Calibrated Reachability for State-Constrained Offline RL in Stochastic MDPs

Liz Lemma Future Detective

January 20, 2026

Abstract

Offline RL methods typically avoid distribution shift by constraining optimization to the dataset's state–action support (batch constraints), which can be overly restrictive when optimal actions are absent from the dataset. Recent work proposed *state-constrained* offline RL in deterministic MDPs, enabling out-of-distribution actions so long as they lead to in-distribution states via a reachability relation, and proved dominance over batch-constrained baselines. We push this paradigm to the modern (2026) regime where environments are stochastic and reachability must be learned with imperfect models. We introduce a probabilistic notion of reachability certification based on lower confidence bounds (LCBs) over target-reaching success probabilities, yielding a single tunable knob: a reachability confidence threshold δ. We define a pessimistic state-constrained Bellman operator that backs up only through certified reachable dataset states and is robust to model errors by pessimistically accounting for failure probability. Our main results show (i) contraction and unique fixed points for the certified operator, (ii) a performance guarantee: compared to any batch-constrained baseline that remains within the dataset state set, the certified state-constrained policy is never substantially worse, with value degradation scaling as $O(\varepsilon/(1-\gamma))$ in the false-positive (miscalibration) rate ε, and (iii) a matching lower bound demonstrating this dependence is unavoidable. We propose a practical algorithm using ensembles plus conformal calibration on model residuals to obtain certified reachability sets, and we outline experiments on D4RL under injected stochasticity (noisy dynamics / domain randomization) to validate the theory and quantify the tightness of the bounds.

Table of Contents

2. 2. Related work: offline RL (BCQ/TD3+BC/IQL/CQL), model uncertainty and pessimism (ensembles, COMBO/RAMBO), reachability/similarity (bisimulation, pseudometrics), and the deterministic state-constrained framework.

3. 3. Preliminaries and setup: MDPs, dataset-induced state set $\mathcal{S}_D$, target-reaching controllers, value bounds, and baseline batch-constrained class for comparison.

4. 4. Problem formulation: probabilistic reachability; calibrated LCBs; definition of certified feasible sets $\widehat{\text{Reach}}_\delta(s)$; conservative/pessimistic backups; statement of objectives and metrics.

5. 5. Oracle analysis (clean core): assume an oracle reachability LCB with bounded false positives; prove contraction, dominance-style comparison, and tight lower bound construction.

6. 6. From oracle to learned reachability: ensemble models for forward/inverse (or target-reaching success) prediction; conformal calibration to guarantee LCB validity; translating calibration error into policy suboptimality.

7. 7. Algorithms: Certified StaCQ (C-StaCQ) for stochastic settings; implementation details; candidate generation/retrieval for reachability; practical considerations (reward model, conservative targets, clipping).

8. 8. Experiments (recommended to strengthen contribution): stochastic D4RL variants, sensitivity to δ, empirical calibration curves vs performance, ablations on FP vs FN, and comparisons to batch-constrained and pessimistic baselines.

9. 9. Discussion and limitations: coverage assumptions, partial observability, continuous states, and what is needed for full function-approximation theory; open directions (state-only logs, k-step certification).

1 Introduction

Offline reinforcement learning (offline RL) asks us to compute a high-performing decision rule from a fixed dataset of transitions, without additional interaction. The central obstacle is distribution shift: the learned policy may select actions or visit states that are poorly represented in the dataset, in which case value estimation becomes unreliable and, in the worst case, arbitrarily wrong. A common practical response is to impose *batch constraints*, i.e., to restrict the learned policy to choose actions that are "close" to those observed in the data. This paradigm underlies a range of algorithms that limit the actor to a dataset-supported action set, penalize out-of-distribution actions, or otherwise enforce conservatism.

We adopt a different viewpoint: rather than constraining actions directly, we constrain *next states*. The dataset induces a set of observed states, and we treat this set as a region in which learning and evaluation are comparatively trustworthy. The basic desideratum is then not that the action itself resembles the behavior action, but that the *resulting transition* remains within the dataset-supported region with high probability. This leads to what we call a *state-constrained* policy class: from any dataset state, the policy must select a control whose one-step transition remains within the dataset state set with a specified success probability. The constraint is therefore formulated in the state space, but it is enforced via a controller that maps a desired next state (a "target") to an action.

This shift from batch constraints to state constraints is motivated by two considerations. First, the offline dataset typically provides more direct evidence about which *states* are well covered than about which *actions* are safe to take. In many problems, the behavior policy may be multi-modal or noisy in action space, so action-level support is a poor proxy for transition reliability. Second, and more importantly, deterministic reasoning about single actions is fragile under stochastic dynamics and model error. If we only ensure that an action is "in distribution," we do not thereby ensure that the next state is in distribution. Conversely, an action that is slightly out of distribution might still reliably transition into a well-covered region, which suggests that action-level constraints can be overly restrictive.

The difficulty is that state constraints are inherently probabilistic. Even if there exists a controller intended to move from a state s to a target s', stochasticity in the environment means that executing the prescribed action may land us elsewhere. Likewise, when the controller is learned from finite data (e.g., via inverse dynamics or goal-conditioned behavior cloning), it will make mistakes. Consequently, any deterministic guarantee of the form "choosing target s' implies $s_{t+1} = s'$" is untenable in the regimes of interest. The appropriate object is instead the *success probability*

$$p(s \to s') \;=\; \Pr\big[s_{t+1} = s' \,\big|\, s_t = s,\; a_t = g(s, s')\big],$$

and the corresponding feasibility notion "$p(s \to s') \geq \delta$" for a chosen confidence level $\delta \in (0, 1]$. In this formulation, stochasticity and controller error are not nuisances to be ignored; they are explicitly incorporated into the constraint.

However, in offline RL we do not know $p(s \to s')$ and cannot estimate it uniformly well over all pairs. The core technical question becomes: *how can we enforce probabilistic state constraints using only finite offline data, while retaining performance guarantees relative to a baseline?* Our answer is to introduce an explicit *reachability certification* step that produces, for each pair (s, s') of dataset states, a conservative lower confidence bound (LCB) $\underline{p}(s \to s')$ on $p(s \to s')$. We then define a *certified* feasible set of targets by thresholding these bounds at level δ, and we plan only over certified targets.

The certification step serves two roles. Operationally, it filters candidate targets to those that are plausibly achievable with probability at least δ. Analytically, it provides a calibrated notion of reliability: even though individual estimates may be wrong, the overall procedure controls the rate at which we mistakenly certify an unsafe transition (a false positive), with respect to a reference distribution over state pairs. We emphasize that this is not an assumption that the learned model is correct; rather, it is a requirement that the uncertainty quantification procedure is calibrated in a distributional sense. This calibration property becomes the bridge between statistical estimation and sequential decision making.

Certification alone is not sufficient, because planning with a hard constraint "$s' \in \widehat{\mathrm{Reach}}_\delta(s)$" would still be brittle: if a false positive slips through, the planner may overcommit to a transition that in fact fails, potentially pushing the agent out of the dataset-supported region and invalidating subsequent value estimates. We therefore couple certification with an explicit *pessimistic* treatment of residual uncertainty. Concretely, when we plan as if targeting s' from s, we treat the event of reaching s' as occurring with probability at most δ and allocate the remaining probability mass to an adverse outcome represented by a known lower bound $V_{\min}$. This device converts an uncertain reachability constraint into a robust Bellman backup: the planner is rewarded for selecting high-value targets, but it must pay an explicit price for the possibility of failure.

The parameter δ is the key knob controlling this tradeoff. At a high level, larger δ demands higher one-step reliability: the certified feasible set shrinks, but the policy is less likely to exit the dataset-supported region. Smaller δ enlarges the feasible set and may improve nominal performance, but it increases sensitivity to both genuine stochasticity and miscalibration. Importantly, the pessimistic backup scales the downstream influence of any target by δ, which causes the dynamic programming operator to contract more strongly as δ decreases. Thus δ simultaneously governs (i) feasibility in the environment (how often we remain within the dataset states), (ii)

conservatism in planning (how much value we attribute to the target state), and (iii) stability of the value iteration process.

The resulting perspective can be summarized as follows. We treat the dataset as specifying a trustworthy region in state space. We endow ourselves with a family of target-reaching controllers $g(s, s')$ (learned or given) that attempt to realize one-step transitions between dataset states. We then use offline data to estimate, and crucially to *certify*, which such transitions are reliable at a chosen level δ. Planning is performed over this certified transition structure, but with a built-in pessimism that anticipates failure. The output is a policy that does not merely imitate the behavior policy, but composes certified one-step skills into multi-step behavior while maintaining an explicit probabilistic link to dataset coverage.

This approach is intended to address a specific failure mode of batch-constrained methods. Batch constraints are typically phrased as restrictions on the action distribution, often enforced by generative models, nearest-neighbor filtering, or explicit penalties. Yet the safety of offline RL is ultimately about *where the system goes*, not about the action marginal per se. Under stochastic dynamics, two actions that are equally "in distribution" can have very different probabilities of leaving the dataset state set; under controller error, an action predicted to be safe can produce an out-of-support transition. State constraints, formulated in terms of one-step reachability and enforced via calibrated certification, target this issue directly.

Finally, the certified reachability lens suggests an algorithmic decomposition that is natural in offline settings: supervised learning for the controller and reachability model, calibration for uncertainty quantification, retrieval for candidate targets, and dynamic programming for planning. Each component can be improved independently (better controllers, better uncertainty estimates, better retrieval indices), while the overall method retains a clear safety-performance mechanism through δ and the pessimistic backup. This modularity is especially attractive when the state space is high-dimensional: we can operate on a dataset-indexed set of candidate next states rather than attempting to learn an unconstrained dynamics model over the entire state space.

