpest consequence. Under stochastic transitions a sampled path is not a lower bound on $Q^*_\mathcal{D}$, so $\hat{y}_K$ can exceed it and the K-step hinges no longer provide lower bounds.

Under continuous states, each state of $\mathcal{D}$ appears once almost surely, so $|\mathcal{A}_\mathcal{D}(s)| = 1$ for every $s \in \mathcal{S}_\mathcal{D}$. The interval in Proposition 4.4 collapses because $G_\mathcal{D} = Q^*_\mathcal{D}$, and the LBLP solution reduces to the discounted return along each dataset trajectory. The same condition gives $\delta_K = 0$ on $\mathcal{X}_\mathcal{D}$, so the stationary point of Corollary I.5 is this trajectory-wise return. Proposition 4.3, Proposition 4.4, and Corollary I.5 therefore characterize the in-sample solution within $\mathcal{M}_\mathcal{D}$. The collapse follows from $|\mathcal{A}_\mathcal{D}(s)| = 1$ alone and is independent of the LP formulation, so the same collapse applies to methods that explicitly target an in-sample optimum (Kostrikov et al., 2022; Garg et al., 2023; Xu et al., 2023).

The results divide into two groups according to whether the neural parameterization affects them. Proposition 4.3, Proposition 4.7, and Proposition 4.4 characterize the in-sample solution underlying the critic loss in equation 5. These results describe the solution of the program on $\mathcal{M}_\mathcal{D}$ rather than the values the neural critic attains. The mean of the learned critic exceeds the mean of $V^*_\mathcal{D}$ in nine of the ten environments (Appendix L.11). The remaining properties concern the structure of the implemented loss and update. Every critic term in equation 5 is evaluated at a pair in $\mathcal{D}$, so critic training never evaluates Q_θ at an action absent from the data. The actor objective in equation 6 evaluates Q_θ at actions outside $\mathcal{D}$, and the behavioral cloning term restricts these evaluations to a neighborhood of the data. Under deterministic dynamics, the one-step hinges imply the sampled K-step hinges (Remark D.1), so the K-step term changes neither the constraint set nor the in-sample solution targeted by the loss, for any K. Proposition 5.1 concerns the update along a common shift of all outputs, which any network with a bias in its final layer admits, so the condition equation 7 governs the neural update directly, and Appendix L.4 confirms the predicted divergence when it is violated. A full characterization of the learned values under neural parameterization remains open.

Theorem 4.5 concerns an idealized coordinatewise iteration rather than the ALBUM update, and Appendix I establishes the existence of a zero of the detached update rather than uniqueness or convergence. The coefficient conditions separate divergence from boundedness and do not separate the settings that perform well from those that do not, as $\lambda_\mathrm{B} = 10$ satisfies them while failing on every task (Appendix L.8). Extending the analysis to stochastic transitions and to an empirical MDP with branching action sets remains future work.

B ALGORITHM PSEUDOCODE

Algorithm 1 Approximate Lifted Bellman Unconstrained Minimization (ALBUM)

Require: Offline dataset $\mathcal{D}$, mini-batch size B, rollout length K, discount factor γ
Require: Objective weights ω_Q, ω_V, penalty weights $\lambda_{\mathrm{B}}, \lambda_{\mathrm{E}}, \lambda_{\mathrm{K}}$, BC weight α_{BC}
Require: Learning rates $\eta_{\mathrm{critic}}, \eta_{\mathrm{actor}}$
1: Initialize Q_θ, V_ϕ, and π_ψ
2: **while** not converged **do**
3: Sample B subsequences of length K from $\mathcal{D}$
4: Compute $\hat{y}_1$ and $\hat{y}_K$.
5: Compute $\mathcal{L}(\theta, \phi)$ using Eq. 5
6: $\theta \leftarrow \theta - \eta_{\mathrm{critic}} \nabla_\theta \mathcal{L}$, $\phi \leftarrow \phi - \eta_{\mathrm{critic}} \nabla_\phi \mathcal{L}$
7: $\psi \leftarrow \psi - \eta_{\mathrm{actor}} \nabla_\psi \mathcal{L}_{\mathrm{actor}}$ using Eq. 6
8: **end while**
9: **return** $Q_\theta, V_\phi, \pi_\psi$

C OFFLINE DATASET AND EMPIRICAL MDP

To discuss LBLP in the offline setting, we define the empirical MDP induced by the dataset, following Fujimoto et al. (2019); Xiao et al. (2023), where we adopt our own notation to explain the construction more clearly. Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ be the underlying MDP and $\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^m$ be a fixed offline dataset of m transitions collected from $\mathcal{M}$; every object below is computed from $\mathcal{D}$ alone, with no further interaction with $\mathcal{M}$. Writing $n_{\mathcal{D}}(s,a) := \sum_{i=1}^m \mathbf{1}\{(s_i, a_i) = (s,a)\}$ for the number of times a pair appears in the data, the pairs seen at least once form the empirical support

$$\mathcal{X}_{\mathcal{D}} := \{(s,a) : n_{\mathcal{D}}(s,a) > 0\},$$

with state marginal and action slices

$$\mathcal{S}_{\mathcal{D}} = \{s : \exists a,\, (s,a) \in \mathcal{X}_{\mathcal{D}}\}, \qquad \mathcal{A}_{\mathcal{D}}(s) = \{a : (s,a) \in \mathcal{X}_{\mathcal{D}}\},$$

so that $\mathcal{X}_{\mathcal{D}} = \{(s,a) : s \in \mathcal{S}_{\mathcal{D}},\, a \in \mathcal{A}_{\mathcal{D}}(s)\}$. All value functions below are vectors in $\mathbb{R}^{|\mathcal{X}_{\mathcal{D}}|}$, i.e. they are defined only on pairs the dataset has actually seen. Normalizing the counts gives the empirical sampling distribution $d_{\mathcal{D}}(s,a) := n_{\mathcal{D}}(s,a)/m$, a probability distribution supported exactly on $\mathcal{X}_{\mathcal{D}}$, which records how strongly each pair is weighted in the empirical objective; we write $d_{\min} := \min_{\mathcal{X}_{\mathcal{D}}} d_{\mathcal{D}}$, $d_{\max} := \max_{\mathcal{X}_{\mathcal{D}}} d_{\mathcal{D}}$, and $D_{\mathcal{D}} := \mathrm{diag}(d_{\mathcal{D}}(s,a))_{(s,a)\in\mathcal{X}_{\mathcal{D}}}$. Averaging the transitions that start at each $(s,a) \in \mathcal{X}_{\mathcal{D}}$ yields the empirical reward and transition kernel

$$R_{\mathcal{D}}(s,a) := \frac{1}{n_{\mathcal{D}}(s,a)} \sum_{i:(s_i,a_i)=(s,a)} r_i, \qquad P_{\mathcal{D}}(s' \mid s,a) := \frac{1}{n_{\mathcal{D}}(s,a)} \sum_{i:(s_i,a_i)=(s,a)} \mathbf{1}\{s_i' = s'\}.$$

The successors at which no action of $\mathcal{D}$ appears form $\mathcal{S}_{\mathcal{D}}^0 := \{s_i' : i \in [m]\} \setminus \mathcal{S}_{\mathcal{D}}$, the states that the data reaches but never leaves. We treat such states as absorbing with zero reward, so that $\mathcal{A}_{\mathcal{D}}(s) = \emptyset$ and

$$\max_{a' \in \mathcal{A}_{\mathcal{D}}(s)} Q(s,a') := 0 \quad \text{whenever } \mathcal{A}_{\mathcal{D}}(s) = \emptyset, \qquad \text{equivalently } V(s) = 0. \tag{9}$$

The resulting empirical MDP is $\mathcal{M}_{\mathcal{D}} = (\mathcal{S}_{\mathcal{D}} \cup \mathcal{S}_{\mathcal{D}}^0, \{\mathcal{A}_{\mathcal{D}}(s)\}_s, P_{\mathcal{D}}, R_{\mathcal{D}}, \gamma)$, and its Bellman optimality operator $\mathcal{T}_{\mathcal{D}} : \mathbb{R}^{|\mathcal{X}_{\mathcal{D}}|} \to \mathbb{R}^{|\mathcal{X}_{\mathcal{D}}|}$ is

$$(\mathcal{T}_{\mathcal{D}} Q)(s,a) = R_{\mathcal{D}}(s,a) + \gamma \mathbb{E}_{s' \sim P_{\mathcal{D}}(\cdot|s,a)} \left[\max_{a' \in \mathcal{A}_{\mathcal{D}}(s')} Q(s',a') \right], \qquad (s,a) \in \mathcal{X}_{\mathcal{D}},$$

with equation 9 applied at terminal successors. This is the ordinary Bellman optimality operator subject to two restrictions, in that the transition kernel and reward are the empirical ones and the inner maximization ranges over the actions the dataset contains.

This definition is well posed because $\mathcal{M}_{\mathcal{D}}$ is closed under its own transitions. For every $(s,a) \in \mathcal{X}_{\mathcal{D}}$ and every s' with $P_{\mathcal{D}}(s' \mid s,a) > 0$, the successor s' lies in $\mathcal{S}_{\mathcal{D}}$ or in $\mathcal{S}_{\mathcal{D}}^0$. Indeed, a positive empirical

transition probability means the transition $(s, a) \to s'$ appears in $\mathcal{D}$, so s' either carries an action of its own or carries none at all. In the first case the inner maximum is taken over the nonempty set $\mathcal{A}_{\mathcal{D}}(s')$ and reads only entries of Q indexed by $\mathcal{X}_{\mathcal{D}}$, and in the second it equals 0 by equation 9. Hence $\mathcal{T}_{\mathcal{D}}$ never queries Q outside $\mathcal{X}_{\mathcal{D}}$, it is a γ-contraction in $\|\cdot\|_\infty$, and it admits a unique fixed point $Q^*_{\mathcal{D}}$. This fixed point is not a new object, as prior work states the same in-sample Bellman optimality equation (Xiao et al., 2023).

D PROOFS

Lemma 4.2, Proposition 4.3, Proposition 4.7, and Proposition 5.1 hold for any finite discounted MDP, and the remaining results assume deterministic dynamics.

D.1 PROOF OF LEMMA 4.2

Proof. Fix $(s, a) \in \mathcal{S} \times \mathcal{A}$. Consider a β-rollout with $(s_0, a_0) = (s, a)$, so that $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ and $a_{t+1} \sim \beta(\cdot \mid s_{t+1})$ for $t \geq 0$. From $g_{\mathrm{E}} \leq 0$ we have $V(s_{t+1}) \geq Q(s_{t+1}, a_{t+1})$ for every a_{t+1}, and averaging over $a_{t+1} \sim \beta(\cdot \mid s_{t+1})$ preserves the inequality,

$$V(s_{t+1}) \;\geq\; \mathbb{E}_{a_{t+1}\sim\beta(\cdot|s_{t+1})}[Q(s_{t+1}, a_{t+1})]\,. \tag{10}$$

Combining $g_{\mathrm{B}} \leq 0$ with equation 10, we have

$$\begin{aligned} Q(s_t, a_t) \;&\geq\; R(s_t, a_t) + \gamma\, \mathbb{E}_{\mathcal{M}}[V(s_{t+1}) \mid s_t, a_t] \\ &\geq\; R(s_t, a_t) + \gamma\, \mathbb{E}_{\mathcal{M},\beta}[Q(s_{t+1}, a_{t+1}) \mid s_t, a_t]\,. \end{aligned} \tag{11}$$

Applying equation 11 at $t = 0$ and then taking the expectation of equation 11 at $t = 1, \ldots, K-2$ under $\mathbb{E}_{\mathcal{M},\beta}[\,\cdot \mid s, a]$, and finally $g_{\mathrm{B}} \leq 0$ at $t = K-1$,

$$Q(s, a) \;\geq\; R(s, a) + \gamma\, \mathbb{E}_{\mathcal{M},\beta}[Q(s_1, a_1) \mid s, a] \tag{12}$$

$$\geq\; \mathbb{E}_{\mathcal{M},\beta}\left[\sum_{t=0}^{1} \gamma^t R(s_t, a_t) + \gamma^2 Q(s_2, a_2) \;\middle|\; s, a\right] \tag{13}$$

$$\vdots$$

$$\geq\; \mathbb{E}_{\mathcal{M},\beta}\left[\sum_{t=0}^{K-2} \gamma^t R(s_t, a_t) + \gamma^{K-1} Q(s_{K-1}, a_{K-1}) \;\middle|\; s, a\right] \tag{14}$$

$$\geq\; \mathbb{E}_{\mathcal{M},\beta}\left[\sum_{t=0}^{K-1} \gamma^t R(s_t, a_t) + \gamma^K V(s_K) \;\middle|\; s, a\right] \;=\; y_K(s, a), \tag{15}$$

which implies $g_{\mathrm{KQ}}(Q, V)(s, a) \leq 0$.

Applying $g_{\mathrm{E}} \leq 0$ at (s, a) we have

$$V(s) \;\geq\; Q(s, a) \;\geq\; y_K(s, a), \tag{16}$$

which implies $g_{\mathrm{KV}}(Q, V)(s, a) \leq 0$. □

Remark D.1. *Suppose the dynamics are deterministic and let $(s_0, a_0), \ldots, (s_{K-1}, a_{K-1}), s_K$ be a segment of a dataset trajectory, so that every (s_t, a_t) lies in $\mathcal{X}_{\mathcal{D}}$ and s_{t+1} is the successor of (s_t, a_t). If (Q, V) satisfies $g_{\mathrm{B}} \leq 0$ and $g_{\mathrm{E}} \leq 0$ on $\mathcal{M}_{\mathcal{D}}$, then*

$$Q(s_t, a_t) \geq r_t + \gamma V(s_{t+1}) \geq r_t + \gamma Q(s_{t+1}, a_{t+1}),$$

and chaining these inequalities as in equation 12–equation 15 gives $\sum_{t<K} \gamma^t r_t + \gamma^K V(s_K) \leq Q(s_0, a_0) \leq V(s_0)$. Hence the sampled rollout constraints are implied by the one-step constraints and leave the solution unchanged.

D.2 Proof of Proposition 4.3

Proof. We first check that (Q^*, V^*) is feasible. Bellman optimality gives $Q^* = R + \gamma PV^*$, so $g_{\mathrm{B}}(Q^*, V^*) = 0$, and $V^*(s) = \max_a Q^*(s,a) \geq Q^*(s,a)$ gives $g_{\mathrm{E}}(Q^*, V^*) \leq 0$. The remaining constraints $g_{\mathrm{KQ}}(Q^*, V^*) \leq 0$ and $g_{\mathrm{KV}}(Q^*, V^*) \leq 0$ hold automatically by Lemma 4.2. Hence the (Q^*, V^*) is feasible.

