AN INFORMATIONAL CURSE OF HORIZON IN GOAL-CONDITIONED POLICY LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

The difficulty of learning goal-reaching policies is often attributed to a "curse of horizon" that manifests as bias accumulation in temporal-difference backups and noisy advantage estimates. In this work, we identify an additional *informational* curse of horizon in goal-conditioned policy learning, where increasing the goal relabeling horizon can significantly reduce policy generalization and performance. Through a series of controlled experiments with oracle planners, we decouple the goal horizons sampled during training from those that the policy is asked to reach at test time. Even when evaluated only on a sequence of nearby subgoals, goal-conditioned behavioral cloning (BC) policies suffer from severe, training horizon-dependent performance degradation that is mitigated by reinforcement learning (RL) objectives. We explain this phenomenon as a horizon-dependent decrease in the conditional mutual information between actions and hindsight-relabeled goals, and find empirically that both BC and RL policies trained on longer-horizon goals exhibit a shift in sensitivity from goal to state information, as measured by the policy's input Jacobians. Motivated by this observation, we find that distilling the input Jacobians of short-horizon policies into long-horizon policies yields significant performance gains, especially in combinatorial manipulation tasks. Taken together, our results highlight goal relabeling horizon as an important consideration when learning generalist policies from offline data.

1 INTRODUCTION

Learning to reach any state from any other state is an elegantly simple problem that allows an agent to acquire diverse behaviors from data, without supervision (Kaelbling, 1993). A core component of standard goal-conditioned policy learning involves hindsight goal relabeling (Andrychowicz et al., 2017), which allows an agent to learn from arbitrary experiences regardless of the actor's original intentions. However, simply providing a policy with auxiliary input in the form of goal observations or natural language instructions does not necessarily produce a policy that is capable of following such instructions. A goal-reaching policy must learn to meaningfully utilize information about the desired goal to generate actions, and we argue that this dependence can only emerge when knowing the goal provides meaningful *additional* information about the policy learning objective.

We formalize this additional goal-related information as the conditional mutual information (MI) $\mathcal{I}(A; G \mid S)$ between the goal G and action A conditioned on the current state S, which we refer to as the **action-goal mutual information**. Under standard hindsight relabeling, action-goal MI is non-increasing with the relabeling horizon [Proposition 3.2]. This points to a distinct *informational* curse of horizon in goal-conditioned policy learning: increasing the horizon of relabeled training goals not only introduces well-characterized learning challenges (Park et al., 2025), but may also alter *how* the policy is learned. Specifically, as distant goals provide progressively less information about which action should be taken, the policy may converge to a solution that is globally less sensitive to goal information—even when conditioned on nearby goals.

However, disentangling these two effects is surprisingly difficult in standard goal-reaching benchmarks (Park et al., 2024). Long-horizon evaluation goals are typically more challenging and require longer sequences of correct actions, while policies must also be trained on distant goals to simply keep them in-distribution at evaluation time. This makes it virtually impossible to distinguish failures due to increased task difficulty from failures due to weaker goal conditioning.

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.

To isolate the effect of training horizon H_{train} on performance, we construct a series of controlled experiments using an oracle hierarchical planner that "prompts" a learned, goal-conditioned policy with optimal, fixed-distance (sub)goals at test time. We find that behavioral cloning (BC) policies degrade severely as H_{train} increases, even when the distance of goals presented at test time H_{plan} remains fixed. On the other hand, reinforcement learning (RL) policies are substantially more robust to increased training goal horizons, although this robustness is closely tied to the RL objective (weighted BC or reparameterized policy gradient) and policy class (Gaussian or conditional flow matching) [Section 3.1]. These results suggest that poor policy performance is not only due to higher task difficulty, but also a degradation in the learned policy's ability to achieve simpler goals.

Our theoretical analysis predicts that longer-horizon goals provide less action-relevant information, but how might this be reflected in a policy parameterized by a neural network? In our empirical studies, we use the Jacobian norm of the action with respect to the state or goal as a local measure of policy sensitivity. For certain parameterizations, e.g., fixed-variance Gaussians, the Jacobian norm can be directly related to Fisher information [Box 3.3]. Across BC and RL objectives, as well as Gaussian and flow-matching policy classes, we find that increasing the training goal horizon systematically shifts sensitivity away from goal conditioning towards the current state. Furthermore, explicitly reducing the action-goal MI by training on some proportion of arbitrary, unrelated goal states produces a similar reduction in goal sensitivity.

To determine if there is a causal relationship between goal sensitivity and performance, we then ask if preserving the sensitivity structure of policies trained exclusively on short-horizon goals can improve the performance of policies trained on longer-horizon goals. We propose **Sobolev bootstrapping**, a simple modification of existing policy bootstrapping methods that additionally distills the *input Jacobians* of a short-horizon teacher policy into a long-horizon student policy. This simple intervention significantly boosts performance in a suite of **16** diverse locomotion and manipulation environments, with especially large gains in combinatorial manipulation tasks.

2 Preliminaries

We consider the problem of offline goal-conditioned RL, described by a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \mathcal{G})$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $\mathcal{R} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ the goal-conditioned reward function, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ the transition function, and $\mathcal{G}$ the goal space. In this work, we assume the goal space is equivalent to the state space.

In the *offline* setting, we are given a reward-free dataset of environment interactions $\mathcal{D}$ and asked to learn a policy $\pi(\cdot \mid s, g)$ to reach a specified goal g. At test time, the learned policy is deployed into the environment for a set of evaluation goals drawn from some goal distribution $p(g)$. Then, we can write the objective as

$$\mathcal{J}(\pi) = \mathbb{E}_{g \sim p(g), \tau \sim p^{\pi}(\tau)} \left[\sum_{t=0}^{\infty} \gamma^t r\left(s_t, g\right) \right], \tag{1}$$

where $p^{\pi}(\tau)$ is the distribution of trajectories generated by the policy π and the transition function $\mathcal{P}$ during (online) evaluation. Importantly, we do not assume access to $p(g)$ and must learn a general policy that is capable of reaching any goal encountered in the dataset.

2.1 Input Jacobians and policy sensitivity

Our analyses draw inspiration from prior work that studies input gradients, defined simply as the gradient of the model output with respect to its inputs, from the perspective of classifier robustness in computer vision applications. In our work, we extend these ideas to the parameters of a policy action distribution π. For exposition, we first study policies modeled as fixed-variance Gaussian distributions, where $a \sim \mathcal{N}(\mu_\theta(s, g), \mathbf{I})$. Differentiating this output with respect to the state or goal yields the policy's *state and goal Jacobians*,

$$J_\pi(s) = \frac{\partial \mu_\theta(a \mid s, g)}{\partial s} \quad \text{and} \quad J_\pi(g) = \frac{\partial \mu_\theta(a \mid s, g)}{\partial g}. \tag{2}$$

Prior work has established that properties of input gradients such as the magnitude of the ℓ_1 norm and degree of alignment with human perception strongly correlate with model robustness (Tsipras

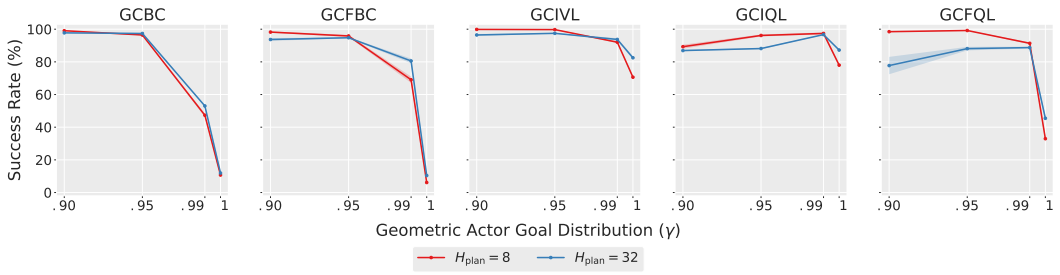

Figure 1: **Training on more distant relabeled goals harms *nearby* goal-reaching capabilities**. We record the success rates across policies trained on a geometric distribution of relabeling goal offsets with discount γ, where $\gamma = 1$ corresponds to a uniform distribution over future states. At evaluation, policies are given optimal H_{plan}-step subgoals from *oracle planners*. Success rates are averaged over the last three checkpoints, three seeds, and four environments: `cube-double`, `cube-triple`, `cube-quadruple`, and `scene`.

et al., 2018; Ganz et al., 2023), and either regularizing the input gradient norm (Drucker & LeCun, 1992; Ross & Doshi-Velez, 2018; Rodríguez-Muñoz et al., 2024) or distilling input gradients from a robust teacher through Sobolev training (Czarnecki et al., 2017; Chan et al., 2020) can significantly improve generalization under perturbations.