In the remainder of the paper, we formalize this framework, define the pessimistic operator induced by certified reachability, and prove that it yields a well-behaved planning problem with explicit robustness to false-positive certifications. We also show that the dependence on the miscalibration rate is unavoidable in the worst case, clarifying what can and cannot be guaranteed in offline control without additional coverage assumptions.

2 Related work

Offline reinforcement learning has developed a large set of algorithmic responses to the distribution-shift pathology that arises when a learned policy queries values outside the support of a fixed dataset. A first line of work enforces *action-side* constraints or regularization, typically by restricting the learned policy to remain close to the behavior distribution. Batch-Constrained Q-learning (BCQ) **?** filters candidate actions through a generative model and then performs approximate maximization over those actions; TD3+BC **?** uses an explicit behavior-cloning penalty added to an actor-critic objective; and related variants enforce nearest-neighbor or model-based action constraints. These methods can be viewed as attempting to control the discrepancy between the learned policy and the behavior policy in action space (or, more precisely, in the joint space of states and actions appearing in the dataset), with the implicit hope that such control prevents the induced state distribution from drifting into poorly covered regions.

A second family, often described as *value-side conservatism*, modifies the critic objective so that Q-values for out-of-distribution actions are underestimated. Conservative Q-learning (CQL) **?** adds a regularizer that penalizes large Q-values on actions sampled from a broad proposal distribution, which yields a lower bound flavor under suitable realizability and sampling assumptions. Implicit Q-learning (IQL) **?** avoids explicit policy constraints by performing a particular asymmetric regression that tends to preserve conservative values and can be paired with advantage-weighted regression for policy extraction. More generally, a substantial portion of the offline RL literature can be interpreted as producing a pessimistic critic, either explicitly through penalties and constraints or implicitly through the choice of objectives and targets **?**. Our work is aligned with this perspective in that our planning operator is pessimistic by design; however, our mechanism for pessimism is tied to a *state-level* feasibility notion rather than an action-level divergence.

Model-based offline RL provides a complementary route to conservatism by exposing uncertainty through explicit dynamics modeling. Approaches such as MOPO **?**, MOReL **?**, COMBO **?**, and RAMBO **?** train dynamics models (often as ensembles) and then penalize or terminate rollouts in regions of high model uncertainty. While these methods differ in how they quantify and operationalize uncertainty, a common template is: (i) learn a model from the dataset, (ii) identify where the model is unreliable (via disagreement, likelihood, or uncertainty estimates), and (iii) plan pessimistically by either restricting rollouts or subtracting uncertainty-dependent penalties. Our framework shares with this literature the central role of pessimism and the use of uncertainty quantification, but it differs in the object being certified. Rather than certifying the accuracy of a *global* transition model (or the safety of multi-step imaginary rollouts), we certify *one-step target transitions between dataset states* through a controller, and we insert pessimism directly

into a Bellman backup that accounts for the probability of failing to realize the target. This yields a planning problem on a dataset-indexed graph of candidate next states, which can be advantageous when full dynamics modeling is difficult or when one wishes to avoid compounding model error in long synthetic rollouts.

Uncertainty estimation in offline RL has been studied through ensembles, Bayesian methods, and distributional critics, and these tools frequently appear as subroutines for conservative control **??**. In the offline setting, however, uncertainty must be tied to a statistical guarantee of some form if it is to support worst-case comparisons. Several works pursue calibrated uncertainty through conformal prediction or other finite-sample tools **?**, though typically not specialized to the sequential reachability events that are relevant for keeping the agent within a dataset-supported region. In our setting, the certification object is a lower confidence bound on a Bernoulli success event of the form "executing the controller intended to reach s' from s actually lands in s'," and our subsequent analysis uses only a miscalibration-rate control (false-positive rate) with respect to a reference distribution. This emphasis separates *calibration* from *accuracy*: we do not require uniformly small error in probability estimates, but rather a bound on the frequency with which the bound is optimistic.

The present work is also connected to research on state abstractions and similarity, in which one seeks to quantify when two states are "close" in terms of their downstream control-relevant behavior. Bisimulation relations and bisimulation metrics formalize behavioral equivalence or near-equivalence of states via reward and transition similarity **??**. More recent representation-learning approaches aim to learn embeddings that preserve such structure and can support planning or generalization **?**. While our algorithmic core is not an abstraction method per se, it relies on a dataset-indexed neighborhood structure over states: we must retrieve plausible target states and reason about reachability between nearby states. Bisimulation-style thinking motivates why such retrieval may be well-behaved when the embedding respects transition structure, and it clarifies that "nearest neighbors" in raw observation space are not necessarily the right candidates. At the same time, our certification step is deliberately agnostic to the choice of retrieval metric: we may use any candidate generator (e.g., kNN in a learned embedding), and then rely on calibrated reachability bounds to filter candidates.

Goal-conditioned RL and skill learning provide another relevant foundation. Controllers of the form $g(s, s')$ are closely related to goal-conditioned policies $g(s, \text{goal})$ trained by supervised imitation or by relabeling methods such as hindsight experience replay **?**, as well as to inverse dynamics models used to infer actions that connect successive states **?**. In many robotic and control applications, the ability to propose an action that aims for a target observation is an available primitive, either via a learned inverse model or via an existing low-level controller. Our use of such a controller is concep-

tually simple: it turns the offline problem into selecting *targets* rather than actions, thereby moving the support constraint into state space. The novelty is not the existence of goal-conditioned controllers, but the combination of (i) a certification step for one-step success probabilities and (ii) a pessimistic Bellman backup that remains meaningful under miscalibration.

Finally, we place our work relative to deterministic or hard state-constrained formulations. Constraint satisfaction in RL is commonly studied through constrained MDPs (CMDPs), Lyapunov methods, and safe RL **??**. Such formulations typically constrain expected cumulative costs or enforce almost-sure safety with respect to a known safe set, often requiring either online interaction for verification or a known model. Offline safe RL is strictly harder: without interaction, a constraint that depends on unobserved transitions is not verifiable, and worst-case guarantees can become vacuous without explicit coverage assumptions. Some deterministic offline approaches therefore impose hard support constraints such as "stay within the dataset" by restricting actions to those observed (or near observed) in each state; these are natural but can be overly conservative, and they do not directly encode the probability of leaving the dataset state set under stochastic dynamics. Our framework can be read as a probabilistic relaxation of deterministic dataset support constraints: we require only that the *one-step* transition remains in the dataset-supported region with probability at least δ, and we incorporate the remaining probability mass pessimistically. This perspective is especially relevant when the environment is stochastic or when the controller is imperfect, in which case deterministic next-state constraints are inappropriate abstractions.

In summary, we build on offline RL conservatism, uncertainty quantification, and goal-conditioned control, while shifting the constraint locus from actions to one-step reachable dataset states. The next section formalizes the MDP and dataset-induced state set, introduces the target-reaching controller interface, and defines the value bounds and baseline class used in our subsequent comparison theorems.

3 Preliminaries and setup

We work with a discounted Markov decision process (MDP) $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ with discount factor $\gamma \in (0, 1)$. The state space $\mathcal{S}$ and action space $\mathcal{A}$ may be large in applications; for our basic definitions and several theory statements we consider $\mathcal{S}$ finite. The transition kernel is $P(\cdot \mid s, a)$, and the reward function is either of the form $r(s, a)$ or $r(s, a, s')$; in either case we assume bounded rewards,

$$|r(s, a)| \le r_{\max} \qquad \text{for all } (s, a) \in \mathcal{S} \times \mathcal{A},$$

(or the analogous bound for $r(s, a, s')$). For a (possibly stochastic) policy $\pi(\cdot \mid s)$, we write the discounted return from state s as

$$V^\pi(s) \;:=\; \mathbb{E}^\pi\Big[\sum_{t=0}^{\infty} \gamma^t\, r(s_t, a_t) \,\Big|\, s_0 = s\Big],$$

where the expectation is over the trajectory induced by $a_t \sim \pi(\cdot \mid s_t)$ and $s_{t+1} \sim P(\cdot \mid s_t, a_t)$. Bounded rewards imply the uniform value bound

$$|V^\pi(s)| \le \frac{r_{\max}}{1-\gamma} \qquad \text{for all } \pi \text{ and } s \in \mathcal{S},$$

and we will frequently use a known lower bound $V_{\min}$ satisfying $V_{\min} \le V^\pi(s)$ for all s and all π under consideration; a canonical choice is $V_{\min} = -r_{\max}/(1-\gamma)$.

Offline dataset and dataset-induced state set. We assume access to an offline dataset of transitions

$$D = \{(s_i, a_i, r_i, s_i')\}_{i=1}^n,$$

generated by some unknown behavior policy interacting with the MDP. We do not assume we can sample new transitions from M after collecting D. The dataset induces a finite subset of states,

$$\mathcal{S}_D \;:=\; \{s : \exists i \text{ with } s_i = s \text{ or } s_i' = s\} \subseteq \mathcal{S},$$

which we treat as the set of "supported" states available for planning. In continuous-state problems, $\mathcal{S}_D$ is a finite collection of observations; in the analysis it is convenient to view each element of $\mathcal{S}_D$ as an atomic node, with the understanding that function approximation may impose additional structure. We emphasize that $\mathcal{S}_D$ is generally not closed under the true dynamics: even if $s \in \mathcal{S}_D$, an action selected at s may transition to $s' \notin \mathcal{S}_D$. Our goal is therefore not to assert that $\mathcal{S}_D$ is invariant, but to control the probability of leaving $\mathcal{S}_D$ under the learned policy.

It is also useful to record the dataset-supported action set at each dataset state,

$$\mathcal{A}_D(s) \;:=\; \{a \in \mathcal{A} : \exists i \text{ with } s_i = s,\; a_i = a\},$$

and the empirical distribution over (s, a) pairs induced by D. While our proposed method will not directly constrain actions to $\mathcal{A}_D(s)$, these objects serve as a point of comparison to standard batch-constrained baselines.