Now we check that (Q^*, V^*) is the unique minimizer. Let (Q, V) be an arbitrary feasible point. From $g_B(Q, V) \leq 0$ and $g_E(Q, V) \leq 0$, we have

$$V(s) \geq Q(s,a) \geq R(s,a) + \gamma(PV)(s,a), \qquad \forall (s,a) \in \mathcal{S} \times \mathcal{A}.$$

Taking the maximum over a on the right, we have $V \geq \mathcal{B}V$ where the Bellman optimality operator on state-value functions is defined as $(\mathcal{B}V)(s) := \max_{a \in \mathcal{A}}[R(s,a) + \gamma(PV)(s,a)]$. Iterating this inequality and using that $\mathcal{B}$ is monotone and a γ-contraction, one gets $V \geq V^*$ (Puterman, 2014). Moreover, by using the monotonicity of P, we get $Q \geq R + \gamma PV \geq R + \gamma PV^* = Q^*$. Therefore,

$$V(s) \geq V^*(s) \quad \forall s \in \mathcal{S}, \qquad\qquad Q(s,a) \geq Q^*(s,a) \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}.$$

Since $c_Q, c_V > 0$,

$$F(Q,V) - F(Q^*, V^*) = \langle c_Q, Q - Q^* \rangle + \langle c_V, V - V^* \rangle \geq 0,$$

which implies that equality holds if and only if $(Q, V) = (Q^*, V^*)$. Hence (Q^*, V^*) is the unique minimizer. □

D.3 Proof of Proposition 4.4

Throughout, write s' for the successor of $(s,a) \in \mathcal{X}_\mathcal{D}$, which is unique by determinism and agrees with the successor stored in $\mathcal{D}$. By Proposition 4.3 the minimizer $(Q^*_\mathcal{D}, V^*_\mathcal{D})$ is the Bellman-optimal pair of $\mathcal{M}_\mathcal{D}$, so it satisfies

$$Q^*_\mathcal{D}(s,a) = R_\mathcal{D}(s,a) + \gamma V^*_\mathcal{D}(s'), \qquad V^*_\mathcal{D}(s) = \max_{a \in \mathcal{A}_\mathcal{D}(s)} Q^*_\mathcal{D}(s,a), \tag{17}$$

for every $(s,a) \in \mathcal{X}_\mathcal{D}$, with the convention $V^*_\mathcal{D}(s) = 0$ at states carrying no action of $\mathcal{A}_\mathcal{D}$. By hypothesis such states are terminal in $\mathcal{M}$, so V^* vanishes there.

Lower bound. Fix $(s,a) \in \mathcal{X}_\mathcal{D}$ and let $(s_0,a_0),(s_1,a_1),\ldots$ be a continuation of (s,a), with $r_t = R_\mathcal{D}(s_t,a_t)$. Every pair of the continuation lies in $\mathcal{X}_\mathcal{D}$ and $a_{t+1} \in \mathcal{A}_\mathcal{D}(s_{t+1})$, so the second identity of equation 17 gives $V^*_\mathcal{D}(s_{t+1}) \geq Q^*_\mathcal{D}(s_{t+1}, a_{t+1})$. Combining this with the first identity,

$$Q^*_\mathcal{D}(s_t,a_t) \;\geq\; r_t + \gamma\, Q^*_\mathcal{D}(s_{t+1}, a_{t+1}).$$

Let s_T be the final state of the continuation. By hypothesis $\mathcal{A}_\mathcal{D}(s_T) = \emptyset$, so $V^*_\mathcal{D}(s_T) = 0$ and $Q^*_\mathcal{D}(s_{T-1}, a_{T-1}) = r_{T-1}$. Unrolling the inequality for $t = 0, \ldots, T-2$ yields

$$Q^*_\mathcal{D}(s,a) \geq \sum_{t<T} \gamma^t r_t = G(\tau),$$

and taking the maximum over $\tau \in \mathcal{H}_\mathcal{D}(s,a)$ gives $G_\mathcal{D}(s,a) \leq Q^*_\mathcal{D}(s,a)$.

Upper bound. Let $\mathcal{T}_\mathcal{D}$ be the Bellman optimality operator of $\mathcal{M}_\mathcal{D}$ and let $Q^*\big|_{\mathcal{X}_\mathcal{D}}$ denote the restriction of Q^* to $\mathcal{X}_\mathcal{D}$. For $(s,a) \in \mathcal{X}_\mathcal{D}$, determinism and the agreement of $R_\mathcal{D}$ and $P_\mathcal{D}$ with R and P on $\mathcal{X}_\mathcal{D}$ give

$$\big(\mathcal{T}_\mathcal{D} Q^*\big|_{\mathcal{X}_\mathcal{D}}\big)(s,a) = R(s,a) + \gamma \max_{a' \in \mathcal{A}_\mathcal{D}(s')} Q^*(s',a') \;\leq\; R(s,a) + \gamma \max_{a' \in \mathcal{A}} Q^*(s',a') = Q^*(s,a),$$

where the inequality uses $\mathcal{A}_\mathcal{D}(s') \subseteq \mathcal{A}$, and holds at a terminal successor as well, since both sides are then $R(s,a)$. Thus $Q^*\big|_{\mathcal{X}_\mathcal{D}}$ satisfies $\mathcal{T}_\mathcal{D} Q^*\big|_{\mathcal{X}_\mathcal{D}} \leq Q^*\big|_{\mathcal{X}_\mathcal{D}}$. Since $\mathcal{T}_\mathcal{D}$ is monotone, iterating preserves the inequality, and since it is a γ-contraction with fixed point $Q^*_\mathcal{D}$, the iterates converge to $Q^*_\mathcal{D}$. Hence $Q^*_\mathcal{D} \leq Q^*$ on $\mathcal{X}_\mathcal{D}$, which is the right inequality.

D.4 VARIANTS OF THE SANDWICH PROPERTY

This section collects two variants of Proposition 4.4 that were stated informally in Section 4.2. The first replaces the recorded return by the value of the behavior policy under a coverage assumption, and the second transfers both bounds to the state-value function V.

Corollary D.2 (Sandwich property under full behavioral coverage)**.** *Assume in addition that* $\operatorname{supp}\beta(\cdot \mid s) \subseteq \mathcal{A}_{\mathcal{D}}(s)$ *for every* $s \in \mathcal{S}_{\mathcal{D}}$*, so that* β *is a policy of* $\mathcal{M}_{\mathcal{D}}$*. Then for every* $(s,a) \in \mathcal{X}_{\mathcal{D}}$*,*

$$Q^{\beta}(s,a) \;\le\; Q^{*}_{\mathcal{D}}(s,a) \;\le\; Q^{*}(s,a),$$

where Q^{β} *denotes the value of the behavior policy in* $\mathcal{M}$*.*

Corollary D.3 (Sandwich property for the state-value function)**.** *Under the assumptions of Proposition 4.4, for every* $s \in \mathcal{S}_{\mathcal{D}}$ *we have* $\max_{a\in\mathcal{A}_{\mathcal{D}}(s)} G_{\mathcal{D}}(s,a) \le V^{*}_{\mathcal{D}}(s) \le V^{*}(s)$*, and under the additional assumption of Corollary D.2 we have* $V^{\beta}(s) \le V^{*}_{\mathcal{D}}(s) \le V^{*}(s)$*.*

D.5 PROPAGATION IDENTITY

The following lemma is used in the proof of Theorem 4.5.

Lemma D.4 (Propagation identity)**.** *Suppose the transition dynamics of* $\mathcal{M}$ *are deterministic. Under the empirical MDP* $\mathcal{M}_{\mathcal{D}}$*, fix, for each* $(s,a) \in \mathcal{X}_{\mathcal{D}}$*, a recorded trajectory starting from* (s,a)*, and let* $s^{(k)}(s,a)$ *be the state reached after* k *transitions along this trajectory while* $G_k(s,a) = \sum_{t<k} \gamma^t r_t$ *collects the discounted rewards along the way. The statement below holds for any such choice. Let us define the horizon-*k *target*

$$L_k[V](s,a) := G_k(s,a) + \gamma^k V\big(s^{(k)}(s,a)\big), \tag{18}$$

and the block coordinate updates

$$Q_{n+1} := \max_{k\in\{1,K\}} L_k[V_n], \qquad V_{n+1}(s) := \max_{a\in\mathcal{A}_{\mathcal{D}}(s)} Q_{n+1}(s,a). \tag{19}$$

Moreover, define the suboptimality of the realized k*-step path and the current value gap*

$$\delta_k(s,a) := Q^{*}_{\mathcal{D}}(s,a) - L_k[V^{*}_{\mathcal{D}}](s,a), \qquad e_n := V^{*}_{\mathcal{D}} - V_n. \tag{20}$$

Then, $\delta_K \ge 0$ *with* $\delta_1 = 0$*. Moreover, if* $Q_0 \le Q^{*}_{\mathcal{D}}$ *and* $V_0 \le V^{*}_{\mathcal{D}}$*, the iterates remain underestimates,* $Q_n \le Q^{*}_{\mathcal{D}}$ *and* $V_n \le V^{*}_{\mathcal{D}}$ *for all* n*, so that* $e_n \ge 0$*. In addition, for every* $(s,a) \in \mathcal{X}_{\mathcal{D}}$*,*

$$\big(Q^{*}_{\mathcal{D}} - Q_{n+1}\big)(s,a) = \min_{k\in\{1,K\}} \Big[\delta_k(s,a) + \gamma^k e_n\big(s^{(k)}(s,a)\big)\Big]. \tag{21}$$

Proof. We first show $\delta_k \ge 0$ with equality at $k = 1$. Write $(s_0,a_0) = (s,a)$ and let $s_1, a_1, \dots$ be the rest of the recorded trajectory so that $s_k = s^{(k)}(s,a)$. By Proposition 4.3 applied to the finite discounted MDP $\mathcal{M}_{\mathcal{D}}$, $(Q^{*}_{\mathcal{D}}, V^{*}_{\mathcal{D}})$ is the Bellman-optimal pair of $\mathcal{M}_{\mathcal{D}}$, so

$$Q^{*}_{\mathcal{D}}(s_t,a_t) = r_t + \gamma V^{*}_{\mathcal{D}}(s_{t+1}), \qquad V^{*}_{\mathcal{D}}(s_t) = \max_{a\in\mathcal{A}_{\mathcal{D}}(s_t)} Q^{*}_{\mathcal{D}}(s_t,a) \ge Q^{*}_{\mathcal{D}}(s_t,a_t),$$

where the first because deterministic dynamics give (s_t,a_t) a unique next state, and the second with equality exactly when the recorded action a_t is optimal at s_t. Alternating the two,

$$\begin{aligned} Q^{*}_{\mathcal{D}}(s_0,a_0) &= r_0 + \gamma V^{*}_{\mathcal{D}}(s_1) \\ &\ge r_0 + \gamma Q^{*}_{\mathcal{D}}(s_1,a_1) \\ &= r_0 + \gamma r_1 + \gamma^2 V^{*}_{\mathcal{D}}(s_2) \\ &\ge \cdots \ge \sum_{t=0}^{k-1} \gamma^t r_t + \gamma^k V^{*}_{\mathcal{D}}(s_k) = L_k[V^{*}_{\mathcal{D}}](s,a), \end{aligned}$$

so $\delta_k \ge 0$ for $k > 1$. For $k = 1$, the first line is an equality, and $\delta_1 = 0$.

For the second claim, expand $L_k[V_n]$ and substitute the definitions of e_n and δ_k:

$$\begin{aligned} L_k[V_n](s,a) &= G_k(s,a) + \gamma^k V_n\big(s^{(k)}(s,a)\big) \\ &= G_k(s,a) + \gamma^k V^{*}_{\mathcal{D}}\big(s^{(k)}(s,a)\big) - \gamma^k e_n\big(s^{(k)}(s,a)\big) \\ &= Q^{*}_{\mathcal{D}}(s,a) - \delta_k(s,a) - \gamma^k e_n\big(s^{(k)}(s,a)\big). \end{aligned} \tag{22}$$

Each horizon target therefore falls short of $Q^*_{\mathcal{D}}(s,a)$ by the amount $\delta_k(s,a) + \gamma^k e_n(s^{(k)}(s,a))$. We argue by induction that this shortfall is nonnegative. At $n = 0$, the underestimate assumption $V_0 \le V^*_{\mathcal{D}}$ gives $e_0 \ge 0$. Assuming $e_n \ge 0$, $L_k[V_n] \le Q^*_{\mathcal{D}}$ for every k with $\delta_k \ge 0$. Taking the maximum over k preserves the inequality, so $Q_{n+1} \le Q^*_{\mathcal{D}}$. Moreover, taking the maximum over $a \in \mathcal{A}_{\mathcal{D}}(s)$ gives $V_{n+1} \le V^*_{\mathcal{D}}$, i.e. $e_{n+1} \ge 0$. The underestimate property is thus preserved for all n.

For the final claim, we have

$$\begin{aligned}\big(Q^*_{\mathcal{D}} - Q_{n+1}\big)(s,a) &= Q^*_{\mathcal{D}}(s,a) - \max_{k\in\{1,K\}} L_k[V_n](s,a) \\ &= \min_{k\in\{1,K\}} \Big[Q^*_{\mathcal{D}}(s,a) - L_k[V_n](s,a)\Big] \\ &= \min_{k\in\{1,K\}} \Big[\delta_k(s,a) + \gamma^k e_n\big(s^{(k)}(s,a)\big)\Big],\end{aligned}$$

which is equation 21. The first equality is the definition of equation 19 and the last equality substitutes equation 22. □

Note that the underestimate initialization in Lemma D.4 is not restrictive in practice. When rewards lie in $[r_{\min}, r_{\max}]$, the choice $V_0 \equiv r_{\min}/(1-\gamma)$ and $Q_0 \equiv r_{\min}/(1-\gamma)$ satisfies the hypothesis for any dataset, since no value in $\mathcal{M}_{\mathcal{D}}$ can fall below that bound.

D.6 Proof of Theorem 4.5

Proof. For the first claim, since the minimum in equation 21 cannot exceed any of its terms, it is bounded by the $k = 1$ term, which equals $\gamma e_n(s^{(1)}(s,a))$ since $\delta_1 = 0$ by Lemma D.4. Thus,

$$\big(Q^*_{\mathcal{D}} - Q_{n+1}\big)(s,a) \le \gamma e_n\big(s^{(1)}(s,a)\big) \le \gamma \max_{s'} e_n(s') = \gamma \|e_n\|_\infty \qquad \text{for every } (s,a) \in \mathcal{X}_{\mathcal{D}}.$$

Taking maxima over $a \in \mathcal{A}_{\mathcal{D}}(s)$ and using $\max_a h_1(a) - \max_a h_2(a) \le \max_a (h_1(a) - h_2(a))$ for any two functions h_1, h_2,

$$e_{n+1}(s) = \max_a Q^*_{\mathcal{D}}(s,a) - \max_a Q_{n+1}(s,a) \le \max_a \big(Q^*_{\mathcal{D}} - Q_{n+1}\big)(s,a) \le \gamma \|e_n\|_\infty$$

for every s. Since $e_{n+1} \ge 0$ by Lemma D.4, this gives $\|e_{n+1}\|_\infty \le \gamma \|e_n\|_\infty$, hence $\|e_n\|_\infty \le \gamma^n \|e_0\|_\infty \to 0$, so $V_n \to V^*_{\mathcal{D}}$ and $Q_{n+1} = \max_{k\in\{1,K\}} L_k[V_n] \to Q^*_{\mathcal{D}}$. Only the $k = 1$ term entered the argument, so the conclusion holds for any rollout length K.