3 An Informational Curse of Horizon

Horizon has many opportunities to influence the performance of a goal-reaching policy in subtly different ways. Longer evaluation horizons allow compounding error to accumulate over time, reducing the success rate of a policy with a fixed per-step error rate ϵ (Ross & Bagnell, 2010). Furthermore, directly learning to reach distant goals is simply *harder* in the first place, due to noisy advantage estimates and compounding bias in value learning [Appendix A.2].

We identify a third challenge in which increasing the horizon of relabeled training goals, H_{train}, can degrade the policy's ability to reach nearby goals as well. Said differently, the difficulties induced by long-horizon training *are not confined to reaching distant goals*.

To study this effect, we introduce a third horizon: the planning horizon, H_{plan}, which separates the total evaluation horizon H_{eval} from the horizon of the goals presented to the learned policy. Using an oracle planner, we decouple H_{plan} from H_{train} and H_{eval}. This reveals severe, training horizon-dependent degradation for goal-conditioned BC policies and a milder, albeit objective-dependent form of the same phenomenon in RL policies. We provide an information-theoretic explanation for these findings and study how it manifests in policies parameterized by neural networks.

3.1 How does horizon affect performance?

To isolate policy quality from task difficulty, we use an oracle planner that presents the policy with optimal subgoals at a fixed H_{plan} while leaving the underlying evaluation task unchanged. H_{plan} is similar in spirit to the subgoal horizon used in hierarchical approaches, but here we use it only at evaluation rather than learning a separate high-level policy.

We vary the distribution over training goal horizons in three complementary ways. First, we sample future in-trajectory goals from a geometric distribution with increasing discount factors γ, which assign more probability mass to longer horizons while maintaining the same goal support set [Figures 1 & 6]. Then, we directly control the range of goal distances the actor sees by truncating uniform distributions to different maximum offsets [Figure 5]. Finally, we mix in arbitrary, unrelated dataset states as goals, directly weakening the predictive relationship between actions and goals [Figure 7].

Across both the fixed- and geometric-horizon experiments, we observe a clear degradation in nearby goal-reaching competence as BC policies are trained on increasingly distant relabeled goals. Because the evaluation task and H_{plan} are fixed, this effect cannot simply be attributed to the policy being asked to reach more distant goals. Instead, it is clear that increasing H_{train} changes the quality of the

learned actor itself. RL policies exhibit a milder version of the same phenomenon, where the degree of robustness to horizon depends heavily on the objective.

Our results demonstrate more nuance than a monotonic relationship between training goal horizon and performance. Geometric relabeling can retain strong performance even with exposure to long-horizon goals, and providing unrelated dataset states as goals results in consistent performance decline across all algorithms. Taken together, our experiments suggest that temporal or spatial training goal distance alone cannot fully account for policy degradation. To resolve this discrepancy, we hypothesize that the informational relationship between actions and goals is the critical factor. Next, we formalize this intuition and investigate how training on extended horizons alters the learned policy for nearby goals.

3.2 Why should training horizon matter?

To understand why training on extended horizons can degrade nearby goal-reaching performance [Section 3.1], we theoretically analyze the divergence between a goal-conditioned policy and its unconditional behavioral prior, focusing on how sources of goal-dependent action information change with relabeling horizon. To make the analysis tractable, we focus on the KL-regularized objective corresponding to Peng et al. (2019). We use uppercase letters (e.g., S_t) to denote random variables, and lower-case letters (e.g., s_t) to denote their instantiations. The full proof is provided in Section B.

Theorem 3.1 (Divergence Bound from the Goal-Agnostic Prior)**.** *Let $\pi^*(a \mid s, g) \propto p_{\mathrm{HR}}(a \mid s, g) \exp(Q(s, a, g)/\alpha)$ be the target policy induced by a KL-regularized policy update, where α is a temperature parameter and p_{HR} is the distribution of tuples $(S, A, G) = (S_t, A_t, S_{t+K})$ induced by the dataset trajectory distribution and a hindsight relabeling rule $K \sim q_{\mathrm{HR}}$. Let $\beta(a \mid s)$ be the goal-agnostic behavior policy induced by the dataset, and let the critic value span be $\Delta_Q(s, g) = \sup_a Q(s, a, g) - \inf_a Q(s, a, g)$ over the support of p_{HR}. Then, the expected squared total variation distance between the two policies is bounded by:*

$$\mathbb{E}_{S,G}\left[D_{\mathrm{TV}}\left(\pi^*(\cdot \mid S, G), \beta(\cdot \mid S)\right)^2\right] \leq \frac{\mathbb{E}_{S,G}\left[\Delta_Q(S, G)^2\right]}{8\alpha^2} + I_{\mathrm{HR}}(A; G \mid S)$$

This bound reveals that π^* degenerates toward the goal-agnostic prior when the value function weakly distinguishes among supported actions relative to α, and the relabeled goal contains little additional information about the action when conditioned on the current state. Critically, both terms decay over the horizon. Notably, the first value-based term applies only to RL, whereas the second is applicable to both BC and RL.

The diminishing action gap. The action gap quantifies the difference in value between the optimal action and sub-optimal actions at a given state (Farahmand, 2011). In goal-conditioned RL with sparse rewards and a discount factor $\gamma < 1$, the relative advantage of any single primitive action decays exponentially as the goal distance $d(s, g)$ increases. Consequently, the expected value span $\Delta_Q(S, G)$ shrinks, and the Q-values provide progressively less information about the optimal action distribution. As $d(s, g) \to \infty$, the first term approaches zero.

Proposition 3.2 (Monotonic Decay of Mutual Information)**.** *Let dataset trajectories be generated by a Markovian behavior policy and transition kernel. For any time horizon (relabeling offset) $k' > k > 0$, given the current state S_t, the sequence $A_t \to S_{t+k} \to S_{t+k'}$ forms a conditional Markov chain. The mutual information between the action and future states monotonically decreases over time:*

$$\begin{aligned} &I\left(A_t; S_{t+k} \mid S_t\right) - I\left(A_t; S_{t+k'} \mid S_t\right) \\ &\qquad = \mathbb{E}_{S_t, S_{t+k}, S_{t+k'}}\left[D_{\mathrm{KL}}\left(p(A_t \mid S_t, S_{t+k}) \| p(A_t \mid S_t, S_{t+k'})\right)\right] \geq 0 \end{aligned}$$

This inequality is strict when there exist at least two intermediate states, $s_{t+k}^{(1)}$ and $s_{t+k}^{(2)}$, that share a common future successor state $s_{t+k'}$ with non-zero probability, but yield distinct intermediate action posteriors $p(A_t \mid s_t, s_{t+k}^{(1)}) \neq p(A_t \mid s_t, s_{t+k}^{(2)})$.