Target-reaching controllers and induced success events. A central interface in our setup is a target-reaching controller (or inverse model) that

maps a current state and a desired next state to an action. Concretely, we assume access to a (possibly learned) mapping

$$g : \mathcal{S}_D \times \mathcal{S}_D \to \mathcal{A}, \qquad (s, s') \mapsto g(s, s'),$$

intended to produce an action that causes a one-step transition from s to s'. The controller may be trained from D as an inverse dynamics model, or may be provided by a lower-level system. For a fixed pair $(s, s') \in \mathcal{S}_D^2$, the execution of $a = g(s, s')$ in the environment induces a Bernoulli success event,

$$\mathbf{1}\{s_{t+1} = s'\} \quad \text{given } s_t = s,\ a_t = g(s, s'),$$

with associated success probability

$$p(s \to s') \ := \ \Pr[s_{t+1} = s' \mid s_t = s,\ a_t = g(s, s')].$$

In deterministic settings with a perfect controller, $p(s \to s')$ may be close to 1 for appropriate targets; in stochastic environments or with imperfect inverse models, $p(s \to s')$ can be substantially smaller and can vary strongly with both s and s'. In addition to reachability, we will sometimes require an estimate of the expected immediate reward obtained when attempting a target transition. We therefore define the one-step reward induced by the controller,

$$r_g(s, s') \ := \ \mathbb{E}\big[r(s, g(s, s')) \mid s_t = s\big],$$

(or the analogous quantity if rewards depend on s_{t+1}), and we allow the algorithm to employ an estimate $\hat{r}(s, s')$ of $r_g(s, s')$ learned from D.

The interpretation of g is that we plan over targets $s' \in \mathcal{S}_D$ rather than over primitive actions. Formally, selecting a target s' at state s induces an action $a = g(s, s')$ and therefore an environment transition distribution $P(\cdot \mid s, g(s, s'))$. This reparameterization moves the usual offline "support" question from actions to *reachable dataset states*: we will prefer targets s' that are plausibly realized by the controller while staying within the region indexed by $\mathcal{S}_D$.

Values restricted to dataset states. Because the output policy will be defined only on $\mathcal{S}_D$ (and because our performance comparisons are stated for $s \in \mathcal{S}_D$), we treat value functions as mappings $V : \mathcal{S}_D \to \mathbb{R}$. When a policy remains within $\mathcal{S}_D$ with high probability, such value functions can be interpreted as approximately capturing the relevant long-horizon consequences. Nevertheless, we must account for the possibility of transitioning to $\mathcal{S} \setminus \mathcal{S}_D$, in which case V is undefined. Our analysis will handle this through pessimistic bounding: whenever there is residual probability of leaving the supported region, we conservatively lower-bound the continuation value by $V_{\min}$. This device is purely analytic at this stage, and it motivates the later definition of a pessimistic Bellman backup in which the continuation value is a convex combination of $V(s')$ and $V_{\min}$.

Baseline class for comparison: batch-constrained policies. To state meaningful improvement guarantees in offline RL, it is standard to compare against a baseline that is itself constrained to remain within the dataset support. We formalize this via a batch-constrained policy class. Let π_{BC} denote any policy satisfying two properties: (i) *action support*—for each $s \in \mathcal{S}_D$, $\pi_{\mathrm{BC}}(\cdot \mid s)$ is supported on $\mathcal{A}_D(s)$ (or on a prescribed relaxation thereof, e.g. nearest-neighbor actions); and (ii) *dataset-state invariance*—starting from any $s \in \mathcal{S}_D$, the one-step next state under π_{BC} lies in $\mathcal{S}_D$ almost surely, i.e.,

$$\Pr[s_{t+1} \in \mathcal{S}_D \mid s_t = s,\ a_t \sim \pi_{\mathrm{BC}}(\cdot \mid s)] = 1 \qquad \text{for all } s \in \mathcal{S}_D.$$

The second condition is an explicit assumption ensuring that the baseline does not "fall off" the dataset state set; it is satisfied, for example, if the environment is effectively deterministic on the dataset support and the baseline selects only actions that were observed to lead to dataset states, or if $\mathcal{S}_D$ is defined as a closed region under the baseline dynamics. We stress that this invariance is used only to define a safe point of comparison; our learned policy will be permitted to have a nonzero probability of leaving $\mathcal{S}_D$, but it will do so in a controlled manner.

This baseline perspective also clarifies the role of the controller interface. A batch-constrained method operates in action space and typically enforces $\pi(\cdot \mid s) \approx \pi_\beta(\cdot \mid s)$ for the behavior policy π_β (or a model thereof). In contrast, our method will operate in target space: at each dataset state s it will select a candidate $s' \in \mathcal{S}_D$ and execute $g(s, s')$. The relevant comparison is then between policies that remain in $\mathcal{S}_D$ because they imitate dataset actions and policies that remain in $\mathcal{S}_D$ because they can *certify* one-step transitions between dataset states. The remainder of the paper specifies how we quantify such certification through conservative lower bounds on the success probabilities $p(s \to s')$, and how we incorporate the resulting feasibility notion into a pessimistic dynamic programming operator.

4 Problem formulation: probabilistic reachability and certified planning

Our goal is to formalize an offline control objective in which we (a) plan *over dataset states* as targets, (b) restrict ourselves to targets that are *likely reachable in one step* under the controller interface, and (c) account for the possibility that any certification procedure may occasionally admit unsafe targets. The central object is therefore a one-step reachability predicate that is inherently probabilistic and must be handled with explicit uncertainty quantification.

Probabilistic one-step reachability under a target controller. Fix the controller $g : \mathcal{S}_D \times \mathcal{S}_D \to \mathcal{A}$. For each ordered pair $(s, s') \in \mathcal{S}_D^2$, recall the induced success probability

$$p(s \to s') \;:=\; \Pr[s_{t+1} = s' \mid s_t = s,\; a_t = g(s, s')].$$

We interpret $p(s \to s')$ as the reliability of the directed "edge" $s \to s'$ when we attempt to realize it via g. In general, even if $s' \in \mathcal{S}_D$, the controller can fail and transition to some other state $\tilde{s} \neq s'$, potentially outside $\mathcal{S}_D$. Accordingly, we cannot treat $\mathcal{S}_D$ as closed, and we must explicitly reason about a nonzero failure probability.

We fix a confidence threshold $\delta \in (0, 1]$ and declare a target s' *δ-feasible* from s if $p(s \to s') \geq \delta$. If we knew all such probabilities, then the feasible target set would be

$$\mathrm{Reach}_\delta(s) \;:=\; \{s' \in \mathcal{S}_D :\; p(s \to s') \geq \delta\}.$$

The constraint $p(s \to s') \geq \delta$ is a one-step probabilistic safety condition: it ensures that, whenever we attempt $s \to s'$, with probability at least δ we land at a dataset state (indeed, at the designated s'). What remains unspecified is the outcome on the complement event of probability at most $1 - \delta$; our subsequent planning rule will treat this complement pessimistically.

Lower confidence bounds and calibrated false positives. In offline learning we do not know $p(s \to s')$, and we therefore rely on a statistical procedure which outputs a lower confidence bound (LCB)

$$\underline{p}(s \to s') \in [0, 1] \qquad \text{for each } (s, s') \in \mathcal{S}_D^2.$$

Because the pairs (s, s') are numerous (and typically structured), we do not assume a uniform, simultaneous guarantee over all pairs. Instead, we adopt a distributional calibration condition with respect to a user-chosen reference distribution μ over pairs in $\mathcal{S}_D^2$. Concretely, for a target miscalibration level $\varepsilon \in (0, 1)$, we require that

$$\Pr_{(s,s') \sim \mu} \left[\underline{p}(s \to s') > p(s \to s')\right] \;\leq\; \varepsilon. \tag{1}$$

We emphasize the direction of the inequality: the event $\{\underline{p} > p\}$ is a *false positive* in the sense that the bound is overly optimistic. Condition (1) asserts that such optimism occurs rarely under μ. This formulation aligns with split conformal prediction and related distribution-free calibration methods, where coverage (or miscoverage) is guaranteed for new draws from the same reference distribution used for calibration.

Given $\underline{p}$, we define the *certified feasible set*

$$\widehat{\mathrm{Reach}}_\delta(s) \;:=\; \{s' \in \mathcal{S}_D :\; \underline{p}(s \to s') \geq \delta\}.$$

This is the set of targets our algorithm is permitted to select. We will impose a mild coverage condition ensuring that the set is nonempty:

$$\widehat{\mathrm{Reach}}_\delta(s) \neq \emptyset \qquad \text{for all } s \in \mathcal{S}_D. \tag{2}$$

Condition (2) prevents degenerate behavior in which the constraint is infeasible; algorithmically, it can be enforced by decreasing δ, enlarging the candidate target pool, or including a fallback self-loop target if such an edge can be certified.

State-constrained policy class induced by certified reachability. We consider policies that act by selecting a target. Formally, a (possibly stochastic) target-selection rule $\rho(\cdot \mid s)$ induces a primitive-action policy via

$$a_t = g(s_t, s_{t+1}^{\mathrm{tar}}), \qquad s_{t+1}^{\mathrm{tar}} \sim \rho(\cdot \mid s_t).$$

We define the *certified state-constrained* policy class at level δ as the set of such policies satisfying

$$\operatorname{supp} \rho(\cdot \mid s) \subseteq \widehat{\mathrm{Reach}}_\delta(s) \qquad \text{for all } s \in \mathcal{S}_D.$$

In words, from each dataset state s the policy may only attempt transitions to targets s' whose certified LCB exceeds δ. This is the basic device by which we control distribution shift: we do not constrain actions to those in the batch, but we constrain *next-state intent* to dataset states that are plausibly achievable in one step.