For the second claim, including the $k = K$ term gives the minimum in equation 21 more terms to choose from, so at every coordinate the error can only get smaller, never larger. Suppose at some (s,a),

$$\delta_K(s,a) + \gamma^K e_n\big(s^{(K)}(s,a)\big) < \gamma e_n\big(s^{(1)}(s,a)\big).$$

Then, the $k = K$ term is strictly smaller than the $k = 1$ term, so the error at (s,a) is governed by $\delta_K(s,a) + \gamma^K e_n(s^{(K)}(s,a))$ rather than by the one-step term. If in addition every action in $\mathcal{D}$ along the K-step path is optimal in $\mathcal{M}_{\mathcal{D}}$, so that $\delta_K(s,a) = 0$, the error at that coordinate contracts by γ^K instead of γ. □

D.7 Proof of Proposition 4.7

Proof. Since $\mathcal{L}_{\mathrm{R}}$ replaces the constraints of LBLP by hinge penalties, the penalty weights must be large enough that violating is never profitable; otherwise the two problems would have different minimizers. Linear programming duality answers this: the optimal dual variables of the constrained program measure the gain from violating each constraint, and therefore define the optimal threshold. Let LP_1 denote the relaxation of LBLP on $\mathcal{M}_{\mathcal{D}}$ obtained by keeping only the one-step and epigraph constraints $g_{\mathrm{B}} \le 0$ and $g_{\mathrm{E}} \le 0$. Since g_{KQ} and g_{KV} are redundant by Lemma 4.2, LP_1 has the same feasible set as the LBLP on $\mathcal{M}_{\mathcal{D}}$, and as the two programs also share the objective F, they have the same minimizers. Proposition 4.3, applied to the finite discounted MDP $\mathcal{M}_{\mathcal{D}}$, identifies this common minimizer as the unique pair $(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$. Hence LP_1 is feasible and bounded below by $p^* := F(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$, attained uniquely there. As LP_1 is a feasible and bounded linear program,

strong duality holds, and its optimal dual variables for g_{B} and g_{E} are the $(\lambda_{\mathrm{B}}^*, \lambda_{\mathrm{E}}^*)$ of the statement, indexed by $\mathcal{X}_{\mathcal{D}}$.

Next, we check that $(Q_{\mathcal{D}}^*, V_{\mathcal{D}}^*)$ minimizes $\mathcal{L}_{\mathrm{R}}$. Consider equation 4 as $\mathcal{L}_{\mathrm{R}} = \Phi_1 + \Phi_{\mathrm{K}}$, where

$$\Phi_1(Q,V) := F(Q,V) + \lambda_{\mathrm{B}} \big\| [g_{\mathrm{B}}(Q,V)]_+ \big\|_1 + \lambda_{\mathrm{E}} \big\| [g_{\mathrm{E}}(Q,V)]_+ \big\|_1,$$
$$\Phi_{\mathrm{K}}(Q,V) = \lambda_{\mathrm{K}} \Big(\big\| [g_{\mathrm{KQ}}(Q,V)]_+ \big\|_1 + \big\| [g_{\mathrm{KV}}(Q,V)]_+ \big\|_1 \Big).$$

Writing $x = (Q,V)$ and $L(x,u,v) = F(x) + \langle u, g_{\mathrm{B}}(x) \rangle + \langle v, g_{\mathrm{E}}(x) \rangle$, strong duality gives $\min_x L(x, \lambda_{\mathrm{B}}^*, \lambda_{\mathrm{E}}^*) = p^*$. Using the definition of $\| \cdot \|_1$, we get

$$\begin{aligned}
\Phi_1(x) =& F(x) + \langle \lambda_{\mathrm{B}} \mathbf{1}, [g_{\mathrm{B}}(x)]_+ \rangle + \langle \lambda_{\mathrm{E}} \mathbf{1}, [g_{\mathrm{E}}(x)]_+ \rangle \\
=& F(x) + \langle \lambda_{\mathrm{B}}^*, [g_{\mathrm{B}}(x)]_+ \rangle + \langle \lambda_{\mathrm{E}}^*, [g_{\mathrm{E}}(x)]_+ \rangle \\
& + \big\langle \lambda_{\mathrm{B}} \mathbf{1} - \lambda_{\mathrm{B}}^*, [g_{\mathrm{B}}(x)]_+ \big\rangle + \big\langle \lambda_{\mathrm{E}} \mathbf{1} - \lambda_{\mathrm{E}}^*, [g_{\mathrm{E}}(x)]_+ \big\rangle \\
\geq& L(x, \lambda_{\mathrm{B}}^*, \lambda_{\mathrm{E}}^*) + \big\langle \lambda_{\mathrm{B}} \mathbf{1} - \lambda_{\mathrm{B}}^*, [g_{\mathrm{B}}(x)]_+ \big\rangle + \big\langle \lambda_{\mathrm{E}} \mathbf{1} - \lambda_{\mathrm{E}}^*, [g_{\mathrm{E}}(x)]_+ \big\rangle \\
\geq& p^* + \big\langle \lambda_{\mathrm{B}} \mathbf{1} - \lambda_{\mathrm{B}}^*, [g_{\mathrm{B}}(x)]_+ \big\rangle + \big\langle \lambda_{\mathrm{E}} \mathbf{1} - \lambda_{\mathrm{E}}^*, [g_{\mathrm{E}}(x)]_+ \big\rangle, \qquad (23)
\end{aligned}$$

where the first inequality utilizes $g_{\mathrm{B}} \leq [g_{\mathrm{B}}]_+$ and $g_{\mathrm{E}} \leq [g_{\mathrm{E}}]_+$ together with $\lambda_{\mathrm{B}}^*, \lambda_{\mathrm{E}}^* \geq 0$, and the last inequality comes from $\min_x L(x, \lambda_{\mathrm{B}}^*, \lambda_{\mathrm{E}}^*) = p^*$. Since $(Q_{\mathcal{D}}^*, V_{\mathcal{D}}^*)$ is feasible for LP_1, both hinges in Φ_1 vanish there. The rollout term Φ_{K} vanishes as well, since by Lemma 4.2, the one-step and epigraph constraints imply $g_{\mathrm{KQ}}(x) \leq 0$ and $g_{\mathrm{KV}}(x) \leq 0$. Therefore, $\mathcal{L}_{\mathrm{R}}(Q_{\mathcal{D}}^*, V_{\mathcal{D}}^*) = p^*$ and equation 23 shows it is a minimizer.

We now check that no other point can be a minimizer. If x violates a one-step or epigraph constraint at some (s,a), the corresponding hinge is strictly positive there and its slack $\lambda_{\mathrm{B}} - \lambda_{\mathrm{B}}^*(s,a)$ or $\lambda_{\mathrm{E}} - \lambda_{\mathrm{E}}^*(s,a)$ is strictly positive by the threshold assumption. Furthermore, since $\lambda_{\mathrm{K}} \geq 0$, we have $\Phi_{\mathrm{K}} \geq 0$ everywhere. Therefore, equation 23 gives $\mathcal{L}_{\mathrm{R}}(x) > p^*$: no infeasible point can then be a minimizer since $(Q_{\mathcal{D}}^*, V_{\mathcal{D}}^*)$ already attains p^*. Every minimizer is therefore feasible for LP_1, where all hinges of Φ_1 vanish and, by Lemma 4.2, so does Φ_{K}, leaving $\mathcal{L}_{\mathrm{R}} = F$. Minimizing $\mathcal{L}_{\mathrm{R}}$ thus reduces to LP_1 itself, whose unique solution is $(Q_{\mathcal{D}}^*, V_{\mathcal{D}}^*)$ by Proposition 4.3 together with the feasible-set identity of Lemma 4.2. Finally, the rollout term entered the argument only through its nonnegativity and its vanishing on the feasible set, never through its magnitude, so the conclusion holds for every $\lambda_{\mathrm{K}} \geq 0$ and every rollout length K. □

D.8 Proof of Proposition 5.1

Proof. Fix any (Q,V) and write $(Q_c, V_c) := (Q - c\mathbf{1},\, V - c\mathbf{1})$ for $c > 0$. We evaluate equation 5 along this family.

The two objective terms are linear, so they contribute

$$\omega_{\mathrm{Q}}\, \mathbb{E}_{\mathcal{D}}[Q_c(s,a)] + \omega_{\mathrm{V}}\, \mathbb{E}_{\mathcal{D}}[V_c(s)] = \omega_{\mathrm{Q}}\, \mathbb{E}_{\mathcal{D}}[Q(s,a)] + \omega_{\mathrm{V}}\, \mathbb{E}_{\mathcal{D}}[V(s)] - c\,(\omega_{\mathrm{Q}} + \omega_{\mathrm{V}}). \qquad (24)$$

For the residuals, the epigraph term is invariant, since

$$Q_c(s,a) - V_c(s) = \big(Q(s,a) - c\big) - \big(V(s) - c\big) = Q(s,a) - V(s). \qquad (25)$$

The one-step target shifts by γc, because $\hat{y}_1(s,a) = r_0 + \gamma V(s_1)$ is affine in V with coefficient γ, so

$$\hat{y}_1(s,a)\big|_{V_c} - Q_c(s,a) = \hat{y}_1(s,a) - Q(s,a) + c\,(1-\gamma). \qquad (26)$$

The rollout target $\hat{y}_K(s,a) = \sum_{t<K} \gamma^t r_t + \gamma^K V(s_K)$ is affine in V with coefficient γ^K, so both rollout residuals shift by $c\,(1-\gamma^K)$,

$$\begin{aligned}
\hat{y}_K(s,a)\big|_{V_c} - Q_c(s,a) &= \hat{y}_K(s,a) - Q(s,a) + c\,(1-\gamma^K), \\
\hat{y}_K(s,a)\big|_{V_c} - V_c(s) &= \hat{y}_K(s,a) - V(s) + c\,(1-\gamma^K).
\end{aligned} \qquad (27)$$

Each residual above is an affine function of c with a positive slope, so there is a finite c_0 beyond which all of them are positive at every sample, and each hinge equals its argument. For $c > c_0$ the objective is therefore affine in c with

$$\frac{d}{dc} L(Q_c, V_c) = -(\omega_{\mathrm{Q}} + \omega_{\mathrm{V}}) + \lambda_{\mathrm{B}}(1-\gamma) + 2\lambda_{\mathrm{K}}(1-\gamma^K), \qquad (28)$$

where the factor 2 collects the two rollout penalties, which shift at the same rate. Without stop gradient, the update is the gradient of equation 5, and its component along this direction is the negative of equation 28. If equation 28 is negative, the gradient update lowers Q and V without bound along this direction, which yields equation 8.

When `stop_gradient` is applied to $\hat{y}_K$, its forward value is unchanged. Hence, the rollout residuals still increase at rate $1-\gamma^K$ along this family, and every hinge is active for $c > c_0$ as before. The derivative used by the update, however, differs. For $c > c_0$, each rollout penalty is affine in c, and without the stop gradient both arguments depend on c,

$$\frac{d}{dc}\,\lambda_{\mathrm{K}}\big(\hat{y}_K(s,a)\big|_{V_c} - Q_c(s,a)\big) = \lambda_{\mathrm{K}}(1-\gamma^K), \qquad \frac{d}{dc}\,\lambda_{\mathrm{K}}\big(\hat{y}_K(s,a)\big|_{V_c} - V_c(s)\big) = \lambda_{\mathrm{K}}(1-\gamma^K), \tag{29}$$

giving the rollout contribution in equation 28. With the stop gradient, $\hat{y}_K$ is treated as constant by the update, so its dependence on V_c is removed from the derivative,

$$\frac{d}{dc}\,\lambda_{\mathrm{K}}\big(\mathrm{sg}[\hat{y}_K(s,a)] - Q_c(s,a)\big) = \lambda_{\mathrm{K}}, \qquad \frac{d}{dc}\,\lambda_{\mathrm{K}}\big(\mathrm{sg}[\hat{y}_K(s,a)] - V_c(s)\big) = \lambda_{\mathrm{K}}. \tag{30}$$

The objective and epigraph terms are unaffected by the stop gradient, while the one-step penalty contributes $\lambda_{\mathrm{B}}(1-\gamma)$ as before. Thus, the component of the update along this direction is

$$-(\omega_{\mathrm{Q}} + \omega_{\mathrm{V}}) + \lambda_{\mathrm{B}}(1-\gamma) + 2\lambda_{\mathrm{K}}. \tag{31}$$

If this quantity is negative, the update induces an unbounded downward drift along this direction, yielding equation 7. □

E THE MUDWORLD ENVIRONMENT

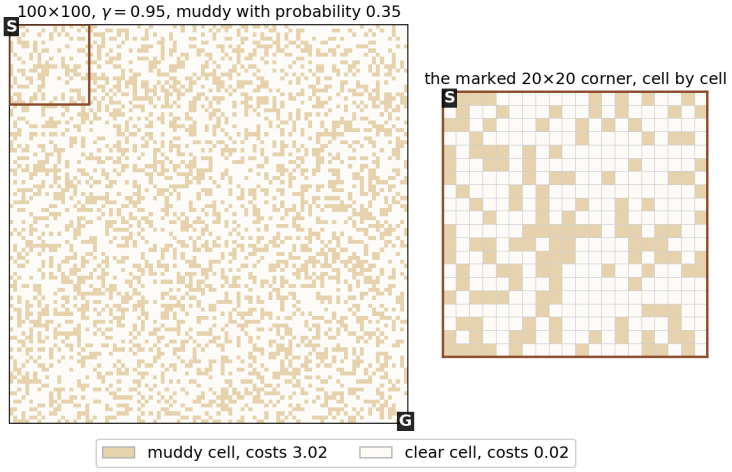

Figure 2: The `MudWorld` environment shared by both numerical experiments. Brown cells are muddy. The right image magnifies the 20×20 corner marked on the left.

To verify the two theoretical statements in Sections 4.2 and 4.3, we construct a custom gridworld, `MudWorld`, for numerical simulation. The corresponding experiments are presented in Appendices F and G, respectively. Reaching a brown cell costs `-3.02`, while reaching a white cell costs only `-0.02`. We name the environment `MudWorld` after these costly cells, which the agent must decide whether to cross or go around. The agent starts at `S` and the goal is to reach `G`. The grid is 100×100, giving $|\mathcal{S}| = 10{,}000$ states, and $\mathcal{A} = \{\texttt{Up}, \texttt{Down}, \texttt{Left}, \texttt{Right}\}$ with $|\mathcal{A}| = 4$, so $|\mathcal{X}| = 40{,}000$. `S` is the top-left cell $(0,0)$ and `G` the bottom-right cell $(99,99)$, a Manhattan distance of 198 apart. The dynamics are deterministic. Each action moves the agent one cell in its direction, and an action that would leave the grid leaves the agent where it is, so every action is available in every state. Each cell carries a cost that is charged on entry. Muddy cells are drawn i.i.d. with probability 0.35 and cost 3.02, and the remaining cells cost 0.02. The reward at a transition into s' is $1 - \mathrm{cost}(s')$ if s' is the goal and $-\mathrm{cost}(s')$ otherwise, so rewards lie in $[-3.02, 0.98]$. We use $\gamma = 0.95$. The reference solution Q^* is obtained by value iteration on $\mathcal{M}$.