Non-increasing mutual information. BC suffers a catastrophic performance degradation when trained on more distant relabeled goals (see GCBC and GCFBC in Figure 1), which is unsurprising as

it does not benefit from the additional goal-related information imparted by a goal-conditioned value function. We formalize this as a value-independent reduction in mutual information $I_{\text{HR}}(A; G \mid S)$ as the relabeling horizon grows [Proposition 3.2]. An extended discussion is included in Section B.

This bound only guarantees non-increasing information. Strictly decreasing information occurs in environments and datasets where distinct trajectories eventually *merge* at a shared future state, which can occur in two common scenarios. First, when learning from a sub-optimal policy, subsequent behavior can "overwrite" the effects of earlier actions. An agent might make a mistake and later correct it, reaching the same future state as a trajectory that took an optimal path. Second, merging occurs when there are multiple valid ways to accomplish the same goal. For example, in a sequential pick-and-place task over three objects, there are up to six valid orderings that all lead to the same final configuration. In both scenarios, longer task and relabeling horizons provide more opportunities for these action-distinguishable trajectories to merge. Thus, temporally distant relabeled goals provide progressively less information about the immediate action.

Our analysis extends to distributions over relabeling offsets when $K \sim q_{\text{HR}}$ is sampled independently of the dataset trajectory. The chain rule for mutual information gives $I_{\text{HR}}(A; G \mid S) = \mathbb{E}_{K \sim q_{\text{HR}}}[I(A_t; S_{t+K} \mid S_t)] - I_{\text{HR}}(A; K \mid S, G) \leq \mathbb{E}_{K \sim q_{\text{HR}}}[I(A_t; S_{t+K} \mid S_t)]$. Therefore, by Proposition 3.2, shifting q_{HR} towards longer offsets decreases this upper bound on the action-goal mutual information.

3.3 From action-goal mutual information to policy sensitivity

While our theoretical results characterize goal dependence as a property of the full distribution, it is difficult to measure this quantity in a policy parameterized by a neural network. One local measure of sensitivity is the input gradient (or Jacobian, for vector-valued outputs). As introduced in Section 2.1, we use the Jacobian of the policy action with respect to the state and goal as a first-order measure of policy sensitivity. A large goal Jacobian indicates that small changes in the goal induce large changes in the action, whereas a small goal Jacobian indicates local invariance to the goal.

It is important to acknowledge that this quantity is not an estimator of $I(A; G \mid S)$. Rather, it allows us to locally probe the action-goal dependence whose global behavior is captured by mutual information. For fixed-variance Gaussian policies, the squared Frobenius norm of the input Jacobian corresponds to the trace of the Fisher information matrix (FIM), and directly measures how rapidly the policy distribution changes with respect to changes in goal information [Box 3.3].

To investigate whether the goal Jacobian norm reflects the predicted decrease in action-goal MI, we measure the ℓ_1 goal Jacobian norm proportion for policies trained on the goal distributions described in Section 3.1. The geometric [Figures 2 & 8] and truncated uniform distributions [Figure 9] vary the proportion of temporally distant goals, which we expect to be less informative about the immediate action. The random goal mixture more directly reduces action-goal MI by introducing unrelated target states [Figure 10]. Across BC and RL objectives, as well as Gaussian and flow-matching policy classes, we observe a consistent, *global* shift in sensitivity away from the goal towards the current state across matched state-goal distances as the training horizon increases. As predicted, we see that goal sensitivity collapses for the BC policies, which do not utilize additional value-related information [Section 3.2].

4 Sobolev Bootstrapping

The above analysis establishes a strong relationship between action-goal information and local policy sensitivity, where policies trained on longer-horizon goals exhibit reduced sensitivity to goal conditioning and a corresponding increase in sensitivity to the current state. This motivates us to investigate whether intervening on the policy input Jacobian can mitigate performance degradation. Direct ℓ_1 and ℓ_2 penalties on the state Jacobian yield modest gains on simpler tasks [Table 2], providing evidence for a causal link between the policy sensitivity and downstream performance. However, these naive penalties fail to scale to more complex, multi-object environments.

We argue that a more principled Jacobian intervention should be *context-dependent*, and propose **Sobolev bootstrapping**. Our approach uses a policy trained only on short-horizon goals as a first-order teacher for the full goal-conditioned "student" policy. Rather than globally suppressing state sensitivity via a uniform norm penalty, we regularize the long-horizon policy's state Jacobian towards

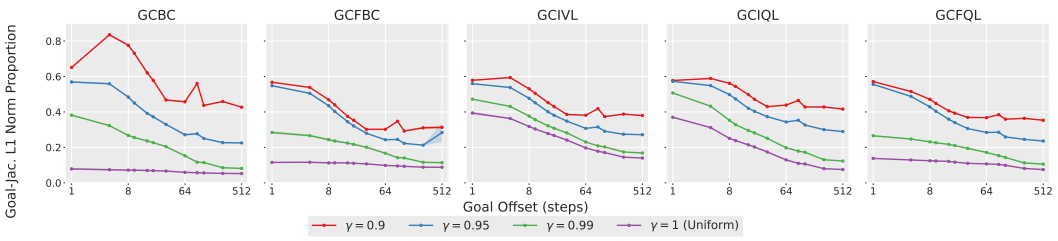

Figure 2: **Increasing training goal horizon *globally* decreases policy goal sensitivity**. We measure the proportion of the input Jacobian norm corresponding to the goal at fixed state-goal offsets, plotted with $\log_2$ scaling in `cube-triple`. Results are averaged over the last three training checkpoints, three seeds, and 1024 randomly sampled state-goal pairs per horizon, with additional environments in Figure 8.

Intuition: Fisher Information Regularization

When π is a Gaussian distribution with fixed variance σ^2, the state Jacobian $J_\pi(s)$ is the square root of the Fisher information matrix (FIM)

$$\mathcal{I}(s) = \mathbb{E}_{\pi_\theta}\left[\nabla_s \log \pi_\theta(a \mid s, g) \cdot \nabla_s \log \pi_\theta(a \mid s, g)^T\right]$$

with respect to the state. Consequently, the squared Frobenius norm of the state Jacobian, $\|J_s(s, g)\|_F^2$, when scaled by σ^2, corresponds exactly to the trace of the FIM and serves as a scalar measure of the local information the policy extracts from the state input s.

Viewed through this lens, our Sobolev bootstrapping objective [Equation 3] acts as a dynamic form of **Fisher information regularization**. By penalizing deviations between the student's state Jacobian and that of the short-horizon teacher, we enforce the local information geometry of the long-horizon policy to be similar to that of the short-horizon policy.

This also provides a mechanical explanation for the findings in Section 5.3 that regularizing the *state* Jacobian also increases the norm of the *goal* Jacobian. By regularizing local state dependence, Sobolev bootstrapping prevents the network from converging to a solution which exploits state-only optimization shortcuts and ignores goal conditioning inputs [Section 3].

that of the short-horizon policy, explicitly distilling the desirable sensitivity structure learned in a regime of high action-goal information. We additionally note that, for fixed-variance Gaussian policies, this objective can be interpreted as a form of Fisher information regularization [Box 3.3].

4.1 Using short-horizon policies as teachers

As the name suggests, Sobolev bootstrapping builds on a class of policy extraction algorithms which perform **policy bootstrapping** (Chane-Sane et al., 2021; Zhou & Kao, 2025). These algorithms provide a natural foundation for our proposed method, since they already maintain a separate one-step policy trained only on short-horizon goals.