Conservative backups via value pessimism. Even under the constraint $\underline{p}(s \to s') \geq \delta$, we must plan with the understanding that (i) certification can be wrong on a small fraction of pairs, and (ii) even for correct certificates the success probability is only lower-bounded by δ, leaving a residual failure probability. We therefore embed the constraint into a pessimistic dynamic programming backup.

Let $V : \mathcal{S}_D \to \mathbb{R}$ be any bounded value function, and let $V_{\min}$ be a known lower bound on continuation values (e.g. $V_{\min} = -r_{\max}/(1-\gamma)$). For planning we also require an estimate $\hat{r}(s, s')$ of the expected immediate reward obtained when attempting target s' from s using g. The key modeling choice is to treat the event of *not* reaching s' as adversarial, and to lower bound the continuation value on that event by $V_{\min}$. This leads to the pessimistic backup

$$(\widehat{T}_\delta V)(s) \;:=\; \max_{s' \in \widehat{\mathrm{Reach}}_\delta(s)} \Big(\hat{r}(s, s') + \gamma\big(\delta V(s') + (1-\delta) V_{\min}\big) \Big). \tag{3}$$

The operator (3) should be read as follows: upon selecting target s', we credit ourselves with at most a δ fraction of the continuation value at s' and

assign the remaining mass to the worst-case lower bound $V_{\min}$. We stress that this is *not* a claim about the true transition distribution, but rather a conservative surrogate used for offline planning in the presence of certification and modeling error. In particular, if the actual success probability exceeds δ then (3) may be loose, but it is designed to be robust to both limited success rates and occasional false positives.

Objectives and evaluation metrics. The algorithmic objective is to compute a value function $\hat{V}_\delta$ as an approximate fixed point of (3) and to output a greedy target policy

$$\hat{s}'(s) \in \arg \max_{s' \in \widehat{\mathrm{Reach}}_\delta(s)} \Big(\hat{r}(s,s') + \gamma(\delta \hat{V}_\delta(s') + (1-\delta)V_{\min})\Big), \qquad \hat{\pi}_\delta(s) := g(s, \hat{s}'(s)),$$

with the understanding that ties may be broken arbitrarily and that in function-approximation settings the maximization may be approximate.

We evaluate the resulting policy along three axes. First, we consider the discounted return $V^{\hat{\pi}_\delta}(s)$ for each $s \in \mathcal{S}_D$ (or averaged over an initial distribution supported on $\mathcal{S}_D$). Second, we consider *safety with respect to dataset support*, quantified implicitly by the one-step success constraint $\underline{p}(s \to s') \geq \delta$ and explicitly by the pessimistic handling of failure through $V_{\min}$. Third, we quantify robustness to miscalibration through the false-positive rate ε in (1); this is the parameter that will govern our comparison to batch-constrained baselines.

Finally, we measure performance relative to a batch-constrained baseline policy π_{BC} whose one-step transitions remain in $\mathcal{S}_D$ almost surely. The comparison is meaningful because π_{BC} serves as a safe reference point under the same dataset support, while $\hat{\pi}_\delta$ may improve return by exploiting certified transitions among dataset states that are not realizable by strict action support constraints. Any remaining discrepancy due to imperfect reward models, approximate DP, or imperfect controllers will be aggregated into an additive approximation term, which we treat separately from the calibration-driven degradation. This setup isolates the conceptual role of certification: the only unavoidable statistical price we aim to pay, beyond approximation effects, is controlled by ε and amplified by the effective horizon $1/(1-\gamma)$.

5 Oracle analysis: a clean core for certified state-constrained planning

In this section we analyze the planning problem induced by a *hypothetical oracle* reachability certification procedure. The purpose is twofold. First, we isolate the algorithmic and dynamic-programming structure of the certified backup from any particular learning mechanism. Second, we make explicit which parts of the final guarantee are information-theoretic (driven

by false positives) as opposed to approximation-theoretic (driven by imperfect value/reward/controller models, deferred to the next section).

Oracle reachability and the nature of the statistical error. We assume that for every ordered pair $(s, s') \in \mathcal{S}_D^2$ we are given a lower confidence bound $\underline{p}(s \to s')$ satisfying the distributional false-positive control (1) with respect to a reference distribution μ on $\mathcal{S}_D^2$, and the nondegeneracy condition (2). We emphasize that (1) is *not* a uniform guarantee over all pairs; it only bounds the frequency (under μ) with which an edge is erroneously certified as more reliable than it is. Thus, for any fixed dataset, there may exist "bad" pairs (s, s') with $\underline{p}(s \to s') \geq \delta$ even though $p(s \to s') < \delta$; the only protection is that such pairs are rare under μ.

To focus on the core planning phenomenon, we temporarily treat $\hat{r}(s, s')$ as an arbitrary but fixed bounded function on $\mathcal{S}_D^2$ (to be instantiated either by the true expected reward under g or by a learned estimator). The sole structural assumption is the existence of a known lower bound $V_{\min} \leq -r_{\max}/(1-\gamma)$.

Contraction, monotonicity, and existence of a unique fixed point. We begin with the operator-theoretic properties of the pessimistic certified backup (3). These properties ensure that planning over targets is well-posed even though we only ever optimize over a *set-valued* reachability predicate.

Proposition 1 (monotonicity). For any bounded $V, W : \mathcal{S}_D \to \mathbb{R}$ with $V(s) \leq W(s)$ for all $s \in \mathcal{S}_D$, we have $(\widehat{T}_\delta V)(s) \leq (\widehat{T}_\delta W)(s)$ for all $s \in \mathcal{S}_D$.

Proof. Fix s. For each $s' \in \widehat{\mathrm{Reach}}_\delta(s)$, the quantity

$$\hat{r}(s, s') + \gamma\big(\delta V(s') + (1-\delta)V_{\min}\big)$$

is nondecreasing in $V(s')$ since $\gamma\delta \geq 0$. Taking maxima over the same feasible set preserves the inequality. □

Proposition 2 (contraction). The operator $\widehat{T}_\delta$ is a $\gamma\delta$-contraction in $\|\cdot\|_\infty$ over bounded functions on $\mathcal{S}_D$.

Proof. Fix V, W and s. For any $s' \in \widehat{\mathrm{Reach}}_\delta(s)$ we have

$$\Big|\gamma\big(\delta V(s')+(1-\delta)V_{\min}\big)-\gamma\big(\delta W(s')+(1-\delta)V_{\min}\big)\Big| = \gamma\delta\,|V(s')-W(s')| \leq \gamma\delta\,\|V-W\|_\infty.$$

The map $x \mapsto \max_{s'} x_{s'}$ is nonexpansive in the sup-norm, hence

$$\|\widehat{T}_\delta V - \widehat{T}_\delta W\|_\infty \leq \gamma\delta\,\|V - W\|_\infty.$$

□

By Banach's fixed point theorem, there exists a unique fixed point $\hat{V}_\delta$ and value iteration $V_{k+1} = \widehat{T}_\delta V_k$ converges to $\hat{V}_\delta$ at rate $(\gamma\delta)^k$. This provides

a clean separation between (i) the computational question of approximately maximizing over $\widehat{\mathrm{Reach}}_\delta(s)$ and (ii) the statistical question of how often $\widehat{\mathrm{Reach}}_\delta(s)$ contains a genuinely unsafe target.

Oracle dominance: why certified state constraints can subsume batch constraints. We next formalize a dominance claim that motivates planning in the space of dataset states rather than dataset actions. The key point is that feasibility is expressed as a constraint on *next-state support*, which can be strictly weaker than constraining actions to those observed in D.

Let π_{BC} be any baseline policy whose one-step transitions remain in $\mathcal{S}_D$ almost surely. Suppose further that the baseline can be realized through the controller interface in the sense that for each $s \in \mathcal{S}_D$ there exists a (possibly randomized) selection over targets s' such that the induced action equals $\pi_{\mathrm{BC}}(s)$ and the next state lies in $\mathcal{S}_D$ with probability one. In this case, the baseline is feasible for any oracle reachability set that contains all baseline-induced next states.

Proposition 3 (policy-set inclusion implies dominance). Assume there exists an "oracle" feasible set $\mathrm{Reach}^*_\delta(s) \subseteq \mathcal{S}_D$ such that (i) for all s the baseline selects targets only in $\mathrm{Reach}^*_\delta(s)$, and (ii) for all $s' \in \mathrm{Reach}^*_\delta(s)$ we have $p(s \to s') \geq \delta$. Let V^*_δ denote the optimal value over policies constrained to select targets from $\mathrm{Reach}^*_\delta(\cdot)$. Then $V^*_\delta(s) \geq V^{\pi_{\mathrm{BC}}}(s)$ for all $s \in \mathcal{S}_D$.

Proof. Under (i), π_{BC} is a member of the oracle constrained class; therefore the supremum of returns over that class is at least the return of π_{BC}. □

This proposition is deliberately elementary: it makes no mention of calibration. Its purpose is only to justify the modeling choice that *state constraints* are a natural relaxation of strict batch constraints. The remaining question is how much this advantage deteriorates when we replace Reach^*_δ by the certified set $\widehat{\mathrm{Reach}}_\delta$ that may contain false positives.

Robustness to false positives: the role of pessimism. We now explain how the pessimistic backup converts a bound on false-positive frequency into an additive value loss. At a high level, the max over $\widehat{\mathrm{Reach}}_\delta(s)$ creates the possibility of selecting a falsely certified edge, but the backup assigns only δ mass to the purported target and sends the remaining $1-\delta$ mass to $V_{\min}$. Consequently, even if the edge is entirely spurious (e.g. it transitions outside $\mathcal{S}_D$ with high probability), the algorithm does not implicitly assume a continuation value larger than $V_{\min}$ on the failure event.