F Numerical Verification of the Sandwich Property

F.1 Dataset

Since the setting is offline, a dataset is collected in advance, and it contains only `Down` and `Right` actions, so no `Up` or `Left` action appears anywhere in the data. The behavior is therefore suboptimal, and the optimal policy of $\mathcal{M}$ cannot be recovered from the dataset support. To collect the offline dataset, we run a behavior policy β that selects uniformly between the two actions, excluding the one that would leave the grid, so every episode reaches the goal in exactly 198 steps. Collecting 2,000 episodes gives 396,000 transitions, covering $|\mathcal{X}_\mathcal{D}| = 9{,}693$ pairs over $|\mathcal{S}_\mathcal{D}| = 5{,}133$ states.

F.2 Solving LBLP

We set $K = 10$, and take the objective weights from the empirical sampling distribution, $c_Q = \omega_Q \hat{d}(s,a)$ and $c_V = \omega_V \hat{d}_\mathcal{S}(s)$ with $\omega_Q = 0.1$ and $\omega_V = 0.05$, where $\hat{d}$ is the visitation frequency of $\mathcal{D}$ and $\hat{d}_\mathcal{S}$ its state marginal. Both are strictly positive on $\mathcal{X}_\mathcal{D}$ and $\mathcal{S}_\mathcal{D}$, so Definition 4.1 applies and the minimizer is unique. The program has 14,826 decision variables, 9,693 for Q on $\mathcal{X}_\mathcal{D}$ and 5,133 for V on $\mathcal{S}_\mathcal{D}$, and 38,772 constraints, 9,693 from each of the four families, with 225,776 nonzeros in the constraint matrix. Assembling y_K as an affine function of V by the K-step recursion takes about 1.5 s; we then solve the program with `scipy.optimize.linprog` using the `highs` method.

The two reference quantities (Q^* and $G_\mathcal{D}$) are obtained independently of the program. Q^* is the value iteration solution of Appendix E. We compute $G_\mathcal{D}$ in two passes over $\mathcal{D}$, first accumulating the discounted return of every suffix backward along each trajectory, then taking, for each pair, the maximum over the positions at which it appears.

F.3 Results

We repeat the simulation over eight random seeds, each generating a different mud layout and a different dataset. As shown in Figure 3, the LBLP minimizer lies between $G_\mathcal{D}$ and Q^* at every pair in all eight seeds.

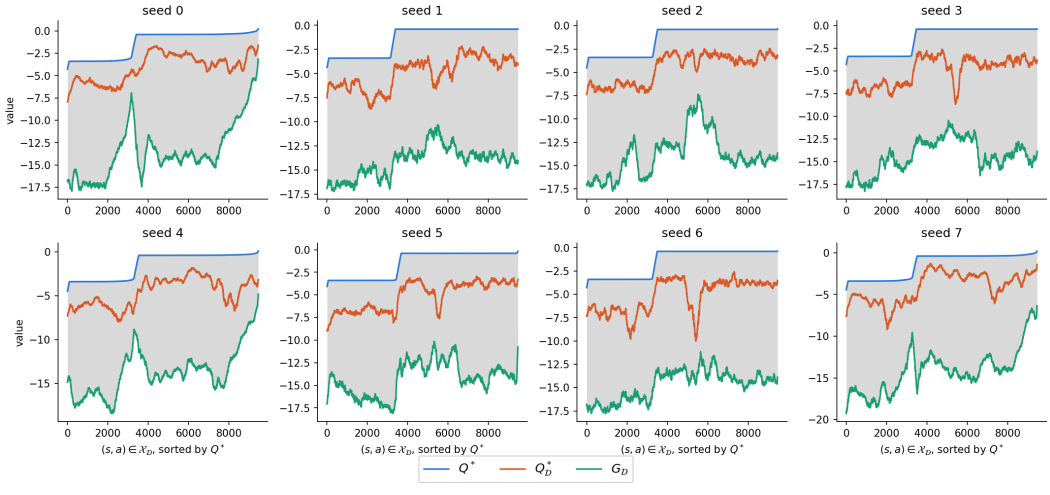

Figure 3: Proposition 4.4 across eight seeds. Each subfigure plots Q^*, $Q^*_\mathcal{D}$, and $G_\mathcal{D}$ at every pair of $\mathcal{X}_\mathcal{D}$, sorted by Q^*.

G Numerical Verification of the K-Rollout Acceleration

G.1 Dataset

In this section we reuse the environment of Appendix E with a different behavior policy. According to Lemma D.4, the error at a pair after one update is $\min_{k\in\{1,K\}}\left[\delta_k + \gamma^k e_n(s^{(k)})\right]$, and $\delta_1 = 0$, so the $k = 1$ term contributes $\gamma e_n(s^{(1)})$ and the $k = K$ term contributes $\delta_K + \gamma^K e_n(s^{(K)})$. While e_n is large, $\gamma^K e_n$ falls below γe_n by enough to absorb δ_K and the K-step term is selected, so the error contracts at γ^K. Once e_n has decayed to the point where $\gamma^K e_n$ is small against δ_K, the one-step term takes over and the rate returns to γ. The residual δ_K thus acts as a floor below which the acceleration no longer operates.

To verify such existence of δ_K and γ^K, we design and collect a second dataset with an ε-greedy policy that takes $\arg\max_a Q^*(s,a)$ with probability $1-\varepsilon$ and a uniform action otherwise, with $\varepsilon = 0.02$, from a start cell drawn uniformly over the 9,999 non-goal cells. Three hundred episodes capped at 400 steps give 41,178 transitions over $|\mathcal{X}_\mathcal{D}| = 3{,}182$ pairs and $|\mathcal{S}_\mathcal{D}| = 2{,}703$ states, with visit counts ranging from 1 to 374.

G.2 Computation

In Figure 4(a), we run equation 3 directly, $Q_{n+1} = \max_{k\in 1,K} L_k[V_n]$ and $V_{n+1}(s) = \max_{a\in\mathcal{A}_\mathcal{D}(s)} Q_{n+1}(s,a)$, starting from the underestimate $V_0 \equiv r_{\min}/(1-\gamma)$ given in Lemma D.4. The reference $(Q^*_\mathcal{D}, V^*_\mathcal{D})$ is obtained by value iteration with $\mathcal{T}_\mathcal{D}$. As shown in Figure 4(a), during the early phase of the iteration, every error curve with $K > 1$ exhibits a fast decay rate close to γ^K. However, the decay rate changes in the later phase and becomes aligned with the slope of the $K = 1$ curve, corresponding to a decay rate of γ. This provides partial evidence that δ_K indeed exists and contributes to the convergence behavior in the late stage of the iteration. (Lemma D.4)

Figure 4(b) compares the idealized iteration (equation 3) with gradient descent on the relaxed LBLP objective (equation 4), using $\lambda_B = \lambda_E = \lambda_K = 2.5$ and a learning rate of 0.03. Since the two methods use different computational procedures and are measured in different metrics, we first measure each method using its own no-rollout version as the baseline, and then compare the resulting speedup factors. Both methods start from $Q \equiv V \equiv r_{\min}/(1-\gamma)$ and are evaluated at the same target error.

Together, Figures 4(a) and 4(b) verify that the acceleration indeed exists. Figure 4(b) further suggests that, under favorable conditions, this acceleration can even outperform the idealized iteration. Both methods achieve a speedup compared with the version without rollout terms.

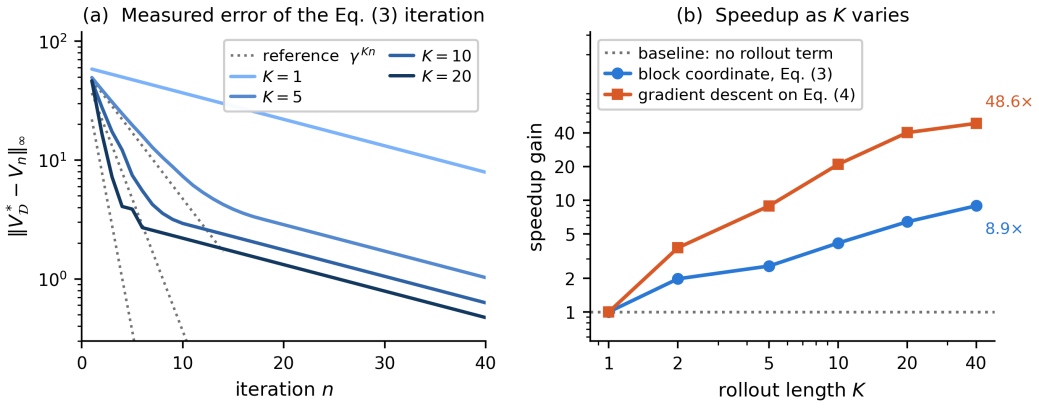

Figure 4: K-rollout acceleration on `MudWorld`. **(a)** Each curve leaves the start on its own γ^{Kn} slope, and at the δ_K floor the curves rejoin the γ^n slope. **(b)** The direct gradient descent on the relaxed LBLP objective exhibits an acceleration and can outperform the idealized iteration under favorable conditions, while both methods achieve a speedup over the version without rollout terms.

H DESIGN OF `STOP_GRADIENT` ON $\hat{y}_K$ UNDER NEURAL PARAMETERIZATION

As mentioned in Section 5, we apply `stop_gradient` to the rollout term $\hat{y}_K$. The device might be familiar from temporal difference methods, where the target is the label of a regression and holding it fixed keeps the regression from chasing its own output. However, our use of it has a different origin, and follows from the structure of the tabular objective of Definition 4.6, which the neural parameterization breaks in one specific way. In equation 4 the value function is a vector with one coordinate per state, and distinct states occupy orthogonal coordinates. Under a network V_ϕ, the values at distinct states share parameters, so a gradient component aimed at one state also moves the others. The following remark makes the consequence precise.

Remark H.1 (One descent step on the rollout penalty)**.** *Fix a recorded pair (s, a) with $s \neq s_K$, suppose the hinge $\lambda_{\mathrm{K}}\big(y_K(s,a) - V(s)\big)_+$ is active, and take one descent step with rate η on this term alone. In equation 4 the value function is a vector with one coordinate per state, and the term has the two derivatives*

$$\frac{\partial}{\partial V(s)}\,\lambda_K\big(y_K(s,a) - V(s)\big)_+ = -\lambda_K, \qquad \frac{\partial}{\partial V(s_K)}\,\lambda_K\big(y_K(s,a) - V(s)\big)_+ = \lambda_K\gamma^K, \quad (32)$$

so the step moves the coordinate $V(s)$ by

$$\Delta V(s) = \eta\lambda_K. \quad (33)$$

The second derivative in equation 32 moves $V(s_K)$ as well, by $-\eta\lambda_K\gamma^K$, but that coordinate does not enter equation 33. In equation 5 the decision variable is ϕ and one network carries V at every state, so the two derivatives of equation 32 collapse into the single gradient

$$\nabla_\phi\,\lambda_K\big(\hat{y}_K(s,a) - V_\phi(s)\big)_+ = \lambda_K\Big(\gamma^K\,\nabla_\phi V_\phi(s_K) - \nabla_\phi V_\phi(s)\Big), \quad (34)$$

the step is $\Delta\phi = -\eta\lambda_K\big(\gamma^K\nabla_\phi V_\phi(s_K) - \nabla_\phi V_\phi(s)\big)$, and the value at s moves by

$$\Delta V_\phi(s) = \eta\lambda_K\Big(\big\|\nabla_\phi V_\phi(s)\big\|^2 - \gamma^K\,\nabla_\phi V_\phi(s)^\top\nabla_\phi V_\phi(s_K)\Big) + O(\eta^2). \quad (35)$$

The second term of equation 35 depends on s_K, whereas equation 33 has no counterpart to it. This term arises because the shared parameters of a neural network allow the gradient of $V_\phi(s_K)$ to contribute to the update at s. Consequently, without `stop_gradient`, the gradient of $\hat{y}_K$ introduces an additional term that is absent from the tabular update and changes the intended gradient direction. Applying `stop_gradient` to $\hat{y}_K$ removes this term, so that equation 35 recovers the gradient structure of equation 33.

I Stationary Point Analysis

The four hinge terms in equation 5 are nondifferentiable at pairs where their constraint residuals equal zero. Hence, each admits a set of subgradients rather than a unique gradient. Let u, v, w, x denote selections from the subdifferentials of $\lambda_{\mathrm{B}}\mathbb{E}_{\mathcal{D}}[(g_{\mathrm{B}})_+]$, $\lambda_{\mathrm{E}}\mathbb{E}_{\mathcal{D}}[(g_{\mathrm{E}})_+]$, $\lambda_{\mathrm{K}}\mathbb{E}_{\mathcal{D}}[(g_{\mathrm{KQ}})_+]$, and $\lambda_{\mathrm{K}}\mathbb{E}_{\mathcal{D}}[(g_{\mathrm{KV}})_+]$, respectively. For a constraint residual g and coefficient λ, an admissible subgradient selection takes the value $\lambda d_{\mathcal{D}}(s,a)$ when $g(s,a) > 0$, the value 0 when $g(s,a) < 0$, and any value in $[0, \lambda d_{\mathcal{D}}(s,a)]$ when $g(s,a) = 0$.

Definition I.1 (Detached update field)**.** *For a fixed $\bar{y}$ on $\mathcal{X}_{\mathcal{D}}$, let us define $\mathcal{L}_R(Q, V; \bar{y})$ as equation 5 with $\mathrm{sg}[\hat{y}_K]$ replaced by $\bar{y}$ and with Q and V treated as free variables. This function is convex in (Q, V), and let us define the detached update field as*

$$\mathcal{G}(Q,V) := \partial_{(Q,V)}\, \mathcal{L}_R(Q, V; \bar{y})\big|_{\bar{y}=\hat{y}_K(V)}, \tag{36}$$

where ∂ denotes the convex subdifferential. The update of ALBUM descends along $\mathcal{G}$, which is in general not the subdifferential of any function of (Q, V).

At $(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$, the admissible selections have the following ranges:

- $u(s,a) \in [0, \lambda_{\mathrm{B}} d_{\mathcal{D}}(s,a)]$ for every $(s,a) \in \mathcal{X}_{\mathcal{D}}$;
- $v(s,a) \in [0, \lambda_{\mathrm{E}} d_{\mathcal{D}}(s,a)]$ when $Q^*_{\mathcal{D}}(s,a) = V^*_{\mathcal{D}}(s)$, and $v(s,a) = 0$ otherwise;
- $w(s,a) \in [0, \lambda_{\mathrm{K}} d_{\mathcal{D}}(s,a)]$ when $g_{\mathrm{KQ}}(s,a) = 0$, and $w(s,a) = 0$ otherwise;
- $x(s,a) \in [0, \lambda_{\mathrm{K}} d_{\mathcal{D}}(s,a)]$ when $g_{\mathrm{KV}}(s,a) = 0$, and $x(s,a) = 0$ otherwise.