Due to space constraints, we provide only a brief intuition here and defer detailed descriptions of the value learning and policy extraction algorithms to Section D. Policy bootstrapping methods leverage the insight that, if an action a is optimal for reaching a subgoal s_g along the optimal path to goal g, then a is also optimal for reaching g. Additionally, goal-conditioned policies are more likely to be accurate for short-horizon subgoals s_g, due to the decreasing "signal-to-noise" ratio of advantage estimates over the horizon (Park et al., 2023). Policy bootstrapping therefore uses $\pi(a \mid s, s_g)$ as a stronger, short-horizon supervisory signal than pure value-based policy extraction, by minimizing the Kullback-Leibler (KL) divergence between their action distributions

$$\arg\min_\theta D_{\mathrm{KL}}\left(\pi_\theta(a \mid s, g) \| \pi^{\mathrm{sub}}(a \mid s, s_g)\right).$$

But how do we find good subgoals for a given goal? Existing approaches such as Reinforcement Learning with Imagined Subgoals (Chane-Sane et al., 2021, RIS) and Subgoal Advantage-Weighted

Environment	Dataset	GCBC	GCIVL	GCIQL	QRL	CRL	HIQL	SAW	S^{2}AW
pointmaze	pointmaze-medium-navigate-v0	9 ±6	63 ±6	53 ±8	82 ±5	29 ±7	79 ±5	**97** ±2	**96** ±3
	pointmaze-large-navigate-v0	29 ±6	45 ±5	34 ±3	**86** ±9	39 ±7	58 ±5	**85** ±10	**83** ±8
	pointmaze-giant-navigate-v0	1 ±2	0 ±0	0 ±0	68 ±7	27 ±10	46 ±9	68 ±8	**73** ±9
antmaze	antmaze-medium-navigate-v0	29 ±4	72 ±8	71 ±4	88 ±3	**95** ±1	**96** ±1	**97** ±1	**97** ±1
	antmaze-large-navigate-v0	24 ±2	16 ±5	34 ±4	75 ±6	83 ±4	**91** ±2	**90** ±3	**93** ±2
	antmaze-giant-navigate-v0	0 ±0	0 ±0	0 ±0	14 ±3	16 ±3	65 ±5	73 ±4	**87** ±3
humanoidmaze	humanoidmaze-medium-navigate-v0	8 ±2	24 ±2	27 ±2	21 ±8	60 ±4	**89** ±2	**88** ±3	**90** ±2
	humanoidmaze-large-navigate-v0	1 ±0	2 ±1	2 ±1	5 ±1	24 ±4	49 ±4	46 ±4	**57** ±5
	humanoidmaze-giant-navigate-v0	0 ±0	0 ±0	0 ±0	1 ±0	3 ±2	12 ±4	35 ±4	**39** ±5
cube	cube-single-play-v0	6 ±2	53 ±4	68 ±6	5 ±1	19 ±2	44 ±9	72 ±5	**86** ±5
	cube-double-play-v0	1 ±1	36 ±3	40 ±5	1 ±0	10 ±2	6 ±2	40 ±7	**57** ±6
	cube-triple-play-v0	1 ±1	1 ±0	3 ±1	0 ±0	4 ±1	3 ±1	4 ±2	**30** ±5
	cube-quadruple-play-v0	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0	**3** ±2
scene	scene-play-v0	5 ±1	42 ±4	51 ±4	5 ±1	19 ±2	38 ±3	63 ±6	**74** ±6
puzzle	puzzle-3x3-play-v0	2 ±0	6 ±1	**95** ±1	1 ±0	3 ±1	12 ±2	9 ±3	15 ±4
	puzzle-4x4-play-v0	0 ±0	13 ±2	26 ±3	0 ±0	0 ±0	7 ±2	26 ±3	**41** ±3

Table 1: **Evaluating Sobolev bootstrapping on offline goal-conditioned RL tasks.** We compare our method's average (binary) success rate (%) against the numbers reported in Park et al. (2024) and Zhou & Kao (2025) across the five test-time goals for each environment, averaged over the last three checkpoints and eight seeds ± standard deviations. Numbers within 5% of the best value in the row are in **bold**.

Policy Bootstrapping (Zhou & Kao, 2025, SAW) differ primarily in how they answer this question. Similar to a bilevel hierarchical policy, RIS learns a generative model $\pi^h(s_g \mid s, g)$ to produce "imagined" subgoals. However, explicitly learning a hierarchical policy is known to suffer from compounding error and bottlenecks performance (Ahn et al., 2025). On the other hand, SAW avoids these optimization difficulties by sampling only from in-trajectory waypoints $s_g \sim p^{\mathcal{D}}(s_{t+k} \mid s_t, g)$. It then re-weights the bootstrapping loss by $\exp(A(s, s_g, g))$, using a subgoal advantage estimate to prioritize high-quality waypoints [Equation 12]. The advantage $A(s, s_g, g)$ is approximated by the learned value function as $V(s_g, g) - V(s, g)$, following Park et al. (2023).

4.2 Input Jacobian distillation

While standard policy bootstrapping only distills the action distribution of the short-horizon teacher, our experiments in Section 3 suggest that the input Jacobian of a short-horizon teacher may also provide useful supervisory signals. Drawing inspiration from Sobolev training (Czarnecki et al., 2017), we augment the standard zeroth-order distillation with a first-order objective that matches the student's state Jacobian to that of the teacher policy $\pi^{\text{sub}}(a \mid s, s_g)$:

$$\mathcal{J}_{\text{SB}} = \mathbb{E}_{s,a,s_g,g}\left[\left\|\frac{\partial \pi_\theta(a \mid s, g)}{\partial s} - \frac{\partial \pi^{\text{sub}}(a \mid s, s_g)}{\partial s}\right\|_2^2\right]. \tag{3}$$

Intuitively, the short-horizon teacher provides a prior not only for what the long-horizon policy should *do*, but also what it should *pay attention to* when deciding what to do. We match only the state Jacobian because the two policies receive different goal inputs: π^{sub} is conditioned on a nearby subgoal s_g and π on the final goal g. Interestingly, we find that regularizing the state Jacobian induces a corresponding change in the *goal* Jacobian, which we discuss in more detail in Section 5.3.

4.3 A practical policy extraction algorithm

Sobolev bootstrapping requires access to differentiable *subgoal*-conditioned "teacher" policies, making existing policy bootstrapping algorithms a natural foundation for our approach. For our primary results in Table 1, we instantiate Sobolev bootstrapping on top of SAW, which we denote as **S^{2}AW** [Algorithm 1], due to its simplicity and strong performance. We modify the objective in Equation 3 to add the same subgoal advantage-weighted coefficient as in SAW [Section D], resulting in the objective

$$\mathcal{J}_{\text{SB}} = \mathbb{E}_{s,a,s_g,g}\left[e^{A(s,s_g,g)} \left\|J_{\pi_\theta}(s) - J_{\pi^{\text{sub}}}(s)\right\|_2^2\right]. \tag{4}$$

While we focus our main analysis on S^{2}AW, we also demonstrate the effectiveness of Sobolev bootstrapping on RIS, achieving substantial gains over the baseline algorithm [Section H.2].

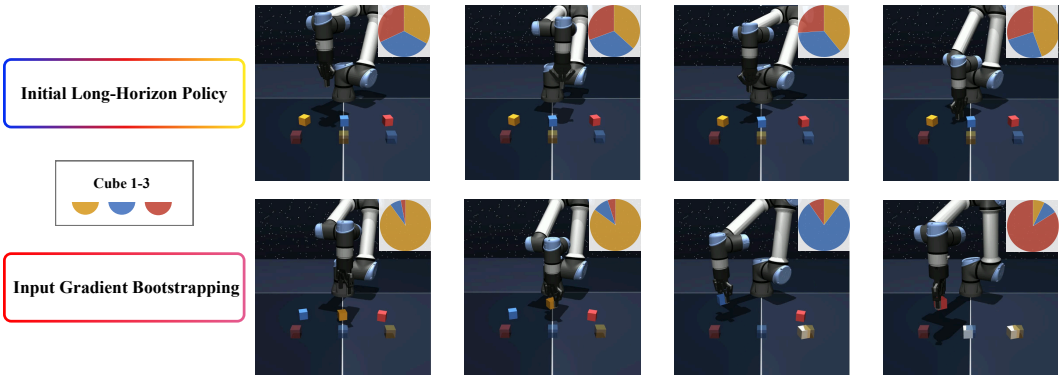

Figure 3: **Sobolev bootstrapping exhibits more interpretable sensitivity structure**. We plot the proportion of the total input Jacobian norm corresponding to each cube during a triple pick-and-place *on-policy* rollout for SAW without (top) and with (bottom) Sobolev bootstrapping. For fairness, we also analyze a scripted optimal trajectory in Figure 11.