To make the dependence explicit, we reason as follows. Consider the idealized case where rewards are bounded by $r_{\max}$ and where, whenever a false positive occurs, the realized continuation value can be as low as $V_{\min}$ rather

than the value of the intended target. The maximal one-step harm of such a deviation is on the order of $\gamma\delta\,(V_{\max} - V_{\min})$, which is $O(\gamma\delta\, r_{\max}/(1-\gamma))$ under bounded rewards. Over an infinite horizon this accumulates geometrically, yielding an effective amplification of order $1/(1-\gamma\delta) \le 1/(1-\gamma)$. The additional factor $1/\delta$ appearing in the final comparison bounds reflects that, to be competitive with a baseline that is *sure* to remain in $\mathcal{S}_D$, our method must tolerate that its own certified edges are only guaranteed at level δ; thus a fixed rate of false positives consumes a larger fraction of the already limited "success budget".

While the precise constants depend on how μ relates to the state–target pairs actually selected by the policy, the qualitative message is invariant: calibration error enters linearly in ε and is amplified by the effective horizon $1/(1-\gamma)$ and by the conservatism parameter $1/\delta$.

A matching lower bound: false positives are information-theoretically costly. Finally, we show that an $O(\varepsilon/(1-\gamma))$-type degradation is unavoidable in general. The construction is standard in spirit: we create a safe branch that is well supported by the dataset and a trap branch that is entered only through a small set of edges that are indistinguishable from safe edges to any procedure that makes false-positive errors with probability at least ε.

Concretely, consider an MDP with an initial dataset state $s_0 \in \mathcal{S}_D$ and two candidate targets $s_{\text{safe}}, s_{\text{trap}} \in \mathcal{S}_D$ (or with $s_{\text{trap}} \notin \mathcal{S}_D$, depending on whether one prefers the trap to be explicitly out-of-support). The controller g admits actions corresponding to "attempt safe" and "attempt trap". Under the true dynamics, attempting safe reaches s_{safe} with probability one and yields reward 0 forever thereafter. Attempting trap reaches a terminal absorbing state with reward $-r_{\max}$ forever (equivalently, transitions to a region outside $\mathcal{S}_D$ where the value equals $V_{\min} = -r_{\max}/(1-\gamma)$). The dataset D is chosen so that both attempts appear equally plausible from s_0 to any learner except through the reachability certification signal. If the certification procedure produces a false positive on the trap edge with probability at least ε under μ, then with probability at least ε the planner may admit the trap edge as δ-feasible and, due to maximization, select it (for instance if we perturb the immediate rewards so that the trap edge appears slightly better under $\hat{r}$). On this event, the realized value suffers a loss of approximately $r_{\max}/(1-\gamma)$ relative to the safe baseline. Therefore the expected degradation is at least on the order of $r_{\max}\varepsilon/(1-\gamma)$.

This lower bound is not an artifact of our particular algorithm; it is a statement about the decision problem defined by one-step certificates with nonzero false-positive rate. Any method that sometimes declares an unsafe edge safe must, on some instance, pay a linear price in that error probability times the horizon. The role of the pessimistic backup is thus not to eliminate

this dependence, but to ensure that it is the *only* unavoidable statistical dependence once approximation errors are controlled.

From oracle to learned reachability. The preceding discussion treats the map $(s, s') \mapsto \underline{p}(s \to s')$ as an oracle object satisfying the calibration condition (C1). We now explain how such certificates can be constructed from an offline dataset using standard predictive models together with a distribution-free calibration step. The central point is that the planning analysis only requires *one-sided* validity of the lower bounds under a reference distribution μ; thus we may modularize the design into (i) a statistical model that produces scores correlated with success and (ii) a calibration procedure that converts scores into LCBs with controlled false-positive rate.

Two modeling routes: inverse–forward versus direct success prediction. We require a controller interface $g(s, s')$ and a success probability $p(s \to s') = \Pr[s_{t+1} = s' \mid s_t = s,\, a_t = g(s, s')]$ (or an appropriate relaxation when states are continuous). There are two natural constructions.

(A) Inverse dynamics plus forward model. We may fit an inverse model g_θ by supervised learning on dataset transitions $(s_t, a_t, s_{t+1}) \in D$, e.g.

$$\theta \in \arg\min_\theta \mathbb{E}_{(s,a,s')\sim D}\big[\ell(g_\theta(s, s'), a)\big],$$

with ℓ a regression or classification loss depending on whether $\mathcal{A}$ is continuous or discrete. Separately, we fit a forward model $\widehat{P}_\phi(\cdot \mid s, a)$, for example by maximizing log-likelihood. For any candidate target $s' \in \mathcal{S}_D$ we then define a *model-based* success estimate

$$\hat{p}(s \to s') := \widehat{P}_\phi(s' \mid s, g_\theta(s, s')).$$

This route makes explicit the causal structure: the only place where the target s' enters is through the controller action $g_\theta(s, s')$.

(B) Direct target-reaching prediction. Alternatively, we may fit a single predictor $\hat{p}_\phi(s, s')$ of the success event under the controller g_θ, viewing (s, s') as the input and $Y = \mathbf{1}\{s_{t+1} = s'\}$ as the label. Since the dataset contains only the realized next state, we typically form training pairs by *relabeling*: for each transition (s_t, a_t, s_{t+1}), we treat (s_t, s_{t+1}) as a positive pair and sample negatives $(s_t, \tilde{s})$ with $\tilde{s} \neq s_{t+1}$ from an auxiliary distribution on $\mathcal{S}_D$. The resulting binary classification problem yields a calibrated score of reachability that can be evaluated on arbitrary candidate targets. This route avoids explicit forward modeling but pushes the burden into the pairwise supervision scheme.

In either case, we emphasize that the planning layer only consumes a number in $[0, 1]$ for each ordered pair (s, s') together with a valid LCB; the internal decomposition is immaterial for the subsequent operator analysis.

Uncertainty quantification via ensembles. Offline datasets may be sparse in the relevant (s, s') pairs, and generalization errors are precisely what induce false positives. As a practical device, we may train an ensemble $\{\hat{p}^{(m)}\}_{m=1}^{M}$ (by bootstrapping the dataset, varying initializations, or using approximate Bayesian methods). The ensemble dispersion provides a nonconformity score for calibration and, operationally, allows us to prioritize targets that are both promising and certain. Concretely, we may compute a point prediction $\bar{p}(s \to s') := \frac{1}{M}\sum_{m=1}^{M} \hat{p}^{(m)}(s \to s')$ and a dispersion proxy $\sigma^2(s \to s') := \frac{1}{M}\sum_{m=1}^{M}(\hat{p}^{(m)} - \bar{p})^2$, and then pass the pair $(\bar{p}, \sigma)$ to the calibration layer. We stress that ensemble variance by itself is not a guarantee; it only serves as an input to a finite-sample calibration step that provides (C1).

Conformal calibration and valid LCBs. We now describe a split-conformal construction that turns any scoring rule into a lower confidence bound with distribution-free validity under μ. Let $\mathcal{C} = \{(x_i, y_i)\}_{i=1}^{n}$ be a calibration set of i.i.d. samples from μ, where $x_i = (s_i, s_i')$ and $y_i \in \{0, 1\}$ indicates whether an execution of $g(s_i, s_i')$ reaches s_i' in one step. (In practice, we obtain such samples either from held-out transitions when the action in the dataset is close to $g(s, s')$, or from a learned success proxy; the analysis only uses exchangeability of the calibration pairs with future test pairs under μ.) Let $\hat{p}(x) \in [0, 1]$ be any predictor trained on a disjoint training split. Define a nonconformity score $\alpha(x, y)$ that is *large* when the predictor is overly optimistic. One convenient choice is

$$\alpha(x, y) := \hat{p}(x) - y,$$

which is positive exactly when $\hat{p}$ exceeds the realized success label. Compute calibration scores $\alpha_i = \alpha(x_i, y_i)$ and let $q_{1-\varepsilon}$ be the $(1-\varepsilon)$ empirical quantile of $\{\alpha_i\}_{i=1}^{n}$ with the usual conformal $+1$ correction. We then set

$$\underline{p}(x) := \big[\hat{p}(x) - q_{1-\varepsilon}\big]_+, \qquad [z]_+ := \max\{z, 0\}.$$

Under exchangeability, split conformal implies

$$\Pr_{(x,y)\sim\mu}\big[\underline{p}(x) \le y\big] \ge 1 - \varepsilon.$$

Interpreting y as a Bernoulli draw with mean $p(x)$, the above is a one-sided control on overly optimistic predictions at the *event* level; standard arguments (as summarized by Theorem 5) upgrade this to a bound of the form

$$\Pr_{x\sim\mu}\big[\underline{p}(x) \le p(x)\big] \ge 1 - \varepsilon,$$

namely exactly (C1). The precise score may be modified to incorporate ensemble dispersion, for instance by taking $\alpha(x, y) := (\hat{p}(x) + \beta\sigma(x)) - y$ for

a tuning parameter $\beta \geq 0$, which trades sharpness for fewer false positives prior to conformalization; the conformal guarantee remains distribution-free as long as the score is computed without using the calibration labels beyond the quantile.

On the role of the reference distribution μ. Condition (C1) is inherently *distributional*: it controls false positives under μ, not uniformly over $\mathcal{S}_D^2$. Consequently, the operational choice of μ is part of the algorithm design. In the present setting, μ should reflect the pairs that the planner will actually query, i.e. the joint distribution of a dataset state s encountered during planning/training and a target s' proposed by the retrieval mechanism. If μ is chosen in this manner, then (C1) bounds the probability that the planner ever treats an unsafe target as δ-feasible when it is sampled from its own query distribution.