The conditions on w and x follow from the two K-step residuals at $(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$. The K-step target along the dataset trajectory of (s,a) never exceeds the in-sample optimum, so

$$g_{\mathrm{KQ}}(s,a) = \hat{y}_K(s,a) - Q^*_{\mathcal{D}}(s,a) \le 0, \qquad g_{\mathrm{KV}}(s,a) = g_{\mathrm{KQ}}(s,a) - \big(V^*_{\mathcal{D}}(s) - Q^*_{\mathcal{D}}(s,a)\big) \le 0,$$

where g_{KQ} vanishes precisely when the dataset continuation of (s,a) is optimal in $\mathcal{M}_{\mathcal{D}}$. Hence g_{KV} vanishes only on those pairs at which a also attains $V^*_{\mathcal{D}}(s)$.

Proposition I.2 (Stationarity of the LBLP solution under the detached update)**.** *Suppose there exist admissible selections u, v, w, x at $(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$ satisfying*

$$\omega_{\mathrm{Q}} d_{\mathcal{D}}(s,a) - u(s,a) + v(s,a) - w(s,a) = 0 \qquad \textit{for every } (s,a) \in \mathcal{X}_{\mathcal{D}}, \tag{37}$$

and

$$\omega_{\mathrm{V}} d_{\mathcal{S}}(s) + \gamma \sum_{(t,b)\in\mathcal{I}(s)} u(t,b) - \sum_{a\in\mathcal{A}_{\mathcal{D}}(s)} \big(v(s,a) + x(s,a)\big) = 0 \qquad \textit{for every } s \in \mathcal{S}_{\mathcal{D}}, \tag{38}$$

where

$$\mathcal{I}(s) := \{(t,b) \in \mathcal{X}_{\mathcal{D}} : t' = s\}$$

collects the pairs whose one-step target contains $V(s)$. Then

$$0 \in \mathcal{G}(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}}).$$

Proof. With $\bar{y}$ held fixed, Definition I.1 gives

$$\begin{aligned}\mathcal{L}_R(Q,V;\bar{y}) = {} & \omega_{\mathrm{Q}}\, \mathbb{E}_{\mathcal{D}}[Q(s,a)] + \omega_{\mathrm{V}}\, \mathbb{E}_{\mathcal{D}}[V(s)] \\ & + \lambda_{\mathrm{B}}\, \mathbb{E}_{\mathcal{D}}\big[(r + \gamma V(s') - Q(s,a))_+\big] + \lambda_{\mathrm{E}}\, \mathbb{E}_{\mathcal{D}}\big[(Q(s,a) - V(s))_+\big] \\ & + \lambda_{\mathrm{K}} \left(\mathbb{E}_{\mathcal{D}}\big[(\bar{y}(s,a) - Q(s,a))_+\big] + \mathbb{E}_{\mathcal{D}}\big[(\bar{y}(s,a) - V(s))_+\big]\right).\end{aligned} \tag{39}$$

Every term is a finite convex function and each hinge composes $[\,\cdot\,]_+$ with an affine map of (Q, V), so the sum rule and the chain rule for affine composition apply (Rockafellar, 1997, Theorems 23.8 and 23.9). Selecting one subgradient from each hinge determines one element of $\mathcal{G}$, and we take the admissible selections u, v, w, x of the statement, evaluated at $(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$ with $\bar{y} = \hat{y}_K$, which already carry the weight $d_{\mathcal{D}}$.

Differentiating the objective terms gives

$$\frac{\partial}{\partial Q(s,a)}\,\omega_{\mathrm{Q}}\mathbb{E}_{\mathcal{D}}[Q]=\omega_{\mathrm{Q}}d_{\mathcal{D}}(s,a),\qquad \frac{\partial}{\partial V(s)}\,\omega_{\mathrm{V}}\mathbb{E}_{\mathcal{D}}[V]=\omega_{\mathrm{V}}d_{\mathcal{S}}(s). \tag{40}$$

For the hinges, the chain rule sends each selection to the transpose of its affine map, so at $Q(s,a)$ we obtain the contributions

$$-u(s,a),\qquad +v(s,a),\qquad -w(s,a),\qquad 0, \tag{41}$$

from g_{B}, g_{E}, g_{KQ} and g_{KV} respectively, and at $V(s)$ the contributions

$$\gamma\sum_{(t,b)\in\mathcal{I}(s)}u(t,b),\qquad -\sum_{a\in\mathcal{A}_{\mathcal{D}}(s)}v(s,a),\qquad 0,\qquad -\sum_{a\in\mathcal{A}_{\mathcal{D}}(s)}x(s,a), \tag{42}$$

where the first sum collects the pairs whose one-step argument $r+\gamma V(t')$ contains $V(s)$, and the two zeros hold because $\bar{y}$ does not depend on (Q,V).

Summing the contributions yields

$$\omega_{\mathrm{Q}}d_{\mathcal{D}}(s,a)-u(s,a)+v(s,a)-w(s,a)=0 \tag{43}$$

by equation 37, and

$$\omega_{\mathrm{V}}d_{\mathcal{S}}(s)+\gamma\sum_{(t,b)\in\mathcal{I}(s)}u(t,b)-\sum_{a\in\mathcal{A}_{\mathcal{D}}(s)}\big(v(s,a)+x(s,a)\big)=0 \tag{44}$$

by equation 38. The selected element of $\mathcal{G}$ therefore has zero component at every $Q(s,a)$ and every $V(s)$, which yields $0\in\mathcal{G}(Q^*_{\mathcal{D}},V^*_{\mathcal{D}})$. □

The existence of such selections is not automatic. The objective terms contribute $\omega_{\mathrm{Q}}d_{\mathcal{D}}(s,a)$ and $\omega_{\mathrm{V}}d_{\mathcal{S}}(s)$ at each coordinate, while the selections enter equation 37 and equation 38 with the opposite sign and are each bounded by their respective penalty coefficients. Which selections carry these contributions separates two regimes, and each yields its own bound on the coefficients.

The one-step selection regime. Setting $w=x=0$ leaves u and v to carry both conditions. The affine map of g_{B} places $u(s,a)$ at $Q(s,a)$ in equation 37 and at $V(s')$ in equation 38 with weight γ, so the same selection enters two conditions and accumulates along a trajectory of $\mathcal{M}_{\mathcal{D}}$. The bound λ_{B} places on it therefore grows with the effective horizon $1/(1-\gamma)$.

Remark I.3 (The one-step selection alone)**.** *Suppose $|\mathcal{A}_{\mathcal{D}}(s)|=1$ for every $s\in\mathcal{S}_{\mathcal{D}}$, so that $d_{\mathcal{D}}(s,a)=d_{\mathcal{S}}(s)$, and consider a dataset trajectory $(s_0,a_0),\dots,(s_{T-1},a_{T-1})$ in which $\mathcal{I}(s_0)$ is empty and $\mathcal{I}(s_t)=\{(s_{t-1},a_{t-1})\}$ for $1\le t\le T-1$, so that each pair appears once and $d_{\mathcal{D}}$ is uniform. At $(Q^*_{\mathcal{D}},V^*_{\mathcal{D}})$ the residual g_{B} vanishes by Bellman optimality, and the single action gives $Q^*_{\mathcal{D}}(s,a)=V^*_{\mathcal{D}}(s)$, so both u and v are free. Fix $w=x=0$ and write $u_t:=u(s_t,a_t)$ and $v_t:=v(s_t,a_t)$. At (s_t,a_t), equation 37 gives*

$$v_t=u_t-\omega_{\mathrm{Q}}\,d_{\mathcal{D}},$$

and at s_t, the single action makes equation 38 reduces to $\omega_{\mathrm{V}}d_{\mathcal{D}}+\gamma u_{t-1}-v_t=0$, so

$$u_t=\gamma u_{t-1}+(\omega_{\mathrm{Q}}+\omega_{\mathrm{V}})\,d_{\mathcal{D}},\qquad u_{-1}=0.$$

Unrolling from $t=0$,

$$u_t=(\omega_{\mathrm{Q}}+\omega_{\mathrm{V}})\,d_{\mathcal{D}}\sum_{j=0}^{t}\gamma^{\,j}=(\omega_{\mathrm{Q}}+\omega_{\mathrm{V}})\,d_{\mathcal{D}}\,\frac{1-\gamma^{\,t+1}}{1-\gamma},$$

which increases in t and stays below $(\omega_{\mathrm{Q}}+\omega_{\mathrm{V}})d_{\mathcal{D}}/(1-\gamma)$. The admissibility bounds $u_t\le\lambda_{\mathrm{B}}d_{\mathcal{D}}$ and $v_t\le\lambda_{\mathrm{E}}d_{\mathcal{D}}$ therefore hold once

$$\lambda_{\mathrm{B}}\;\ge\;\frac{\omega_{\mathrm{Q}}+\omega_{\mathrm{V}}}{1-\gamma},\qquad \lambda_{\mathrm{E}}\;\ge\;\frac{\omega_{\mathrm{Q}}+\omega_{\mathrm{V}}}{1-\gamma}-\omega_{\mathrm{Q}},$$

so $0\in\mathcal{G}(Q^*_{\mathcal{D}},V^*_{\mathcal{D}})$ holds without the rollout selections at coefficients of this order. The same bound on λ_{B} appears in equation 7 at $\lambda_{\mathrm{K}}=0$.

Without uniformity the ratio $d_{\max}/d_{\min}$ enters this bound, so the stationarity conditions require a larger λ_{B} than equation 7 does at $\lambda_{\mathrm{K}}=0$. The aggregation of Remark I.6 replaces the coordinate-wise bounds by their totals, which carry no dependence on $d_{\mathcal{D}}$.

The rollout selections regime. Taking w and x nonzero instead allows them to carry the objective terms. The selection $w(s,a)$ enters equation 37 at (s,a) and $x(s,a)$ enters equation 38 at s, and neither appears elsewhere, since $\bar{y}$ is held fixed in Definition I.1 and contributes nothing at $V(s_K)$. Unlike u, which the affine map of g_{B} places at both $Q(s,a)$ and $V(s')$, they offset the objective term at that coordinate alone, so the resulting bound on λ_{K} involves neither γ nor K.

Proposition I.4 (Stationarity of the LBLP solution under the detached update)**.** *Suppose the transition dynamics of $\mathcal{M}$ are deterministic, and let δ_K be as in Lemma D.4. Let us define $\mathcal{A}^*_{\mathcal{D}}(s) := \{a \in \mathcal{A}_{\mathcal{D}}(s) : Q^*_{\mathcal{D}}(s,a) = V^*_{\mathcal{D}}(s)\}$. Assume $\delta_K(s,a) = 0$ for every $(s,a) \in \mathcal{X}_{\mathcal{D}}$ and*

$$\lambda_{\mathrm{K}} \geq \omega_{\mathrm{Q}}, \qquad \lambda_{\mathrm{K}} \sum_{a \in \mathcal{A}^*_{\mathcal{D}}(s)} d_{\mathcal{D}}(s,a) \;\geq\; \omega_{\mathrm{V}}\, d_{\mathcal{S}}(s) \quad \text{for every } s \in \mathcal{S}_{\mathcal{D}}. \tag{45}$$

Then $0 \in \mathcal{G}(Q^*_{\mathcal{D}}, V^*_{\mathcal{D}})$ for every $\lambda_{\mathrm{B}} \geq 0$ and every $\lambda_{\mathrm{E}} \geq 0$.

Proof. The assumption $\delta_K = 0$ gives $g_{\mathrm{KQ}}(s,a) = 0$ on $\mathcal{X}_{\mathcal{D}}$, and $g_{\mathrm{KV}}(s,a) = 0$ exactly on the pairs with $a \in \mathcal{A}^*_{\mathcal{D}}(s)$. The selections

$$u = 0, \qquad v = 0, \qquad w(s,a) = \omega_{\mathrm{Q}} d_{\mathcal{D}}(s,a),$$

$$x(s,a) = \omega_{\mathrm{V}}\, d_{\mathcal{S}}(s)\, \frac{d_{\mathcal{D}}(s,a)}{\sum_{a' \in \mathcal{A}^*_{\mathcal{D}}(s)} d_{\mathcal{D}}(s,a')}\, \mathbf{1}\{a \in \mathcal{A}^*_{\mathcal{D}}(s)\}.$$

are admissible under equation 45, since $w(s,a) \leq \lambda_{\mathrm{K}} d_{\mathcal{D}}(s,a)$ and $x(s,a) \leq \lambda_{\mathrm{K}} d_{\mathcal{D}}(s,a)$, and they vanish wherever the corresponding residual is strictly negative. Substituting them into equation 37 and equation 38 yields $\omega_{\mathrm{Q}} d_{\mathcal{D}} - w = 0$ and $\omega_{\mathrm{V}} d_{\mathcal{S}}(s) - \sum_{a \in \mathcal{A}_{\mathcal{D}}(s)} x(s,a) = 0$, so Proposition I.2 applies. □

Corollary I.5 (Single-action empirical MDP)**.** *If $|\mathcal{A}_{\mathcal{D}}(s)| = 1$ for every $s \in \mathcal{S}_{\mathcal{D}}$, then $\delta_K = 0$ on $\mathcal{X}_{\mathcal{D}}$ and $d_{\mathcal{D}}(s,a) = d_{\mathcal{S}}(s)$, so equation 45 reduces to $\lambda_{\mathrm{K}} \geq \max(\omega_{\mathrm{Q}}, \omega_{\mathrm{V}})$. The coefficients of Table 3 satisfy this condition.*

Remark I.6 (The boundedness conditions along the common shift direction)**.** *Summing equation 37 over $(s,a) \in \mathcal{X}_{\mathcal{D}}$ and equation 38 over $s \in \mathcal{S}_{\mathcal{D}}$, the selection v cancels, since it appears with $+v(s,a)$ in the first and with $-v(s,a)$ in the second, and the selection u appears once with $-u(s,a)$ and once with $+\gamma u(s,a)$. With $\sum_{\mathcal{X}_{\mathcal{D}}} d_{\mathcal{D}} = \sum_{\mathcal{S}_{\mathcal{D}}} d_{\mathcal{S}} = 1$,*

$$\omega_{\mathrm{Q}} + \omega_{\mathrm{V}} = (1-\gamma) \sum_{\mathcal{X}_{\mathcal{D}}} u + \sum_{\mathcal{X}_{\mathcal{D}}} w + \sum_{\mathcal{X}_{\mathcal{D}}} x.$$

Admissibility gives $u(s,a) \leq \lambda_{\mathrm{B}} d_{\mathcal{D}}(s,a)$ and $w(s,a), x(s,a) \leq \lambda_{\mathrm{K}} d_{\mathcal{D}}(s,a)$, so the three sums are bounded by λ_{B}, λ_{K} and λ_{K} respectively, and

$$\omega_{\mathrm{Q}} + \omega_{\mathrm{V}} \;\leq\; \lambda_{\mathrm{B}}(1-\gamma) + 2\lambda_{\mathrm{K}},$$

which is equation 7. Setting $\lambda_{\mathrm{K}} = 0$ leaves $\lambda_{\mathrm{B}} \geq (\omega_{\mathrm{Q}} + \omega_{\mathrm{V}})/(1-\gamma)$, the bound of Remark I.3. Without `stop_gradient` each rollout selection reappears at $V(s_K)$ with weight γ^K, and the same computation yields equation 8. Condition equation 7 is the component of equation 37 and equation 38 along the common shift direction.