5 Experiments

We evaluate Sobolev bootstrapping on the OGBench suite (Park et al., 2024), which spans long-horizon locomotion and combinatorial manipulation tasks. Our experiments address two central questions: (1) does Sobolev bootstrapping improve goal-reaching performance, and (2) does it reverse the horizon-induced shift in policy sensitivity identified in our previous experiments? Interestingly, although Sobolev bootstrapping only distills the *state* Jacobian, we find that it also restores goal-conditioning sensitivity [Section 5.3].

5.1 Sobolev bootstrapping boosts long-horizon performance

Adding Sobolev bootstrapping to SAW (S^2AW) improves performance across most environments [Table 1], with the largest gains in combinatorial manipulation tasks such as the `cube` and `puzzle` environments. In particular, S^2AW improves from 40% to 57% on `cube-double`, 4% to 30% on `cube-triple`, and 26% to 41% on `puzzle-4x4`. We also observe substantial improvements in long-horizon locomotion, although these results may be partially attributed to architectural differences [Appendix H.1]. Additional horizon reduction baselines are provided in Section H.3.

The improvements in the discrete `puzzle` environments demonstrate that first-order distillation can be practically useful even when the true decision boundary is non-smooth, since the short-horizon teacher policy provides a (piecewise) smooth, differentiable approximation of these hard boundaries.

5.2 Effects on state sensitivity

To examine how Sobolev bootstrapping qualitatively changes policy sensitivity to state information, we decompose the state Jacobian norm by object over the course of a `cube-triple` rollout [Figure 3], allowing us to visualize where the policy's "attention" is allocated. Vanilla SAW exhibits diffuse sensitivity across all cubes and immediately fails at the first task stage, whereas S^2AW dynamically allocates sensitivity to the focal cube at each stage and successfully completes the full triple pick-and-place task. To control for differences in policy competence, we repeat the analysis on an independent, scripted optimal trajectory in Figure 11 and observe the same pattern.

We hypothesize that the observed shift in sensitivity away from goal information and towards state inputs may also result in policies that overfit to task-irrelevant "distractors." To test this, we evaluate policies trained in multi-cube environments on single pick-and-place tasks. The first evaluation task in `cube-double` and `triple` already tests this capability, and we add a similar task to `cube-quadruple`, which we refer to as Task 6. While naive ℓ_1 and ℓ_2 Jacobian norm penalties produce small improvements, context-dependent Sobolev bootstrapping significantly improves performance even on `cube-triple` and `quadruple` [Table 2].

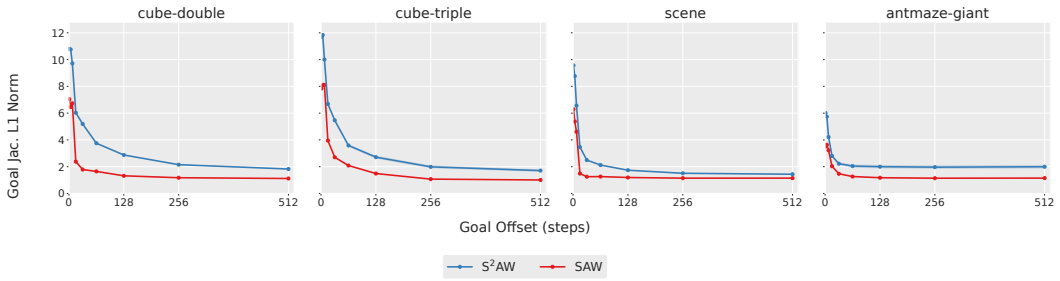

Figure 4: **Sobolev bootstrapping recovers policy goal sensitivity**. We compare the *absolute* goal Jacobian norm between S^2AW and SAW, averaging over batches of goals sampled at fixed intervals from the current state up to the maximum horizon. Results are averaged across four seeds and 1024 state-goal pairs for each horizon, with shaded standard deviations.

Environment	GCIQL	GCIQL$_{\ell_1}$	SAW	SAW$_{\ell_1}$	SAW$_{\ell_2}$	S^2AW
`cube-single`	68 ± 6	74 ± 4	76 ± 8	**85** ± 6	**85** ± 4	**86** ± 5
`cube-double-task1`	74 ± 8	76 ± 8	64 ± 9	**81** ± 9	65 ± 14	**78** ± 12
`cube-triple-task1`	13 ± 3	16 ± 10	18 ± 11	24 ± 9	17 ± 12	**74** ± 9
`cube-quadruple-task6`	0 ± 1	0 ± 1	2 ± 2	2 ± 2	1 ± 1	**35** ± 14

Table 2: **Sobolev bootstrapping improves single pick-and-place performance in the presence of distractor objects.** We evaluate policies trained in multi-cube environments on single pick-and-place tasks. While naive ℓ_1 or ℓ_2 Jacobian regularization [Appendix H.4] yields modest improvements on simpler tasks, Sobolev bootstrapping matches or outperforms all baselines. Other settings are identical to Table 1.

5.3 Effects on Goal Conditioning

Although Sobolev bootstrapping explicitly matches only the *state* Jacobian, we find that it induces an increase in the policy's *absolute* goal Jacobian norm [Figure 4]. This suggests that the policy has converged to a different, more goal-sensitive solution instead of simply smoothing out state sensitivity. We note that a decrease in sensitivity to distant goals is unsurprising—for instance, perturbing the target placement in a pick-and-place task should not affect the policy before grasping the cube. However, standard long-horizon policies exhibit *global* insensitivity across all goal distances, including *nearby* goals [Figure 4], and it is this pathology that Sobolev bootstrapping addresses.

6 Discussion

In this work, we identify an *informational* curse of horizon in goal-conditioned policy learning. By decoupling the training goal horizon from the subgoal distance presented at test time, we show that policies trained on distant relabeled goals can become worse at reaching nearby goals. We explain this effect through the decay of action-goal mutual information under hindsight relabeling: as training goals become more distant, they provide progressively less information about the immediate action. This is accompanied by a global shift in sensitivity away from the goal and toward the current state in neural network policies, across all goal distances. Although input Jacobians are only a local probe of this dependence, explicitly weakening action-goal information produces a corresponding collapse in goal sensitivity. We then propose Sobolev bootstrapping, which exploits this observation by distilling the sensitivity structure of short-horizon policies into long-horizon policies and achieves large gains in combinatorial manipulation tasks. Overall, our results suggest that the goal-relabeling horizon should be treated as an important design choice, rather than a mere consequence of task horizon.

Our work does have several limitations. We observed that the optimal goal-horizon distribution depends strongly on the policy objective, prompting further study on how different policy extraction methods utilize goal information. Additionally, input Jacobians capture only local sensitivity and are not direct estimators of mutual information, and the Sobolev bootstrapping algorithm derived from those insights introduces significant additional computation. These limitations motivate future work on horizon-aware relabeling and policy-learning objectives that preserve goal dependence without explicit Jacobian distillation.