When μ differs from the induced query distribution ν, one may still proceed provided a mild absolute-continuity condition holds, e.g. $\nu(x) \leq C\,\mu(x)$ for all relevant pairs $x = (s, s')$. In that case, the effective false-positive probability under ν is at most $C\varepsilon$, and all subsequent performance bounds hold with ε replaced by $C\varepsilon$. This is the familiar "concentrability" phenomenon in offline RL, appearing here at the level of *edge certification* rather than action-value estimation.

Translating calibration error into policy suboptimality. We finally connect the statistical guarantee (C1) to the planning objective. The core mechanism is already visible in the operator $\widehat{T}_\delta$: the backup explicitly assigns only δ mass to the target value and pessimistically assigns $(1-\delta)$ mass to $V_{\min}$. This design ensures that if a certified edge (s, s') is a false positive (so that $p(s \to s') < \delta$), then the Bellman target is not catastrophically optimistic about the failure event.

To make this precise, suppose that the only source of statistical error is certification, and that rewards are bounded by $r_{\max}$. Consider an update at state s that selects a target $\tilde{s} \in \widehat{\mathrm{Reach}}_\delta(s)$. If the edge is truly δ-feasible, then the continuation value is at least $\delta V(\tilde{s}) + (1-\delta)V_{\min}$ by definition of $V_{\min}$. If the edge is a false positive, we may still lower bound the realized continuation value by $V_{\min}$, hence the *excess optimism* of treating the edge as δ-feasible is at most

$$\gamma\delta\big(V(\tilde{s}) - V_{\min}\big) \leq \gamma\delta\Big(\frac{r_{\max}}{1-\gamma} - V_{\min}\Big) \leq \frac{2\gamma\delta\, r_{\max}}{1-\gamma},$$

where we used $V_{\min} \leq -r_{\max}/(1-\gamma)$ and $V(\cdot) \leq r_{\max}/(1-\gamma)$ for bounded rewards. Thus each false positive can cost at most a constant multiple of $r_{\max}/(1-\gamma)$ in discounted value, and by (C1) such events occur with probability at most ε under μ (or $C\varepsilon$ under a shift factor C as above). Iterating

through the contraction of $\widehat{T}_\delta$ converts these one-step losses into an infinite-horizon bound, yielding an overall degradation linear in ε and amplified by the effective horizon. This is the quantitative content behind the comparison statement of Theorem 3: calibration error enters additively as $O\Big(\frac{r_{\max}}{1-\gamma} \cdot \frac{\varepsilon}{\delta}\Big)$ once we compare to a baseline that never leaves $\mathcal{S}_D$.

Where approximation errors enter. In the learned setting, we additionally incur approximation error from (i) the controller g_θ (which may fail to realize the intended transition), (ii) the reward estimator $\hat{r}$, and (iii) the value function approximation used in fitted iteration. These errors appear through the usual approximate dynamic programming inequalities: if $\widehat{T}_\delta$ is implemented approximately by an operator $\widetilde{T}_\delta$ with a uniform Bellman error bound $\|\widetilde{T}_\delta V - \widehat{T}_\delta V\|_\infty \le \eta$, then the fixed point $\tilde{V}$ of $\widetilde{T}_\delta$ satisfies

$$\|\tilde{V} - \hat{V}_\delta\|_\infty \le \frac{\eta}{1-\gamma\delta},$$

and the resulting greedy policy suffers an additional $O(\eta/(1-\gamma))$ loss. In our final guarantees we collect these terms into ApproxErr, thereby isolating the statistical contribution of false-positive certification from the approximation-theoretic contribution of imperfect models.

6 Algorithms: Certified StaCQ (C-StaCQ) for stochastic settings

We now spell out an implementation of *Certified StaCQ* (C-StaCQ) tailored to stochastic environments, emphasizing the components that are typically left implicit in the operator-level analysis: (i) how we parameterize the target-conditioned controller interface, (ii) how we generate and maintain candidate targets efficiently, and (iii) how we incorporate reward estimation and additional conservatism (beyond the hard certification constraint) to stabilize training.

Target-conditioned control as a one-step (or short-horizon) option. In the tabular exposition, a decision at $s \in \mathcal{S}_D$ consists of selecting a target $s' \in \mathcal{S}_D$ and executing the action $a = g(s, s')$. In stochastic systems, it is often preferable to interpret g as a *short-horizon* controller (an "option") intended to reach s' within H environment steps, rather than in exactly one step. Concretely, we may define a termination event $E_{s\to s'} := \{\exists t \le H : s_t \in B(s')\}$ for a neighborhood $B(s')$ (e.g. an ℓ_2-ball in state space or a discretization cell), and define the success probability

$$p_H(s \to s') := \Pr(E_{s\to s'} \mid s_0 = s,\ \text{execute } g(\cdot, s') \text{ for } H \text{ steps}).$$

The same certification and planning logic applies provided we (i) replace p by p_H, (ii) use an effective discount γ^H in the backup, and (iii) define the reward estimate as the expected discounted return accumulated during the option execution. For notational simplicity we continue to write $p(s \to s')$ and $\hat{r}(s, s')$, with the understanding that these may already aggregate over an internal horizon H and a relaxed success set $B(s')$.

Certified target selection and target-mixture policies. Given a certified set $\widehat{\mathrm{Reach}}_\delta(s)$, the simplest policy is deterministic:

$$\hat{s}(s) \in \arg\max_{s' \in \widehat{\mathrm{Reach}}_\delta(s)} \Big(\hat{r}(s, s') + \gamma(\delta \hat{V}_\delta(s') + (1-\delta) V_{\min})\Big), \qquad \hat{\pi}_\delta(\cdot \mid s) = \delta_{g(s, \hat{s}(s))}.$$

In continuous control, a stochastic mixture over targets is frequently more stable, especially early in training when $\hat{V}$ is inaccurate. A convenient parameterization is a Gibbs distribution over *certified* targets:

$$\rho_\tau(s' \mid s) \;\propto\; \mathbf{1}\{s' \in \widehat{\mathrm{Reach}}_\delta(s)\} \exp\Big(\tfrac{1}{\tau} U(s, s')\Big), \qquad U(s, s') := \hat{r}(s, s') + \gamma(\delta \hat{V}(s') + (1-\delta) V_{\min}),$$

followed by sampling $s' \sim \rho_\tau(\cdot \mid s)$ and executing $g(s, s')$. The hard constraint (support restricted to $\widehat{\mathrm{Reach}}_\delta(s)$) is preserved, while $\tau > 0$ interpolates between exploration over certified targets and near-greediness.

Candidate generation and retrieval for reachability queries. The planning layer does *not* enumerate $\mathcal{S}_D$; it only evaluates a small candidate list $\mathcal{K}(s) \subset \mathcal{S}_D$ and then certifies within that list. Thus, a critical design choice is the retrieval mechanism $s \mapsto \mathcal{K}(s)$.

We have found the following sources of candidates to be complementary.

1. *Empirical successors.* If D contains transitions out of s, we include the observed next states (or their discretized representatives). This gives high-quality "local" moves and improves coverage.

2. *Nearest neighbors in a representation.* We maintain an embedding $z = h_\psi(s)$ (either learned jointly with the reachability model or taken from a pretrained encoder), build an approximate nearest-neighbor index over $\{z(s) : s \in \mathcal{S}_D\}$, and query K neighbors of $z(s)$. This yields candidates that are geometrically plausible even when s has few outgoing samples in D.

3. *Goal-biased candidates.* If a separate value model suggests that certain regions of $\mathcal{S}_D$ have high value, we additionally retrieve neighbors of a small set of "elite" states under $\hat{V}$. This encourages longer-range planning through chained certified moves.

We then set $\mathcal{K}(s)$ as the union of these sources (capped at a fixed budget), compute $\underline{p}(s \to s')$ for each $s' \in \mathcal{K}(s)$, and define $\widehat{\mathrm{Reach}}_\delta(s) = \{s' \in \mathcal{K}(s) : \underline{p}(s \to s') \geq \delta\}$. In practice, we also enforce (C2) by adding a fallback target such as $s' = s$ (a "do nothing" or "stay" controller) and certifying it separately; if it fails certification, we enlarge $\mathcal{K}(s)$ adaptively until at least one certified target is found.

Implementing the certification model efficiently. The certification call $(s, s') \mapsto \underline{p}(s \to s')$ must be cheap, since it is invoked $O(K)$ times per visited state. Two implementation patterns are common.

- *Direct pair model.* A network takes $(h(s), h(s'))$ (or their concatenation/difference) and outputs $\hat{p}(s, s')$, optionally with an ensemble to provide dispersion features. This is naturally compatible with negative sampling.
- *Controller plus forward model.* If g outputs an action, and a forward model predicts a distribution over next-state embeddings, then $\hat{p}$ is computed by evaluating the likelihood of the target under that distribution (or by Monte Carlo). This is more expensive but can generalize well when the dynamics structure is learnable.

In either case, we clip $\underline{p}$ to $[0, 1]$ after conformalization, and we recommend caching $\underline{p}(s \to s')$ for frequently queried pairs (e.g. within a minibatch) to avoid repeated forward passes.

Reward modeling and consistency with the target abstraction. C-StaCQ requires a reward estimate $\hat{r}(s, s')$ aligned with the controller abstraction. If rewards are Markovian and depend on (s, a, s^+), but the policy chooses s' and executes $a = g(s, s')$, we may define

$$\hat{r}(s, s') \approx \mathbb{E}\big[r(s, g(s, s'), s^+)\big],$$

estimated by (i) a learned reward model $\hat{r}_\omega(s, a, s^+)$ combined with a learned forward model for s^+, or (ii) direct regression $\hat{r}_\omega(s, s')$ using the same relabeling scheme as for $\hat{p}$. When rewards are observed in D and the controller g is close to the behavior actions, a simple and effective baseline is to set $\hat{r}(s, s')$ as the empirical average reward among transitions whose (s, s^+) pair matches (s, s') up to discretization.