J OGBENCH EXPERIMENT DETAILS

Hyperparameter	Value
Actor learning rate	3×10^{-4}
Critic learning rate	1×10^{-4}
Optimizer	Adam (Kingma & Ba, 2014)
Gradient steps	1,000,000
Minibatch size	256
MLP dimensions	$[512, 512, 512, 512]$ (actor and critic)
Nonlinearity	GELU (Hendrycks & Gimpel, 2016)
Layer normalization	critic only
Critic ensemble size	1
Policy	Gaussian, state-independent σ
Discount factor γ	0.995
Sequence length K	10
ω_Q	0.1
ω_V	0.05
λ_B	2.5
λ_E	2.5
λ_K	1.0
BC coefficient α	task-specific (see below)

Table 3: Hyperparameters for ALBUM

Environment	α
antmaze-large-navigate	0.01
antmaze-giant-navigate	0.003
humanoidmaze-medium-navigate	0.003
humanoidmaze-large-navigate	0.003
antsoccer-arena-navigate	0.01
cube-single-play	0.3
cube-double-play	0.1
scene-play	0.1
puzzle-3x3-play	0.1
puzzle-4x4-play	0.1

Table 4: BC coefficient α per environment.

K ADDITIONAL RESULTS ON OGBENCH

		Gaussian Policies			Diffusion Policies			Flow Policies				Gaussian (Ours)
Environment	Task	BC	IQL	ReBRAC	IDQL	SRPO	CAC	FAWAC	FBRAC	IFQL	FQL	ALBUM (Ours)
antmaze-large-navigate	1*	0	48	91	0	0	42	1	70	24	80	**92.5** ±3.1
	2	6	42	**88**	14	4	1	0	35	8	57	80.2 ±6.0
	3	29	72	51	26	3	49	12	83	52	**93**	91.8 ±2.7
	4	8	51	84	62	45	17	10	37	18	80	**92.5** ±4.0
	5	10	54	90	2	1	55	9	76	38	83	**92.0** ±4.7
antmaze-giant-navigate	1*	0	0	**27**	0	0	0	0	0	0	4	3.5 ±3.6
	2	0	1	16	0	0	0	0	4	0	9	**76.2** ±7.2
	3	0	0	**34**	0	0	0	0	0	0	0	0.0 ±0.0
	4	0	0	5	0	0	0	0	9	0	14	**41.2** ±21.2
	5	1	19	**49**	0	0	0	0	6	13	16	19.5 ±25.6
humanoidmaze-medium-navigate	1*	1	32	16	1	0	38	6	25	**69**	19	2.0 ±2.8
	2	1	41	18	1	1	47	40	76	85	**94**	79.0 ±6.9
	3	6	25	36	0	2	**83**	19	27	49	74	4.8 ±3.7
	4	0	0	**15**	1	1	5	1	1	1	3	2.2 ±2.5
	5	2	66	24	1	3	91	31	63	**98**	97	89.5 ±3.1
humanoidmaze-large-navigate	1*	0	3	2	0	0	1	0	0	6	**7**	0.0 ±0.0
	2	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0.0** ±0.0
	3	1	7	8	3	1	2	1	10	**48**	11	6.2 ±4.8
	4	1	1	1	0	0	0	0	0	1	**2**	0.5 ±1.3
	5	0	1	**2**	0	0	0	0	1	0	1	1.0 ±1.0
antsoccer-arena-navigate	1	2	14	0	44	2	1	22	17	61	**77**	66.0 ±7.0
	2	2	17	0	15	3	0	8	8	75	**88**	58.0 ±7.8
	3	0	6	0	0	0	8	11	16	14	**61**	4.8 ±6.9
	4*	1	3	0	0	0	0	12	24	16	**39**	30.2 ±9.1
	5	0	2	0	0	0	0	9	15	0	**36**	21.5 ±5.6
cube-single-play	1	10	88	89	95	89	77	81	73	79	**97**	84.5 ±10.6
	2*	3	85	92	96	82	80	81	83	73	**97**	89.8 ±6.1
	3	9	91	93	**99**	96	98	87	82	88	98	90.8 ±4.1
	4	2	73	92	93	70	91	79	79	79	**94**	84.2 ±6.5
	5	3	78	87	90	61	80	78	76	77	**93**	67.0 ±6.0
cube-double-play	1	8	27	45	39	7	21	21	47	35	**61**	47.2 ±8.8
	2*	0	1	7	16	0	2	2	22	9	**36**	5.2 ±4.7
	3	0	0	4	17	0	3	1	4	8	**22**	4.5 ±3.1
	4	0	0	1	0	0	0	0	0	1	**5**	2.0 ±1.4
	5	0	4	4	1	0	3	2	2	17	**19**	7.8 ±7.0
scene-play	1	19	94	95	**100**	94	**100**	87	96	98	**100**	91.8 ±4.9
	2*	1	12	50	33	2	50	18	46	0	76	**93.2** ±3.2
	3	1	32	55	94	4	49	38	78	54	**98**	70.8 ±11.7
	4	2	0	3	4	0	0	6	4	0	5	**66.2** ±9.5
	5	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0.0** ±0.0
puzzle-3x3-play	1	5	33	**97**	52	89	**97**	25	63	94	90	96.2 ±4.1
	2	1	4	1	0	0	0	4	2	1	16	**77.5** ±32.6
	3	1	3	3	0	0	0	1	1	0	10	**39.0** ±16.7
	4*	1	2	2	0	0	0	1	2	0	16	**93.0** ±7.9
	5	1	3	5	0	0	0	1	2	0	**16**	0.8 ±1.4
puzzle-4x4-play	1	1	12	26	48	24	44	1	32	**49**	34	40.2 ±6.4
	2	0	7	12	14	0	0	0	5	4	**16**	11.2 ±3.5
	3	0	9	15	34	21	29	1	20	**50**	18	33.2 ±15.0
	4*	0	5	10	**26**	7	1	0	5	21	11	11.8 ±4.7
	5	0	4	7	**24**	1	0	0	4	2	7	6.2 ±3.1

Table 5: Per-task results against all baselines. Ours is the mean over 8 seeds at the final evaluation, ± the standard deviation across seeds; baseline means are taken from the FQL paper. The best entry in each row is bold. * indicates the default task.

L Ablation studies and Further Analysis

This appendix gives the full results for the questions raised in Section 6.1.

- Why does ALBUM remain stable without target networks or critic ensembles?(Appendix L.1)
- How does ALBUM compare with action-chunking methods? (Appendix L.2)
- How much does ALBUM reduce computational cost? (Appendix L.3)
- Why should ALBUM satisfy the boundedness condition? (Appendix L.4)
- How do the rollout horizon K and λ_{K} affect learning? (Appendix L.5)
- Does raising the one-step penalties recover performance without the K-step constraint? (Appendix L.6)
- Does the ALBUM critic improvement transfer to a generative policy? (Appendix L.7)

It also answers the following questions, which the main text has no space for.

- How sensitive is ALBUM to the penalty coefficients? (Appendix L.8)
- Is ALBUM's lower performance on the harder navigation domains a matter of convergence speed? (Appendix L.9)
- Is the benefit of the K-step penalties specific to the LBLP formulation? (Appendix L.10)
- In continuous state spaces the program's solution reduces to the discounted return of each dataset trajectory. What is the gap between the neural critic and this quantity? (Appendix L.11)

L.1 Target Networks and Critic Ensembles

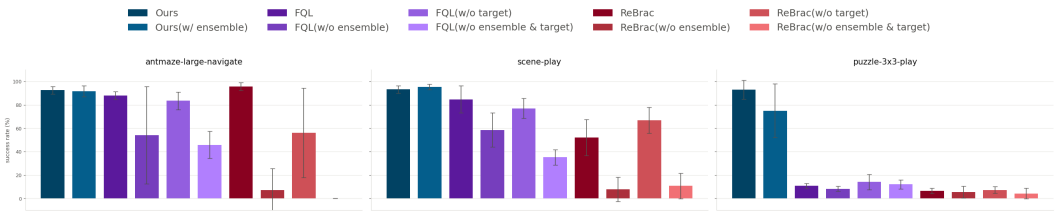

Figure 5: Critic ensemble and target network. ALBUM has no target network and appears once; Both baselines degrade when their ensemble and target are removed, ReBRAC most.

We compare against FQL and ReBRAC, the strongest flow-based and Gaussian methods in Table 1. Both rely on a target network and a two-critic ensemble, which have been standard in offline actor-critic methods since TD3 (Fujimoto et al., 2018) and TD3+BC (Fujimoto & Gu, 2021). ReBRAC provides the closer comparison, since ALBUM also extracts a Gaussian policy, so differences in policy expressiveness do not account for the performance gap. We use the default task of each environment, which is task 1 of `antmaze-large-navigate`, task 2 of `scene-play`, and task 4 of `puzzle-3x3-play`. Figure 5 removes the critic ensemble, the target network, and both. ALBUM uses a single critic and no target network, the configuration with both devices removed. In contrast, both baselines degrade on `antmaze-large-navigate` and `scene-play` when the ensemble is removed and when both devices are removed, most notably ReBRAC. Removing the target network alone degrades ReBRAC most on `antmaze-large-navigate`.

This behavior follows from where the two devices come from. Target networks with EMA updates trace back to DQN and DDPG (Mnih et al., 2015; Lillicrap et al., 2015), where the semi-gradient update of Q-learning regresses Q_θ onto a target that itself depends on Q_θ, and the slowly moving copy is what keeps that regression from chasing its own output. The ALBUM objective contains no squared regression and no such target, since Q_θ and V_ϕ are decision variables of a single objective, so the mechanism the device was introduced to control is absent. Critic ensembles address a different failure. The maximization inside the Q-learning update selects whichever action carries the largest error, which biases the backup upward, and double Q-learning and its ensemble descendants avoid this by decorrelating the estimates that the maximum is taken over (Hasselt, 2010; Van Hasselt et al., 2016; Fujimoto et al., 2018). In LBLP the maximization is a family of constraints rather than a step of the update rule, so the critic update never selects an action through a maximum over estimated values.

L.2 Comparison with Action-Chunking Methods

Comparison with horizon-reduction methods. We also compare ALBUM with methods that shorten the horizon through multi-step returns or action chunks, such as QC-FQL (Li et al., 2025), DEAS (Kim et al., 2026), DQC (Li et al., 2026), CGQ (Song et al., 2026), and NFQL, an n-step variant of FQL that CGQ evaluates as a baseline. Table 6 reports their scores from CGQ, restricted to the five task categories whose dataset, reward, and number of gradient steps match those of Park et al. (2025b) and ours, so the subset is determined by protocol agreement rather than by task performance. ALBUM attains the highest score on `antmaze-large-navigate` and `antmaze-giant-navigate`, exceeding DQC, the strongest action-chunking method on both domains, by 18.8 and 16.4 points, respectively. Its average of 37.5 is higher than those of NFQL and QC-FQL and lower than those of FQL, DEAS, CGQ, and DQC, and the methods with action-chunked critics lead on `cube-double-play` and `puzzle-4x4-play`. These methods rely on chunked critics, and those marked with † in Table 6 also execute action sequences, whereas ALBUM uses a single critic and a Gaussian policy that executes one action at a time. It also requires far less computation than QC-FQL, DQC, and CGQ, as we quantify in Appendix L.3.

	Flow, no Q-chunking		Flow, Q-chunking				Gaussian, no Q-chunking
Task Category	FQL	NFQL	QC-FQL†	DEAS†	DQC	CGQ	ALBUM (Ours)
antmaze-large-navigate	<u>79</u>	47	20	67	71	67	**89.8**
antmaze-giant-navigate	9	2	0	8	<u>10</u>	4	**26.4**
humanoidmaze-medium-navigate	<u>58</u>	23	4	37	**93**	34	34.0
cube-double-play	29	11	39	<u>48</u>	31	**69**	14.6
puzzle-4x4-play	17	23	26	**39**	8	<u>28</u>	22.7
Average	38.4	21.2	17.8	39.8	**42.6**	<u>40.4</u>	37.5

Table 6: Comparison with multi-step and Q-chunking methods. Baseline numbers obtained from CGQ. †Methods executing action sequences $\in \mathcal{A}^h$. Baselines report the mean of the last three evaluations (150 episodes each) over 4 seeds, and ALBUM the mean of the last three evaluations (50 episodes each) over 8 seeds. The ALBUM runs are the same as those in Table 1, which reports the evaluation at 1M steps only, so the two tables differ only in the aggregation of evaluation points. Best in bold, second underlined.

L.3 Computational Cost Comparison

In this section, we measure the computational cost of each method in terms of the number of network parameters, GFLOPs per step, wall-clock time per step, the number of value networks, and peak GPU memory usage. We use `antmaze-large-navigate-singletask-task1-v0` and run three seeds on a single NVIDIA GeForce RTX 3080. Every method uses its default configuration, including its critic ensemble and target copies, and the action-chunking methods use the chunk lengths reported for `antmaze-large` in Table 8 of Song et al. (2026), which are 5 for QC-FQL, 25 for DQC, and 10 for CGQ. Since ReBRAC skips its actor update on alternate steps, we also report ReBRAC*, which updates the actor at every step, so that every method updates its actor at the same rate. All methods use a minibatch of 256 transitions and four hidden layers of width 512. Parameter counts and GFLOPs per step are read from the compiled program, and peak GPU memory is read from the device allocator. Reported times are the mean over three seeds, and the standard deviation stays below 3% in every case.

ALBUM trains with 2.43M parameters and holds two value networks, whereas the other methods use 4.86M to 10.05M parameters and hold four to eight value networks. Its peak GPU memory of 78 MiB is the lowest among all methods, a reduction of 39% against FQL and of 71% against CGQ. When every method updates its actor at each step, ALBUM uses 28% fewer FLOPs and 1% less time per step than ReBRAC*, 7% fewer FLOPs and 8% less time than IQL, and 42% fewer FLOPs and 44% less time than FQL. Against the action-chunking methods, it uses 40% fewer FLOPs and 39% less time than QC-FQL, 23% fewer FLOPs and 47% less time than DQC, and 72% fewer FLOPs and 74% less time than CGQ. ReBRAC with its default delayed actor update remains 15% faster than ALBUM per step while using twice the parameters. Figure 6 places each method by its average success rate and its cost per step, and ALBUM lies on the Pareto frontier under both task sets.

Method	Params	GFLOP/step	ms/step	Q/V nets	Peak GPU (MiB)
ReBRAC	4.86M	**3.67**	**0.99**	4	132
ALBUM (ours)	**2.43M**	3.82	1.17	**2**	**78**
ReBRAC*	4.86M	5.33	1.19	4	130
IQL	4.86M	4.12	1.28	5	134
QC-FQL	5.00M	6.41	1.93	4	130
NFQL	4.87M	6.22	2.06	4	130
FQL	4.87M	6.64	2.11	4	128
DQC	6.69M	4.96	2.22	7	165
CGQ	10.05M	13.79	4.59	8	265

Table 7: Training cost per gradient step on antmaze-large-navigate. Q/V nets counts every value and critic network the agent holds, ensemble members and target copies included. Best in each column is bold.