7 STATEMENTS

AI USE STATEMENT

In this work, we used generative AI tools to assist in the writing of proofs. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, support qualitative and thematic data analysis, or interpret results. Generating synthetic datasets, assisting with translation, and cleaning and reformatting datasets are not applicable to this work. Additionally, we used generative AI tools to assist with plotting code. We have reviewed all AI-assisted work. Figure generation and plotting code was inspected by the lead authors, and proofs were verified by hand. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

This work uses open-source benchmarks and datasets, and all code will be open-sourced upon paper release. We provided full proofs of all claims in Appendix B. All hyperparameters are specified in Appendix F, and compute resources used to conduct the experiments are described in Appendix G.

REFERENCES

Ahn, H., Choi, H., Han, J., and Moon, T. Option-aware temporally abstracted value for offline goal-conditioned reinforcement learning. In *Neural Information Processing Systems*, 2025.

Amit, R., Meir, R., and Ciosek, K. Discount factor as a regularizer in reinforcement learning. In *International Conference on Machine Learning*, 2020.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W. Hindsight experience replay. In *Neural Information Processing Systems*, 2017.

Boucheron, S., Lugosi, G., and Massart, P. *Concentration inequalities : a non asymptotic theory of independence*. Oxford University Press, 2013.

Chan, A., Tay, Y., and Ong, Y.-S. What it thinks is important is important: Robustness transfers through input gradients. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Chane-Sane, E., Schmid, C., and Laptev, I. Goal-conditioned reinforcement learning with imagined subgoals. In *International Conference on Machine Learning*, 2021.

Czarnecki, W. M., Osindero, S., Jaderberg, M., Świrszcz, G., and Pascanu, R. Sobolev training for neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2017.

Drucker, H. and LeCun, Y. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992.

Du, Y., Yang, M., Dai, B., Dai, H., Nachum, O., Tenenbaum, J. B., Schuurmans, D., and Abbeel, P. Learning universal policies via text-guided video generation. arXiv preprint arXiv:2302.00111, 2023.

Eysenbach, B., Zhang, T., Salakhutdinov, R., and Levine, S. Contrastive learning as goal-conditioned reinforcement learning. In *Neural Information Processing Systems*, 2022.

Farahmand, A.-m. Action-gap phenomenon in reinforcement learning. In *Neural Information Processing Systems*, 2011.

Fujimoto, S. and Gu, S. A minimalist approach to offline reinforcement learning. In *Neural Information Processing Systems*, 2021.

Ganz, R., Kawar, B., and Elad, M. Do perceptually aligned gradients imply robustness? In *International Conference on Machine Learning*, 2023.

Hu, H., Yang, Y., Zhao, Q., and Zhang, C. On the role of discount factor in offline reinforcement learning. In *International Conference on Machine Learning*, 2022.

Jiang, N., Kulesza, A., Singh, S., and Lewis, R. The dependence of effective planning horizon on model accuracy. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2015.

Jiang, N., Singh, S., and Tewari, A. On structural properties of MDPs that bound loss due to shallow planning. In *International Joint Conferences on Artificial Intelligence*, 2016.

Kaelbling, L. P. Learning to achieve goals. In *IJCAI*, 1993.

Korniak, M., Dybek, K., Eysenbach, B., Bagatella, M., and Bortkiewicz, M. Three steps at a time: Learning representations from action sequences in contrastive rl. arXiv preprint arXiv:2608.30640, 2026.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2021.

Levine, S. Reinforcement learning and control as probabilistic inference: Tutorial and review. arXiv preprint arXiv:1805.00909, 2018.

Li, L., Zhang, Q., Luo, Y., Yang, S., Wang, R., Han, F., Yu, M., Gao, Z., Xue, N., Zhu, X., Shen, Y., and Xu, Y. Causal world modeling for robot control. arXiv preprint arXiv:2601.21998, 2026a.

Li, Q., Zhou, Z., and Levine, S. Reinforcement learning with action chunking. In *Neural Information Processing Systems*, 2025.

Li, Q., Park, S., and Levine, S. Decoupled q-chunking. In *International Conference on Learning Representations*, 2026b.

Mishra, U. A., Chen, Y., Xu, D., Liu, Y., Chen, X., and Mao, J. Understanding and mitigating the video-action generalization gap via temporal ratio. arXiv preprint arXiv:2607.08127, 2026.

Park, S., Ghosh, D., Eysenbach, B., and Levine, S. HIQL: Offline goal-conditioned RL with latent states as actions. In *Neural Information Processing Systems*, 2023.

Park, S., Frans, K., Eysenbach, B., and Levine, S. OGBench: Benchmarking offline goal-conditioned RL. In *International Conference on Learning Representations (ICLR)*, 2024.

Park, S., Frans, K., Mann, D., Eysenbach, B., Kumar, A., and Levine, S. Horizon reduction makes RL scalable. In *Neural Information Processing Systems*, 2025.

Park, S., Oberai, A., Atreya, P., and Levine, S. Transitive RL: Value learning via divide and conquer. In *International Conference on Learning Representations*, 2026.

Peng, X. B., Kumar, A., Zhang, G., and Levine, S. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. arXiv preprint arXiv:1910.00177, 2019.

Physical Intelligence, Ai, B., Amin, A., Aniceto, R., Balakrishna, A., Balke, G., Black, K., Bokinsky, G., Cao, S., Charbonnier, T., Choudhary, V., Collins, F., Conley, K., Connors, G., Darpinian, J., Dhabalia, K., Dhaka, M., DiCarlo, J., Driess, D., Equi, M., Esmail, A., Fang, Y., Finn, C., Glossop, C., Godden, T., Goryachev, I., Groom, L., Habeeb, H., Hancock, H., Hausman, K., Hussein, G., Hwang, V., Ichter, B., Jacobsen, C., Jakubczak, S., Jen, R., Jones, T., Kammerer, G., Katz, B., Ke, L., Khadikov, M., Kuchi, C., Lamb, M., LeBlanc, D., LeCount, B., Levine, S., Li, X., Li-Bell, A., Lialin, V., Liang, Z., Lim, W., Lu, Y., Luo, E., Mano, V., Marwaha, N., Mongush, A., Murphy, L., Nair, S., Patterson, T., Pertsch, K., Ren, A. Z., Schelske, G., Sharma, C., Shi, B., Shi, L. X., Smith, L., Springenberg, J. T., Stachowicz, K., Stoeckle, W., Tang, J., Tanner, J., Tekeste, S., Torne, M., Vedder, K., Vuong, Q., Walling, A., Wang, H., Wang, J., Wang, X., Whalen, C., Whitmore, S., Williams, B., Xu, C., Yoo, S., Yu, L., Zhang, W., Zhang, Z., and Zhilinsky, U. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. arXiv preprint arXiv:2604.15483, 2026.

Rodríguez-Muñoz, A., Wang, T., and Torralba, A. Characterizing model robustness via natural input gradients. In *European Conference on Computer Vision*, 2024.

Ross, A. and Doshi-Velez, F. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *AAAI Conference on Artificial Intelligence*, 2018.

Ross, S. and Bagnell, D. Efficient reductions for imitation learning. In *International Conference on Artificial Intelligence and Statistics*, 2010.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2018.

Wang, T., Torralba, A., Isola, P., and Zhang, A. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*, 2023.

Zheng, B., Myers, V., Eysenbach, B., and Levine, S. Scaling goal-conditioned reinforcement learning with multistep quasimetric distances. In *International Conference on Learning Representations*, 2026.

Zhou, J. L. and Kao, J. Flattening hierarchies with policy bootstrapping. In *Neural Information Processing Systems*, 2025.

Zhou, P., Chen, L., Chen, S., Chen, D., Zhao, W., Jin, R., Ren, G., and Luo, J. Act2goal: From world model to general goal-conditioned policy. *arXiv preprint arXiv:2512.23541*, 2025.