Since the analysis assumes bounded rewards, we explicitly clip reward predictions: $\hat{r}(s, s') \leftarrow \mathrm{clip}(\hat{r}(s, s'), -r_{\max}, r_{\max})$. This prevents rare modeling artifacts from dominating the max over targets.

Additional conservatism: using $\underline{p}$ inside the backup. The theoretical operator uses a fixed δ in the continuation term to separate *feasibility* (hard constraint) from *pessimism* (fixed mass split). In implementations, we can increase robustness by also using the pair-dependent LCB in the backup, replacing δ by $\underline{p}(s \to s')$:

$$(\widehat{T}V)(s) \;:=\; \max_{s' \in \widehat{\mathrm{Reach}}_\delta(s)} \Big(\hat{r}(s,s') + \gamma\big(\underline{p}(s \to s')V(s') + (1 - \underline{p}(s \to s'))V_{\min}\big)\Big).$$

This modification remains pessimistic whenever $\underline{p} \le p$ (the intended calibration outcome), and it encourages selecting targets that are not only above threshold but also *well* above it. Empirically, this reduces sensitivity to the choice of δ and mitigates brittleness when the certified set is large.

Value clipping and numerical stability. We finally note two stability measures that are innocuous in the tabular theory but important with function approximation. First, we clip learned value estimates to known bounds, e.g.

$$V(s) \leftarrow \mathrm{clip}\left(V(s),\, V_{\min},\, \frac{r_{\max}}{1-\gamma}\right),$$

ensuring that the pessimistic term $(1-\delta)V_{\min}$ remains meaningful. Second, we recommend tie-breaking among targets by preferring larger $\underline{p}(s \to s')$ (or smaller ensemble dispersion) when their backed-up utilities are nearly equal; this reduces variance in target choice without changing the certified feasibility constraint.

These implementation choices instantiate the abstract operator analysis in a form that is computationally viable in large $\mathcal{S}_D$ and robust to stochasticity, thereby setting up the empirical study in the next section.

7 Experiments: stochastic benchmarks, calibration–performance link, and ablations

We evaluate C-StaCQ on stochastic offline-control benchmarks with the goal of isolating the role of certified reachability, the confidence threshold δ, and the calibration error rate ε in the resulting policy performance. The experimental questions we target are: (Q1) whether certification improves robustness relative to standard batch-constrained and pessimistic baselines under increased stochasticity; (Q2) how sensitive performance is to δ; (Q3) whether empirical calibration curves predict performance degradation in the manner suggested by Theorem 3; and (Q4) whether the method fails primarily due to false positives (unsafe edges) or false negatives (overly conservative pruning).

Benchmarks: stochastic D4RL variants. Standard D4RL MuJoCo tasks are only mildly stochastic (beyond observation noise and finite-sample effects), whereas our analysis is motivated by explicit reachability uncertainty. We therefore consider stochastic variants of D4RL datasets constructed by modifying the transition kernel at evaluation time while keeping the offline dataset fixed. Concretely, for each environment we introduce a family of perturbed evaluation dynamics indexed by $\eta \in [0, 1]$:

$$P_\eta(\cdot \mid s, a) \;:=\; (1-\eta)P_0(\cdot \mid s, a) \;+\; \eta\, \widetilde{P}(\cdot \mid s),$$

where P_0 is the original environment dynamics and $\widetilde{P}(\cdot \mid s)$ is a state-dependent "stochastic reset" distribution implemented by injecting Gaussian noise in velocities (for MuJoCo) and/or randomly replacing the next state by a nearest-neighbor state in $\mathcal{S}_D$ (to preserve state plausibility while breaking determinism). We report results on (i) the unperturbed case $\eta = 0$, (ii) moderate stochasticity $\eta = 0.1$, and (iii) heavy stochasticity $\eta \in \{0.2, 0.3\}$, where long-horizon compounding effects become visible. This protocol keeps the offline training distribution fixed and induces controlled distribution shift at test time.

Algorithms compared. We compare against baselines intended to span the common offline RL design space: behavior cloning (BC), a batch-constrained algorithm (BCQ or a nearest-neighbor action constraint), and pessimistic value-based methods (CQL and IQL). We also include a pure "reachability-max" ablation that uses $\widehat{\mathrm{Reach}}_\delta(s)$ but replaces the pessimistic mass split by the optimistic continuation $\gamma V(s')$; this ablation directly tests the necessity of pessimism under certification error. For C-StaCQ we evaluate both the fixed-δ operator described in the theory and the implementation variant that uses the pair-dependent LCB $\underline{p}(s \to s')$ inside the backup. All methods are trained on identical datasets and evaluated in the same stochastic evaluation environments P_η.

Implementation details: controller, retrieval, and calibration. We instantiate $g(s, s')$ as a target-conditioned inverse dynamics model trained by supervised regression on dataset transitions. For continuous actions we use a Gaussian policy with mean $\mu_\theta(s, s')$ and fixed or learned diagonal covariance; $g(s, s')$ is taken as the mean action at evaluation time. Candidate targets are obtained via approximate nearest neighbors in an embedding space $z = h_\psi(s)$ learned jointly with the reachability predictor, together with empirical successors as described in the previous section. We set a fixed retrieval budget K (e.g. $K \in \{50, 100, 200\}$) and measure how often coverage (C2) fails before fallback or adaptive expansion. To obtain $\underline{p}(s \to s')$ we train an ensemble predictor of success events (either one-step or option-level) and apply split conformal calibration on a held-out calibration set drawn from the

reference distribution μ induced by the retrieval procedure (i.e. pairs (s, s') that are actually queried during planning). This choice aligns the finite-sample guarantee in Theorem 5 with the distribution on which certification is used.

Evaluation metrics. We report standard normalized D4RL return, but also explicitly measure safety-relevant quantities tied to our analysis. First, we estimate the realized one-step (or option-level) success rate of chosen targets:

$$\widehat{\mathrm{Succ}}(\pi) \;:=\; \mathbb{E}\big[\mathbf{1}\{s_{t+1} \in B(s'_t)\}\big], \qquad s'_t \sim \text{target distribution of } \pi(\cdot \mid s_t),$$

with $B(s')$ the success neighborhood used in training. Second, we compute an empirical false-positive rate under μ,

$$\widehat{\varepsilon} \;:=\; \Pr_{(s,s')\sim\mu}\big[\underline{p}(s \to s') > \widehat{p}_{\mathrm{MC}}(s \to s')\big],$$

where $\widehat{p}_{\mathrm{MC}}$ is a Monte Carlo estimate of the true success probability using repeated rollouts of $g(s, s')$ from state s in the evaluation environment. Third, we report the certified set size $\big|\widehat{\mathrm{Reach}}_\delta(s)\big|$ averaged over visited states, as a proxy for conservatism and planning flexibility.

Sensitivity to the confidence threshold δ. We sweep δ over a log-spaced grid (e.g. $\{0.5, 0.6, 0.7, 0.8, 0.9, 0.95\}$) and evaluate returns and realized success rates across η. We expect a characteristic bias–variance tradeoff: small δ enlarges feasible sets but increases the magnitude of the pessimistic penalty $(1-\delta)V_{\min}$, while large δ restricts planning and can induce coverage failures when the dataset is sparse. We therefore plot (i) return versus δ and (ii) coverage failure frequency versus δ, and we additionally report the effective contraction modulus $\gamma\delta$ as a diagnostic of optimization stability in fitted value iteration. In practice we find it informative to compare fixed-δ backups against the $\underline{p}$-weighted backup: the latter often reduces sensitivity by smoothly interpolating between near-threshold and well-supported targets.

Calibration curves and performance prediction. To connect empirical behavior to the theory, we produce calibration plots that relate LCB conservatism to realized success. Specifically, for a set of queried pairs (s, s') we bucket them by $\underline{p}(s \to s')$ and plot the empirical success frequency within each bucket; we additionally plot the coverage curve $\Pr[p(s \to s') \geq x \mid \underline{p}(s \to s') \geq x]$ as a function of x. We then correlate these calibration statistics with downstream return across δ and η. The hypothesis suggested by Theorem 3 is that, holding other errors fixed, performance degradation tracks the mass of false positives relative to the chosen confidence level, approximately through a factor of ε/δ. Empirically, we therefore report scatter

plots of return gap to a batch-constrained baseline versus the measured ratio $\widehat{\varepsilon}/\delta$, and we test whether the relationship is monotone across tasks and stochasticity levels.

Ablations separating false positives from false negatives. We isolate the effect of (i) erroneously including unsafe targets (false positives) and (ii) excluding safe targets (false negatives). Since ground-truth $p(s \to s')$ is not directly available, we approximate it via Monte Carlo rollouts for a subset of queried pairs and define an empirical "safe" edge as one with $\widehat{p}_{\mathrm{MC}}(s \to s') \geq \delta$. We then construct two diagnostic variants: (A) an FP-inflated variant that artificially increases $\underline{p}$ by a constant shift before thresholding, raising false positives at fixed δ; and (B) an FN-inflated variant that decreases $\underline{p}$, increasing conservatism. We expect (A) to harm return sharply under stochastic evaluation (large η), especially for the optimistic "reachability-max" ablation, while (B) should primarily reduce return via limited connectivity and smaller $\widehat{\mathrm{Reach}}_\delta(s)$, but with comparatively stable realized success. This experiment clarifies whether the empirical failure mode is brittleness to occasional unsafe edges (the setting of Theorem 4) or excessive conservatism from mis-specified μ or underpowered models.