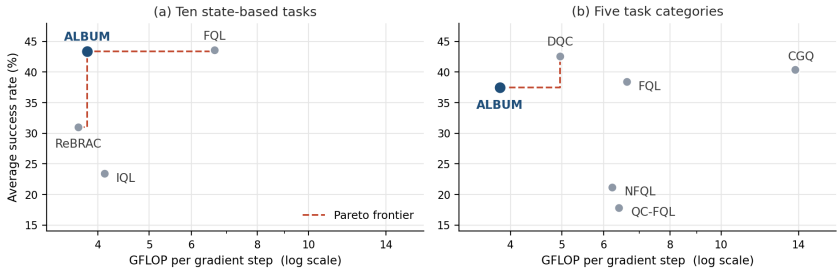

Figure 6: Average success rate against training cost per gradient step. Left: the ten state-based tasks of Table 1, for the methods measured in Table 7. Right: the five task categories of Table 6. Cost is measured on antmaze-large-navigate with each method in its default configuration, and the dashed line marks the Pareto frontier. ALBUM lies on the frontier in both, matching the average of FQL at 42% fewer FLOPs per step.

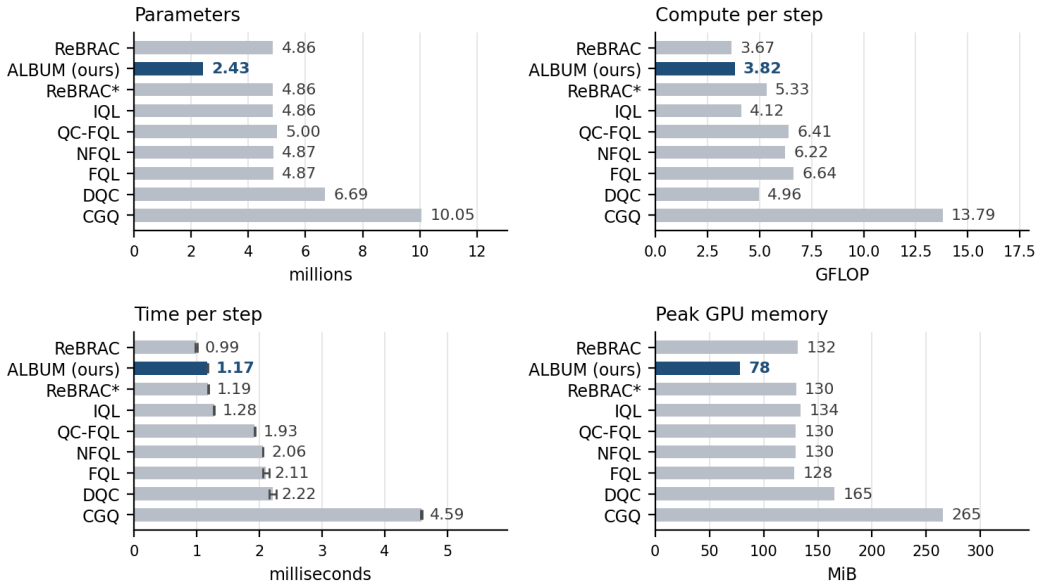

Figure 7: Computational cost comparison regarding the number of network parameters, GFLOP per step, wall-clock time per step, the number of critic networks, and peak GPU memory usage.

L.4 Violating the Necessary Condition

Proposition 5.1 states a necessary condition on the coefficients for the update to stay bounded along the common shift direction, and it carries over to parameterizations whose outputs admit such a shift, including the networks of our experiments. The default coefficients of Table 3 are chosen to satisfy it, with $\lambda_B(1-\gamma)+2\lambda_K = 2.0125$ against $\omega_Q+\omega_V = 0.15$. We evaluate the condition with $\lambda_K = 0$, which violates it at the default coefficients. The resulting critic statistics and policy performance are reported in Table 8.

	`antmaze-large`		`scene`		`puzzle-3x3`	
λ_K	0	1	0	1	0	1
$\lambda_B(1-\gamma)+2\lambda_K$	0.013	2.01	0.013	2.01	0.013	2.01
$V_{\max}$	-2.2×10^6	0.6	-4.0×10^5	-15	-5.2×10^5	-40
V_{mean}	-5.9×10^6	-136	-1.3×10^6	-285	-8.6×10^5	-369
$V_{\min}$	-5.9×10^6	-199	-1.3×10^6	-478	-8.6×10^5	-521
$V_{\max}-V_{\text{mean}}$	3.7×10^6	137	9.3×10^5	271	3.4×10^5	328
$g_B \le 0$ (%)	1	65	0	64	1	68
Success (%)	0.5	92.5	1.0	93.2	0.5	93.0

Table 8: Critic statistics at 1M steps with $\lambda_B{=}\lambda_E{=}2.5$ and $K{=}10$ for $\lambda_K \in \{0,1\}$. The first row reports the left side of equation 7, whose right side is $\omega_Q{+}\omega_V = 0.15$ throughout. Entries in red violate the condition. Setting $\lambda_K{=}0$ produces extreme downward drift in V, accompanied by severe violations of the Bellman constraints and near-zero success, whereas $\lambda_K{=}1$ satisfies the condition and avoids this behavior.

With $\lambda_K = 0$, the critic diverges on all three tasks. On `antmaze-large-navigate`, for example, the mean value reaches -5.9×10^6, more than four orders of magnitude below the physical lower bound $-1/(1-\gamma) = -200$ induced by the per-step reward of -1. The corresponding $g_B \le 0$ rate is only 1%. This behavior matches the descent direction identified in the proof of Proposition 5.1.

L.5 Rollout Horizon K and Coefficient λ_{K}

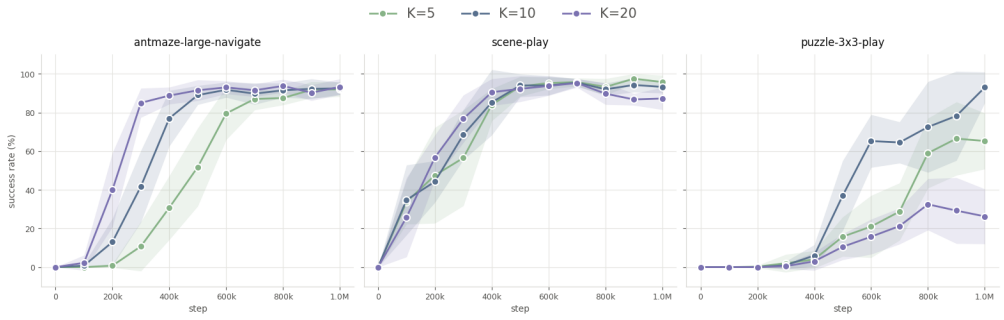

Figure 8: Rollout horizon K at the default λ_{K}. Longer horizons help on goal-directed data and hurt on play data.

	antmaze-large-navigate	scene-play	puzzle-3x3-play
K=5	88.2 $\pm$4.4	**95.8** $\pm$3.5	65.2 $\pm$14.5
K=10	92.5 $\pm$3.1	93.2 $\pm$3.2	**93.0** $\pm$7.9
K=20	**93.5** $\pm$4.2	87.2 $\pm$5.7	26.2 $\pm$14.2

Table 9: Success rate (%) at the final evaluation, 8 seeds $\times$ 50 episodes; $\pm$ std across seeds.

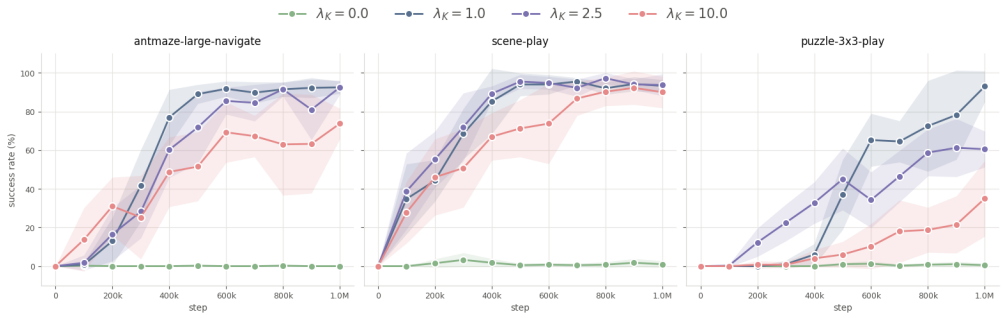

Figure 9: Rollout coefficient λ_{K} at $K = 10$. $\lambda_{\mathrm{K}} = 1$ is the only setting that exceeds 90% success on all three tasks, and $\lambda_{\mathrm{K}} = 0$ fails on all of them.

	antmaze-large-navigate	scene-play	puzzle-3x3-play
$\lambda_K = 0.0$	0.5 $\pm$0.9	1.0 $\pm$1.7	0.5 $\pm$0.9
$\lambda_K = 1.0$	**92.5** $\pm$3.1	93.2 $\pm$3.2	**93.0** $\pm$7.9
$\lambda_K = 2.5$	79.5 $\pm$26.2	**93.8** $\pm$5.6	60.5 $\pm$9.3
$\lambda_K = 10.0$	73.0 $\pm$18.6	90.0 $\pm$8.1	35.0 $\pm$19.5

Table 10: Success rate (%) at the final evaluation, 8 seeds $\times$ 50 episodes; $\pm$ std across seeds.

Figure 8 and Table 9 vary the rollout horizon K. On `antmaze-large-navigate` a longer horizon accelerates learning, with $K = 20$ reaching 50% success at 300k steps against 500k for $K = 5$, and the final success rate is unchanged. This matches Theorem 4.5, under which a longer horizon never slows the idealized iteration and contracts at γ^K where the K-step term is selected. On `puzzle-3x3-play` the final success rate drops to 26.2% at $K = 20$ against 93.0% at $K = 10$, which the idealized iteration does not predict. The source lies outside that analysis, in the sampled constraint and the function approximation.

Figure 9 and Table 10 vary λ_{K}. Setting $\lambda_{\mathrm{K}} = 0$ fails on all three tasks, consistent with Proposition 5.1 and Appendix L.6. $\lambda_{\mathrm{K}} = 1$ is the best setting on `antmaze-large-navigate` and `puzzle-3x3-play`, and `scene-play` attains 93.8 at $\lambda_{\mathrm{K}} = 2.5$ against 93.2 at 1. Table 3 adopts $K = 10$ and $\lambda_{\mathrm{K}} = 1$.

L.6 Additional Tuning under $\lambda_K = 0$

Environment	$\lambda_B{=}\lambda_E$	λ_K	V_{max}	V_{mean}	V_{min}	Success (%)
`antmaze-large-navigate`	2.5	0	$-2.2{\times}10^6$	$-5.9{\times}10^6$	$-5.9{\times}10^6$	0.5 ± 0.9
	25	0	-207.7	-973.8	-982.0	0.0 ± 0.0
	250	0	1.7	-163.3	-168.4	0.2 ± 0.7
	2500	0	-2.1	-29.1	-36.0	4.2 ± 2.7
	2.5	1	0.6	-136.2	-199.4	$\mathbf{92.5} \pm 3.1$
`scene-play`	2.5	0	$-4.0{\times}10^5$	$-1.3{\times}10^6$	$-1.3{\times}10^6$	1.0 ± 1.9
	25	0	-947.4	-3222.3	-3236.7	1.2 ± 1.5
	250	0	-84.2	-374.4	-393.5	1.5 ± 1.8
	2500	0	-0.5	-43.1	-63.7	2.0 ± 2.4
	2.5	1	-14.7	-285.4	-477.5	$\mathbf{93.2} \pm 3.4$
`puzzle-3x3-play`	2.5	0	$-5.2{\times}10^5$	$-8.6{\times}10^5$	$-8.6{\times}10^5$	0.5 ± 0.9
	25	0	-3998.5	-7678.6	-7726.9	0.2 ± 0.7
	250	0	-344.7	-573.7	-587.7	0.5 ± 0.9
	2500	0	-22.4	-60.7	-75.8	0.0 ± 0.0
	2.5	1	-40.2	-368.6	-520.5	$\mathbf{93.0} \pm 8.5$

Table 11: Raising λ_B past the value $(\omega_Q + \omega_V)/(1-\gamma) = 30$ required by Proposition 5.1 at $\lambda_K = 0$, with the rollout term removed, 1M steps, 8 seeds. The last row of each group is the default setting. The values become bounded once the condition holds, and no setting without the rollout term exceeds 4.2% on any of the three tasks.

$\lambda_B{=}\lambda_E$	λ_K	V_{max}	V_{mean}	V_{min}	$V_{max}{-}V_{mean}$	$V_{max}{-}V_{min}$	Success (%)
2.5	0	$6.1{\times}10^3$	$-4.4{\times}10^6$	$-4.5{\times}10^6$	$4.4{\times}10^6$	$4.6{\times}10^6$	8.5 ± 5.6
1000	0	0.3	-16.9	-24.0	17.2	24.3	66.0 ± 15.2
10000	0	0.4	-6.1	-8.8	6.4	9.1	49.5 ± 14.1
2.5	1	1.4	-63.1	-106.5	64.5	107.9	$\mathbf{89.8} \pm 6.5$

Table 12: `cube-single-play`, task 2, 8 seeds. Raising λ_B bounds the values and lifts success to 66.0%, but compresses the range they span and never reaches the default; pushing it further to 10^4 compresses the range again and costs performance.

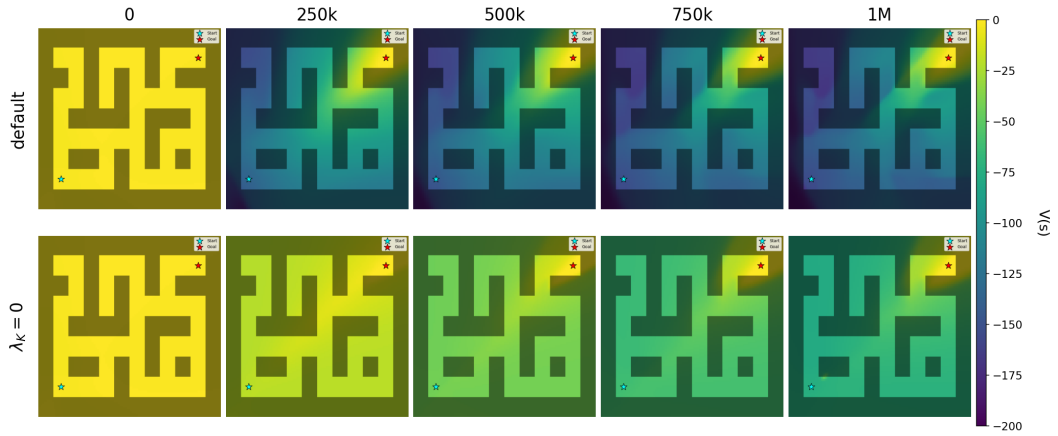

Figure 10: Value maps of V_ϕ on `antmaze-large-navigate` at [250k, 500k, 750k, and 1M] training steps, for the default setting ($\lambda_K = 1$, top) and for $\lambda_K = 0$ with $\lambda_B{=}\lambda_E = 1000$ (bottom). Each cell shows V_ϕ evaluated at the cell position with all other state coordinates set to zero, and every map uses this same input and the same color scale $[-1/(1-\gamma), 0]$. The marker indicates the start and goal.