A Related Work

A.1 Offline goal-conditioned policy learning

The objective of offline goal-conditioned policy learning is to learn a policy which can reach any state from any other state from a fixed dataset of offline trajectories. This can be accomplished with goal-conditioned behavioral cloning (GCBC) that relabels state-action pairs with future trajectory states as goals; or goal-conditioned reinforcement learning (GCRL), which makes use of learned goal-conditioned value functions. It combines the challenges of conservative offline RL with the sparse rewards and recursive structure of goal-reaching tasks (Kaelbling, 1993). Standard algorithms typically use behavior-regularized actor-critic methods or learn an *implicit* value function with in-sample value iteration (Kostrikov et al., 2021), then extract a policy using weighted behavioral cloning (Peng et al., 2019) or reparameterized policy gradient with a behavioral cloning penalty (Fujimoto & Gu, 2021). The recursive structure of goal-conditioned RL also admits a variety of non-traditional value learning algorithms based on successor measures (Eysenbach et al., 2022) and quasimetric structure (Wang et al., 2023; Park et al., 2026).

A.2 The curse of horizon

The challenges introduced by horizon in sequential decision-making tasks are well known. In addition to the error bounds of Ross & Bagnell (2010) in imitation learning, reinforcement learning introduces a whole host of additional horizon-related difficulties. The effective task horizon, set by the discount factor γ, is known to control the complexity of the policy class (Jiang et al., 2015; 2016), and lower discount factors can provide pessimistic (Hu et al., 2022) and regularizing effects (Amit et al., 2020). However, artificially decreasing the discount factor is not feasible when the true task horizon is long and reward signals are sparse, as in goal-reaching tasks.

Increasing the discount factor in off-policy RL leads to a host of additional problems, typically attributed to bias accumulation in value learning (Park et al., 2025) and difficulty in advantage estimation for primitive actions in policy extraction. To mitigate these difficulties, recent approaches have employed various techniques to reduce the effective horizon relative to decision-making frequency, both for value learning (Ahn et al., 2025; Zheng et al., 2026) and policy extraction (Li et al., 2025; 2026b). Our work specifically focuses on the effects of horizon on *goal-conditioned* policy learning, where we identify a particular *informational* curse of horizon: as the horizon increases, the informative signal between the policy and the goal diminishes, causing the policy to overfit to spurious state features at the expense of goal conditioning.

A.3 Steerable robot policies

Effective goal conditioning is increasingly critical for modern robotic systems, including subgoal-conditioned policies (Zhou et al., 2025; Physical Intelligence et al., 2026) and inverse-kinematic-style World Action Models (Du et al., 2023; Li et al., 2026a). Recent works demonstrate the significance of effectively utilizing goal information: Mishra et al. (2026) links video-action model (VAM) performance to a model's reliance on the generated future frames, as measured through cross-attention ratios. Korniak et al. (2026) shows that action chunking increases the mutual information between the state-action pair and the goal, which facilitates critic learning in goal-conditioned RL.

B Extended Proofs

Below, we use uppercase letters (e.g., S_t) to denote random variables, and lower-case letters (e.g., s_t) to denote their instantiations.

Lemma B.1. *Let P be a base probability distribution and let $f(x)$ be a bounded scalar function such that $f(x) \in [a, b]$ for all x in the support of P. Define the exponentially reweighted distribution P_f as $P_f(x) = \frac{P(x)e^{f(x)}}{Z}$, where $Z = \mathbb{E}_{X\sim P}[e^{f(X)}]$ is the partition function. Then, the Kullback-Leibler divergence is bounded by:*

$$D_{\mathrm{KL}}(P_f \| P) \leq \frac{(b-a)^2}{8}$$

Proof. The KL divergence between a base distribution P and its exponentially reweighted version $P_f \propto Pe^f$ can be expressed in terms of the log-partition function $\psi(\lambda) = \log \mathbb{E}_{X\sim P}[e^{\lambda f(X)}]$, where $\lambda \in [0, 1]$ is a scalar parameter that defines the intermediate distributions between the base distribution and the exponentially reweighted target. By taking a second-order Taylor expansion of $\psi(\lambda)$, we can rewrite this KL divergence as the remainder $\frac{1}{2}\psi''(c)$ for some $c \in (0, 1)$. Because $f(X)$ is bounded in $[a, b]$, we can apply Hoeffding's Lemma (see Lemma 2.2 in Boucheron et al. (2013)) to bound this second derivative, yielding a maximum KL divergence of $\frac{(b-a)^2}{8}$. □

Theorem B.2 (Divergence Bound from the Goal-Agnostic Prior). *Let $\pi^*(a \mid s, g) \propto p_{\mathrm{HR}}(a \mid s, g)e^{Q(s,a,g)/\alpha}$ be the target distribution induced by the KL-regularized policy update, where α is a temperature parameter and p_{HR} is the distribution of tuples $(S, A, G) = (S_t, A_t, S_{t+K})$ induced by the dataset trajectory distribution and a hindsight relabeling rule $K \sim q_{\mathrm{HR}}$. Let $\beta(a \mid s)$ be the goal-agnostic behavior policy induced by the dataset, and let the critic value span be $\Delta_Q(s, g) = \sup_a Q(s, a, g) - \inf_a Q(s, a, g)$ over the support of p_{HR}. Then, the expected squared total variation distance between the two policies is bounded by:*

$$\mathbb{E}_{S,G}\left[D_{\mathrm{TV}}\left(\pi^*(\cdot \mid S, G), \beta(\cdot \mid S)\right)^2\right] \leq \frac{\mathbb{E}_{S,G}\left[\Delta_Q(S, G)^2\right]}{8\alpha^2} + I_{\mathrm{HR}}(A; G \mid S)$$

Proof. We bound the total variation distance (TVD) between the target policy $\pi^*(\cdot \mid s, g)$ and the behavioral policy $\beta(\cdot \mid s)$ by introducing the hindsight-relabeled distribution $p_{\mathrm{HR}}(\cdot \mid s, g)$ as an intermediate distribution. We denote $\pi^*(\cdot \mid s, g)$ as π^* below for brevity. The same notation applies to $\beta(\cdot \mid s)$ and $p_{\mathrm{HR}}(\cdot \mid s, g)$.

Because TVD is a metric, it satisfies the triangle inequality. For any given state s and goal g:

$$D_{\mathrm{TV}}(\pi^*, \beta) \leq D_{\mathrm{TV}}(\pi^*, p_{\mathrm{HR}}) + D_{\mathrm{TV}}(p_{\mathrm{HR}}, \beta).$$

After squaring both sides and applying the inequality $(x + y)^2 \leq 2x^2 + 2y^2$, we have

$$D_{\mathrm{TV}}(\pi^*, \beta)^2 \leq 2D_{\mathrm{TV}}(\pi^*, p_{\mathrm{HR}})^2 + 2D_{\mathrm{TV}}(p_{\mathrm{HR}}, \beta)^2.$$

Pinsker's inequality, $D_{\mathrm{TV}}(P, Q)^2 \leq \frac{1}{2}D_{\mathrm{KL}}(P\|Q)$, applied to $2D_{\mathrm{TV}}(\pi^*, p_{\mathrm{HR}})^2$, gives us

$$2D_{\mathrm{TV}}(\pi^*, p_{\mathrm{HR}})^2 \leq D_{\mathrm{KL}}(\pi^*\|p_{\mathrm{HR}}).$$

Because π^* is an exponentially reweighted version of p_{HR}, we can apply Lemma 1 to $D_{\mathrm{KL}}(\pi^*\|p_{\mathrm{HR}})$ with $f(a) = Q(s, a, g)/\alpha$, yielding

$$D_{\mathrm{KL}}(\pi^*\|p_{\mathrm{HR}}) \leq \frac{\Delta_Q(s, g)^2}{8\alpha^2}.$$