Comparisons to batch-constrained and pessimistic baselines under stochasticity. Finally, we compare C-StaCQ against BCQ-like constraints and against pessimistic Q-learning methods across η. The central claim we test is that explicit certified state transitions provide a complementary robustness mechanism: batch action constraints can still permit transitions to out-of-distribution states under perturbed dynamics, while purely value-based pessimism can be overly diffuse without an explicit notion of feasible next-state support. We therefore report not only normalized return but also (i) visitation concentration in $\mathcal{S}_D$ measured by nearest-neighbor distance to dataset states, and (ii) the frequency of entering low-support regions (large distance), which should increase with η for methods lacking an explicit state-constraint mechanism. We emphasize that these measurements are descriptive rather than guaranteed in function approximation; their purpose is to substantiate the qualitative interpretation of certified reachability as a tool for controlling distribution shift.

Together, these experiments aim to make the operator-level guarantees operational: we measure the calibration properties that instantiate (C1), verify coverage behavior related to (C2), and quantify how false-positive reachability errors translate into return degradation under controlled stochasticity.

8 Discussion and limitations

Our development isolates a particular mechanism for robustness in offline control: we constrain one-step transitions by certifying the feasibility of target states inside the dataset support. This yields an operator whose contraction modulus is explicitly improved from γ to $\gamma\delta$ and whose performance degradation can be related to the false-positive rate ε through a factor resembling ε/δ. The same isolation also makes clear what our approach does *not* address; we summarize the principal limitations and the mathematical obstacles that remain.

Coverage and connectivity assumptions. The coverage condition (C2) is structurally necessary for any method that enforces feasibility through a hard constraint $s' \in \widehat{\mathrm{Reach}}_\delta(s)$. When (C2) fails for some $s \in \mathcal{S}_D$, the operator $\widehat{T}_\delta$ is not well-defined unless we supply a fallback action (or equivalently we add an absorbing failure state with value $V_{\min}$). In practice, (C2) is best interpreted as a statement about the *connectivity* of the induced certified graph on $\mathcal{S}_D$: if for many states the certified out-degree is near zero, planning degenerates into a short-horizon policy that repeatedly invokes the fallback. This phenomenon is not captured by ε alone; false negatives (overly conservative $\underline{p}$) can collapse the reachable set while still satisfying calibration.

There is also a subtler limitation: even when each state has at least one certified successor, the *global* geometry of $\widehat{\mathrm{Reach}}_\delta$ may prevent reaching high-value regions because the certified edges do not compose. One-step feasibility does not guarantee multi-step feasibility under distribution shift, and our pessimistic backup only addresses failure at the immediate step. A natural refinement is to couple (C2) with a mixing or expansion condition on the certified graph, e.g. that for each s there exists a path of certified edges to a set of "well-supported" states. Formalizing such a condition in continuous control (where $\mathcal{S}_D$ is large and irregular) remains open.

The role and fragility of the reference distribution μ. Calibration (C1) is stated under a fixed reference distribution μ on pairs (s, s'). This is mathematically clean but operationally delicate: if the planning procedure adaptively changes which pairs are queried, then the empirical distribution of queried pairs may drift away from the calibration distribution. In the present work we partially mitigate this by defining μ to be induced by the retrieval mechanism, hence aligning certification with use. Nevertheless, there is no guarantee that the state-visitation distribution under $\hat{\pi}_\delta$ will match the distribution of states used for calibration, especially under heavy stochasticity. This mismatch is a standard instance of the "offline-to-online shift" for uncertainty quantification: conformal validity is distribution-free *given*

exchangeability, but it is not automatically robust to adaptive querying when the set of queried points is itself a function of the learned model.

A principled direction is to treat certification as an online prediction problem and use sequential conformal methods or martingale-based concentration to control miscoverage under adaptive selection. Another direction is to incorporate explicit importance weighting to target the induced distribution of queried pairs. Either approach requires additional assumptions (bounded likelihood ratios or stable selection), and the extent to which such assumptions hold in high-dimensional continuous control is unclear.

Partial observability and representation dependence. Our analysis is formulated in terms of a Markov state $s \in \mathcal{S}$, whereas many offline datasets are effectively partially observable: logged observations o_t may omit latent variables that influence dynamics, and the learned controller $g(o, o')$ may exploit spurious correlations. In this setting, the event "reach o' from o" is not a well-defined property of the environment; it depends on the unobserved latent state and on the history. One can attempt to lift the state to a history window $h_t = (o_{t-k+1}, \ldots, o_t)$ or to a learned belief embedding z_t, and to run the same machinery on z. However, certification in representation space introduces new failure modes: an LCB $\underline{p}(z \to z')$ may be well-calibrated for the learned predictor yet correspond poorly to the true probability of reaching an observation-neighborhood in the environment.

From a theoretical perspective, the correct object is a belief-MDP (or a predictive state representation) under which the Markov property holds. A complete extension would require (i) a definition of success events that is invariant to the chosen embedding, and (ii) calibration guarantees that are stable under representation learning. The latter is particularly challenging because the calibration set is no longer independent of the representation if the representation is trained on the full dataset, and because the relevant notion of distance (for "neighborhood" success) is itself learned.

Continuous states, neighborhood success, and discretization error. Even in fully observed control, $\mathcal{S}$ is continuous, while our constrained operator is expressed over the finite set $\mathcal{S}_D$. Empirically we interpret "target state" as "target neighborhood" and measure success via membership in a ball $B(s')$. This induces a discretization: the planner chooses among finitely many anchors in $\mathcal{S}_D$, and feasibility is assessed with respect to neighborhoods around these anchors. The resulting approximation error has two components that are not explicit in the tabular theory:

1. *Geometric error*: high-value regions may lie between dataset states, and the nearest dataset state may be dynamically unreachable with high probability even if the region itself is reachable.

2. *Metric error*: the choice of neighborhood $B(s')$ depends on a metric (often in an embedding space), and the success label $\mathbf{1}\{s_{t+1} \in B(s')\}$ may not correspond to semantic reachability.

Both effects can be viewed as a form of additional pessimism: the certified set is conservative relative to the true set of controllable next-state *regions*. A more faithful continuous-state formulation would treat the action as selecting a *distribution* over next states and would constrain a notion of support or density ratio with respect to the dataset next-state distribution. Connecting our target-state graph view to such density-based constraints is an open theoretical step.

What is needed for full function-approximation theory. Our operator-level analysis relies on contraction and monotonicity, which are robust properties, but the moment we implement fitted value iteration with function approximation, additional phenomena appear: projection error, non-realizability, and distribution shift between successive fitted iterations. In the approximate setting, one would like bounds of the schematic form

$$\|V^{\hat{\pi}_\delta} - V^{\pi_{\mathrm{BC}}}\|_\infty \;\geq\; -\underbrace{C(\gamma,\delta)\cdot\varepsilon}_{\text{certification error}} \;-\underbrace{C'(\gamma,\delta)\cdot\mathcal{E}_{\mathrm{proj}}}_{\text{approximation}} \;-\underbrace{C''(\gamma,\delta)\cdot\mathcal{E}_{\mathrm{shift}}}_{\text{sampling/shift}},$$

with explicit constants and with $\mathcal{E}_{\mathrm{proj}}, \mathcal{E}_{\mathrm{shift}}$ defined in terms of the function class and the data distribution. Achieving such a statement requires assumptions analogous to those in approximate dynamic programming: a concentrability (or coverage) coefficient controlling how state distributions induced by the learned policy relate to the dataset distribution, and a stability condition ensuring that the greedy improvement step does not move the policy into regions where the certification model is untrained. Our certified feasibility constraint is intended to reduce precisely this drift, but proving a nontrivial concentrability bound from certification alone is not immediate, because a certified one-step constraint does not control the multi-step state distribution without additional mixing assumptions on the certified transitions.

State-only logs and identifiability of control. A practically important regime is "state-only" (or observation-only) datasets, where actions are missing or unreliable. Our method can be applied by learning an inverse model $g(s, s')$ from state transitions, but this is an additional source of non-identifiability: if multiple actions can explain the same observed transition, then g may choose an action that is inconsistent with the environment's true actuation constraints. In that case, the estimated success probability $\hat{p}(s \to s')$ can be arbitrarily miscalibrated because the intervention $a = g(s, s')$ is not represented in the logged data. A principled approach

would require either (i) additional structure (e.g. known action bounds and smooth dynamics) or (ii) experimental interventions beyond passive offline logs. At minimum, one should treat controller error as part of ApproxErr and expect it to dominate when actions are unobserved.

k-step (option-level) certification and compounding uncertainty. One-step certification is conceptually clean but may be inefficient: long-horizon tasks require many certified steps, and even small false-negative rates can impede exploration of the certified graph. This motivates certifying *k-step* reachability, in which a target corresponds to an option whose internal policy attempts to reach s' within k steps, and feasibility is defined by $\Pr[s_{t+k} \in B(s') \mid s_t = s, \text{option}(s, s')] \geq \delta$. While this can increase effective connectivity, it introduces compounding uncertainty: the success event is rarer, the calibration problem is harder, and the induced failure cost may be larger because the agent commits to the option for multiple steps. A careful design must therefore couple k-step certification with a pessimistic backup that accounts for intra-option failure; a direct analogue of our mass-splitting would replace $\gamma(\delta V(s') + (1-\delta)V_{\min})$ with an appropriate option-value lower bound. Establishing tight ε-dependence in this setting is open.

Summary of open directions. The central open problems are thus (i) replacing (C2) by verifiable and non-vacuous connectivity conditions in continuous control; (ii) providing calibration guarantees under adaptive querying and representation learning; (iii) deriving approximate-DP performance bounds with explicit dependence on projection and shift errors; and (iv) extending certification from one-step to option-level while controlling compounding miscalibration. We regard these as necessary steps for a complete function-approximation theory of certified state-constrained offline RL.