At $\lambda_{\mathrm{K}} = 0$, equation 7 requires $\lambda_{\mathrm{B}} \geq (\omega_Q + \omega_V)/(1-\gamma) = 30$. Table 11 places this threshold between $\lambda_{\mathrm{B}} = 25$ and $\lambda_{\mathrm{B}} = 250$. At $\lambda_{\mathrm{B}} = 25$, $V_{\min}$ lies below the range of $V^*_{\mathcal{D}}$ in Table 17 on all three tasks, and from $\lambda_{\mathrm{B}} = 250$ upward it lies inside this range.

Boundedness does not restore performance. No setting with $\lambda_{\mathrm{K}} = 0$ exceeds 4.2% on the three tasks, and `cube-single-play` reaches 66.0% at $\lambda_{\mathrm{B}} = 1000$ against 89.8% at the default (Table 12). The range $V_{\max} - V_{\min}$ contracts as λ_{B} grows, from 170.1 at $\lambda_{\mathrm{B}} = 250$ to 33.9 at $\lambda_{\mathrm{B}} = 2500$ on `antmaze-large-navigate` and from 24.3 to 9.1 on `cube-single-play`, against 200.0 and 107.9 at the default.

Figure 10 compares V_ϕ on `antmaze-large-navigate` under the default setting ($\lambda_{\mathrm{K}} = 1$) and under $\lambda_{\mathrm{K}} = 0$ with $\lambda_{\mathrm{B}}{=}\lambda_{\mathrm{E}} = 1000$. Under the default setting, values spread from the goal across the maze from 250k steps onward. In the bounded setting without the rollout term, V_ϕ remains nearly uniform across the maze through 1M steps.

L.7 ALBUM Critic with Generative Policies

In this section, we examine whether the critic contribution observed in the main experiments persists when the policy is replaced by a generative policy. The main experiments use DDPG+BC to isolate the effect of the critic under a common Gaussian policy, leaving open whether it extends to generative policy classes. To assess this, we replace the IQL (Kostrikov et al., 2022) critic in IFQL (Park et al., 2025b) with the ALBUM critic while keeping its generative policy and action extraction procedure unchanged.

IFQL trains its IQL critic independently of the policy, and IQL approximates the in-sample maximum through expectile regression, so the ALBUM critic, which also targets the in-sample maximum, replaces it directly. This comparison evaluates ALBUM and IQL under the same generative policy and therefore isolates the effect of replacing the critic.

Environment	Task	IFQL	IFQL-ALBUM	Δ	ALBUM
antmaze-large-navigate	1*	24	25.5 ±14.7	+1.5	**92.5** ±3.1
	2	8	45.0 ±5.4	+37.0	**80.2** ±6.0
	3	52	51.0 ±11.4	−1.0	**91.8** ±2.7
	4	18	16.5 ±11.5	−1.5	**92.5** ±4.0
	5	38	26.5 ±21.2	−11.5	**92.0** ±4.7
	Average	28.0	32.9	+4.9	**89.8**
scene-play	1	98	**98.5** ±1.7	+0.5	91.8 ±4.9
	2*	0	**97.0** ±4.1	+97.0	93.2 ±3.2
	3	54	**94.5** ±6.5	+40.5	70.8 ±11.7
	4	0	54.0 ±26.8	+54.0	**66.2** ±9.5
	5	0	0.0 ±0.0	0.0	0.0 ±0.0
	Average	30.4	**68.8**	+38.4	64.4
puzzle-3x3-play	1	94	**98.5** ±1.7	+4.5	96.2 ±4.1
	2	1	**97.0** ±2.2	+96.0	77.5 ±32.6
	3	0	**63.5** ±19.3	+63.5	39.0 ±16.7
	4*	0	**96.0** ±3.2	+96.0	93.0 ±7.9
	5	0	**6.5** ±3.0	+6.5	0.8 ±1.4
	Average	19.0	**72.3**	+53.3	61.2

Table 13: Success rate (%) of IFQL-ALBUM, which replaces the IQL critic of IFQL with the ALBUM critic. IFQL results are taken from Park et al. (2025b). IFQL-ALBUM reports the mean over 4 seeds and ALBUM the mean over 8 seeds, ± the standard deviation across seeds. Δ denotes the change from IFQL in points. * denotes the default task.

The ALBUM critic raises the five-task average of IFQL from 30.4 to 68.8 on `scene-play` and from 19.0 to 72.3 on `puzzle-3x3-play`, and changes it from 28.0 to 32.9 on `antmaze-large-navigate`. The gains concentrate on the tasks that IFQL fails to solve. Of the six tasks at which IFQL reports 0, four reach between 54.0 and 97.0.

L.8 Sensitivity of Penalty Coefficients

Figure 11 and Table 14 vary $\lambda_B{=}\lambda_E$ over $\{1, 2.5, 10, 25\}$ at the default λ_K. The value 2.5 is the only setting that succeeds on all three tasks. At 1 the two play tasks fall below it with wider spreads across seeds, while `antmaze-large-navigate` is unaffected. At 10 and above every setting fails on the play tasks, and `antmaze-large-navigate` follows at 25. Table 3 adopts $\lambda_B{=}\lambda_E{=}2.5$ across every domain.

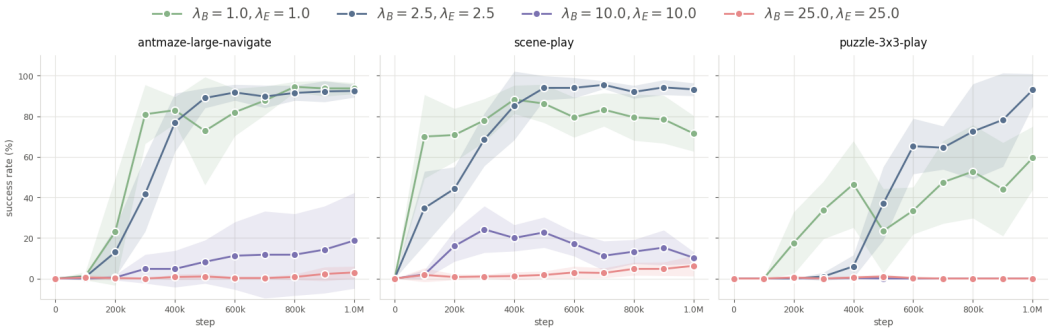

Figure 11: Penalty coefficients $\lambda_B{=}\lambda_E$ at the default λ_K.

Task	$\lambda_B{=}\lambda_E{=}1$	2.5	10	25
`antmaze-large-navigate`	**93.8** $\pm$2.7	92.5 $\pm$3.1	13.0 $\pm$26.2	0.2 $\pm$0.7
`scene-play`	71.5 $\pm$8.8	**93.2** $\pm$3.2	10.2 $\pm$2.9	6.2 $\pm$5.0
`puzzle-3x3-play`	59.5 $\pm$15.5	**93.0** $\pm$7.9	0.0 $\pm$0.0	0.0 $\pm$0.0

Table 14: Success rate (%) at the final evaluation, 8 seeds $\times$ 50 episodes, $\pm$ std across seeds.

L.9 3M Training on Long horizon and Complex Tasks

	antmaze-giant-navigate			humanoidmaze-medium-navigate		
Task	1M	2M	3M	1M	2M	3M
1	3.5 ± 3.8	20.5 ± 17.8	22.0 ± 19.2	2.0 ± 3.0	65.2 ± 11.1	82.2 ± 8.7
2	76.2 ± 7.7	81.0 ± 5.1	80.5 ± 3.7	79.0 ± 7.3	90.5 ± 5.8	91.5 ± 4.5
3	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	4.8 ± 4.0	76.8 ± 10.1	92.0 ± 6.8
4	41.2 ± 22.6	53.5 ± 23.0	47.5 ± 22.5	2.2 ± 2.7	57.0 ± 5.7	69.2 ± 8.7
5	19.5 ± 27.4	30.8 ± 34.3	34.0 ± 36.2	89.5 ± 3.3	91.5 ± 5.6	93.5 ± 4.9
Average	28.1 ± 5.1	37.1 ± 10.0	36.8 ± 11.4	35.5 ± 1.8	76.2 ± 3.7	85.7 ± 3.8

Table 15: Success rate (%) of ALBUM with an extended training budget, 8 seeds $\times$ 50 evaluation episodes. Per-task rows report the spread across seeds; the average row reports the spread of the per-seed task averages.

Table 15 extends training from 1M to 3M steps on `antmaze-giant-navigate` and `humanoidmaze-medium-navigate`. On `humanoidmaze-medium-navigate` the average rises from 35.5% at 1M steps to 85.7% at 3M steps, and tasks 1, 3, and 4 rise from below 5% to between 69.2% and 92.0%, so the 1M score on this environment reflects incomplete training. On `antmaze-giant-navigate` the average rises from 28.1% to 37.1% at 2M steps without further increase, and task 3 remains at 0.0%. The baselines of Table 1 are evaluated at 1M steps only, so these results characterize the convergence of ALBUM and are not a comparison with the baselines.

These results also do not bear on Theorem 4.5. The theorem compares the iteration in equation 3 with $K > 1$ against the same iteration with $K = 1$, and Appendix L.5 reports the corresponding comparison within ALBUM. A comparison between ALBUM and TD-based methods is outside the scope of the theorem, since their convergence rates depend on optimization hyperparameters not included in the analysis. In particular, the critic learning rate is 10^{-4} for ALBUM and 3×10^{-4} for the baselines in Table 1.

L.10 Hinge-IQL

Environment	Task	IQL	hinge-IQL
antmaze-large-navigate	1^*	**48**	12.5 ± 14.9
scene-play	2^*	**12**	**12.0** ± 7.1
puzzle-3x3-play	4^*	**2**	0.0 ± 0.0

Table 16: Adding ALBUM's K-step hinge to IQL. hinge-IQL is IQL with $\lambda_{\mathrm{K}}[\mathrm{sg}[\hat{y}_K(s,a)] - Q(s,a)]_+$ added to the critic loss and $\lambda_{\mathrm{K}}[\mathrm{sg}[\hat{y}_K(s,a)] - V(s)]_+$ added to the value loss at $\lambda_{\mathrm{K}} = 1$, where $\hat{y}_K$ is computed with IQL's value network and detached as in ALBUM, keeping IQL's AWR actor at $\alpha = 10$. Ours is the mean over 4 seeds at the final evaluation, $\pm$ the standard deviation across seeds, and the IQL column is taken from the FQL paper. The best entry in each row is bold. * indicates the default task.

In this section we isolate the contribution of the K-step rollout hinge by asking whether it improves a critic that is not derived from LBLP. We add the two penalties to the critic and value losses of IQL, keeping the hyperparameters reported in Park et al. (2025b), and run the default tasks of `antmaze-large-navigate`, `scene-play` and `puzzle-3x3-play`. As Table 16 shows, the penalty does not improve IQL on any of the three tasks, and it degrades performance substantially on `antmaze-large-navigate`. The benefit of the K-step penalties therefore does not transfer on its own. The critic in IQL is regressed onto a bootstrapped target, so a lower bound added to that loss has no descent term to work against, whereas in ALBUM the same bound enters a program whose objective descends on Q and V.

L.11 Neural Critic versus In-Sample Optimum

Environment		$V_{\max}$	V_{mean}	$V_{\min}$	$V_{\max} - V_{\text{mean}}$	$V_{\max} - V_{\min}$	$g_{\text{B}} \leq 0$ (%)	Success (%)
antmaze-large-navigate	$V^*_{\mathcal{D}}$	0.0	−156.8	−198.7	156.8	198.7	—	11.8
	learned	0.6	−136.2	−199.4	136.8	200.0	65	92.5
antmaze-giant-navigate	$V^*_{\mathcal{D}}$	0.0	−178.1	−200.0	178.1	200.0	—	10.6
	learned	−86.0	−172.4	−199.3	86.4	113.4	81	3.5
humanoidmaze-medium-navigate	$V^*_{\mathcal{D}}$	0.0	−177.1	−200.0	177.1	200.0	—	15.1
	learned	−0.3	−116.5	−157.1	116.2	156.8	67	2.0
humanoidmaze-large-navigate	$V^*_{\mathcal{D}}$	0.0	−178.6	−200.0	178.6	200.0	—	7.7
	learned	−199.5	−199.5	−199.5	0.0	0.0	50	0.0
antsoccer-arena-navigate	$V^*_{\mathcal{D}}$	0.0	−156.7	−198.7	156.7	198.7	—	10.9
	learned	−0.9	−103.6	−198.5	102.7	197.6	74	66.0
cube-single-play	$V^*_{\mathcal{D}}$	0.0	−147.7	−198.7	147.7	198.7	—	35.6
	learned	2.3	−72.0	−117.5	74.3	119.8	72	84.5
cube-double-play	$V^*_{\mathcal{D}}$	0.0	−311.8	−397.3	311.8	397.3	—	1.5
	learned	−85.3	−309.3	−377.5	224.0	292.2	54	5.2
scene-play	$V^*_{\mathcal{D}}$	0.0	−500.0	−972.3	500.0	972.3	—	4.0
	learned	−14.7	−285.4	−477.5	270.6	462.7	64	93.2
puzzle-3x3-play	$V^*_{\mathcal{D}}$	0.0	−714.0	−1309.0	714.0	1309.0	—	5.4
	learned	−40.2	−368.6	−520.5	328.4	480.3	68	93.0
puzzle-4x4-play	$V^*_{\mathcal{D}}$	−3.0	−1282.1	−2241.7	1279.1	2238.7	—	0.0
	learned	−696.1	−1031.7	−1234.3	335.5	538.2	71	11.8

Table 17: The learned value function against $V^*_{\mathcal{D}}$ on one task per environment, which is the default task except for task 1 of `antsoccer-arena-navigate` and `cube-single-play`, with the default configuration ($\lambda_{\text{B}} = \lambda_{\text{E}} = 2.5$, $\lambda_{\text{K}} = 1$, $K = 10$), 1M steps, and 8 seeds. For $V^*_{\mathcal{D}}$ the Success column reports the fraction of dataset trajectories that reach the goal. Reward scales differ across environments, so values compare within a row pair but not across them.

Under deterministic dynamics and a continuous state space, each state of $\mathcal{D}$ carries a single action, so $V^*_{\mathcal{D}}(s) = Q^*_{\mathcal{D}}(s, a) = G_{\mathcal{D}}(s, a)$, the discounted return along the dataset trajectory through s. Table 17 compares the statistics of $V^*_{\mathcal{D}}$ over $\mathcal{D}$ with those of the learned V_ϕ. The mean of V_ϕ exceeds the mean of $V^*_{\mathcal{D}}$ in nine of ten environments. The exception is `humanoidmaze-large-navigate`, with V_ϕ constant at -199.5, close to $-1/(1-\gamma) = -200$. The detached K-step hinges in equation 5 bound $V_\phi(s)$ from below, and V_ϕ shares parameters across states of different trajectories, so a value above the return of the trajectory through s is consistent with lower bounds transferred from neighboring trajectories with higher returns. Rewards are nonpositive, so this transfer does not account for the entries $V_{\max} = 0.6$ on `antmaze-large-navigate` and $V_{\max} = 2.3$ on `cube-single-play`, which are attributable to function approximation error. The constraint $g_{\text{B}} \leq 0$ holds on 50% to 81% of the samples, so the learned pair is not feasible for LBLP on $\mathcal{M}_{\mathcal{D}}$.