Now we have shown

$$D_{\mathrm{TV}}(\pi^*, \beta)^2 \leq \frac{\Delta_Q(s, g)^2}{8\alpha^2} + 2D_{\mathrm{TV}}(p_{\mathrm{HR}}, \beta)^2. \tag{5}$$

Next, we apply Pinsker's inequality again to $2D_{\mathrm{TV}}(p_{\mathrm{HR}}, \beta)^2$, leading to

$$2D_{\mathrm{TV}}(p_{\mathrm{HR}}, \beta)^2 \leq D_{\mathrm{KL}}(p_{\mathrm{HR}}(\cdot \mid s, g)\|\beta(\cdot \mid s)).$$

Note that hindsight relabeling does not change the marginal state distribution, implying $\beta(a \mid s) = p_{\mathrm{HR}}(a \mid s)$. Then taking the expectation of this KL divergence over S and G yields the conditional mutual information between actions and goals given the state:

$$\mathbb{E}_{S,G}\left[D_{\mathrm{KL}}(p_{\mathrm{HR}}(\cdot \mid S, G)\|p_{\mathrm{HR}}(\cdot \mid S))\right] = I_{\mathrm{HR}}(A; G \mid S) \tag{6}$$

Taking the expectation of equation 5 over S and G and combining it with equation 6 concludes the proof. □

Proposition B.3 (Monotonic Decay of Mutual Information)**.** *Let dataset trajectories be generated by a Markovian behavior policy and transition kernel. For any time horizon (relabeling offset) $k' > k > 0$, given the current state S_t, the sequence $A_t \to S_{t+k} \to S_{t+k'}$ forms a conditional Markov chain. The mutual information between the action and future states monotonically decreases over time:*

$$I\left(A_t; S_{t+k} \mid S_t\right) - I\left(A_t; S_{t+k'} \mid S_t\right) = \mathbb{E}_{S_t, S_{t+k}, S_{t+k'}}\left[D_{\mathrm{KL}}\left(p(A_t \mid S_t, S_{t+k}) \| p(A_t \mid S_t, S_{t+k'})\right)\right] \geq 0$$

This inequality is strict when there exist at least two intermediate states, $s_{t+k}^{(1)}$ and $s_{t+k}^{(2)}$, that share a common future successor state $s_{t+k'}$ with non-zero probability, but yield distinct intermediate action posteriors $p(A_t \mid s_t, s_{t+k}^{(1)}) \neq p(A_t \mid s_t, s_{t+k}^{(2)})$.

Proof. Expand $I(A_t; S_{t+k}, S_{t+k'} \mid S_t)$ with mutual information chain rule in both ways and leverage $I\left(A_t; S_{t+k'} \mid S_t, S_{t+k}\right) = 0$ implied by the Markov chain, we have

$$I\left(A_t; S_{t+k} \mid S_t\right) - I\left(A_t; S_{t+k'} \mid S_t\right) = I\left(A_t; S_{t+k} \mid S_t, S_{t+k'}\right) \geq 0.$$

Next, we rewrite $I\left(A_t; S_{t+k} \mid S_t, S_{t+k'}\right)$ as an expected KL divergence

$$I(A_t; S_{t+k} \mid S_t, S_{t+k'}) = \mathbb{E}_{S_t, S_{t+k'}}\left[D_{\mathrm{KL}}\left(p(A_t, S_{t+k} \mid S_t, S_{t+k'}) \| p(A_t \mid S_t, S_{t+k'}) p(S_{t+k} \mid S_t, S_{t+k'})\right)\right].$$

Since $p(A_t, S_{t+k} \mid S_t, S_{t+k'}) = p(S_{t+k} \mid S_t, S_{t+k'}) p(A_t \mid S_t, S_{t+k})$, we can simplify the KL term

$$I(A_t; S_{t+k} \mid S_t, S_{t+k'}) = \mathbb{E}_{S_t, S_{t+k'}, (S_{t+k} \mid S_t, S_{t+k'})}\left[D_{\mathrm{KL}}\left(p\left(A_t \mid S_t, S_{t+k}\right) \| p\left(A_t \mid S_t, S_{t+k'}\right)\right)\right].$$

To see when this inequality is strict (i.e., when mutual information strictly decreases with an increase in relabeling horizon), we compare the intermediate action posterior $p(A_t \mid S_t, S_{t+k})$ and the future action posterior $p(A_t \mid S_t, S_{t+k'})$ within the KL term. We instantiate some start state s_t and future state $s_{t+k'}$ to focus on the KL term inside the expectation. By the law of total probability, the future posterior is the marginalization of the intermediate posteriors over all possible intermediate states:

$$p(A_t \mid s_t, s_{t+k'}) = \mathbb{E}_{S_{t+k} \mid s_t, s_{t+k'}}[p(A_t \mid s_t, S_{t+k})].$$

Now, to make the KL term non-zero, we need $\mathbb{E}_{S_{t+k} \mid s_t, s_{t+k'}}[p(A_t \mid s_t, S_{t+k})] \neq p(A_t \mid s_t, s_{t+k})$ for some instantiation s_{t+k}. In other words, the intermediate posterior $p(A_t \mid s_t, S_{t+k})$ cannot be almost-surely constant with respect to S_{t+k}.

For the intermediate posterior $p(A_t \mid s_t, S_{t+k})$ to vary depending on S_{t+k}, we need at least two instantiations, $s_{t+k}^{(1)}$ and $s_{t+k}^{(2)}$, that simultaneously satisfy the two conditions below:

Merging paths: both intermediate states must be viable nodes on the path from the start state s_t to the future state $s_{t+k'}$. In other words, they must both occur with non-zero probability given the endpoints:

$$p(s_{t+k}^{(1)} \mid s_t, s_{t+k'}) > 0 \quad \text{and} \quad p(s_{t+k}^{(2)} \mid s_t, s_{t+k'}) > 0$$

Different actions: knowing which of these intermediate states the agent passed through must change our belief about the action taken at time t:

$$p(A_t \mid s_t, s_{t+k}^{(1)}) \neq p(A_t \mid s_t, s_{t+k}^{(2)})$$

The two conditions are satisfied and strict loss of information occurs in environments and datasets where distinct trajectories eventually "merge" at a shared future state. Intuitively, this can occur in two common scenarios. First, when learning from a sub-optimal policy, subsequent behavior can *overwrite* the effects of earlier actions. An agent might make a mistake and later correct it, reaching the same future state as a trajectory that took an optimal path. Second, merging occurs when there are

multiple valid ways to accomplish the same goal. For example, in a task requiring an agent to pick and place three cubes, there are six valid orderings that all lead to the same final state. In both scenarios, longer task and relabeling horizons provide more opportunities for these action-distinguishable trajectories to merge. Thus, temporally distant relabeled goals become progressively less informative about the original action. □

C Algorithm Pseudocode

Algorithm 1 Sobolev Bootstrapping

Input: offline dataset $\mathcal{D}$, goal distribution $p(g)$.
Initialize value function V_β, target subpolicy π_ϕ^{sub}, and policy π_θ.
while not converged **do**
  Train value function: $\beta \leftarrow \beta - \lambda \nabla_\beta \mathcal{L}_{\text{GCIVL}}(\beta)$ with $(s_t, s_{t+1}) \sim p^{\mathcal{D}}, g \sim p(g)$ [Equation 9]
end while
while not converged **do**
  Train target subpolicy: $\phi \leftarrow \phi + \lambda \nabla_\phi \mathcal{J}_{\text{AWR}}(\phi)$ with $(s_t, a, s_g) \sim p^{\mathcal{D}}$ [Equation 13]
end while
while not converged **do**
  Train policy: $\theta \leftarrow \theta - \lambda \nabla_\theta \left(\mathcal{J}_{\text{SB}}(\theta) - \mathcal{J}_{\text{SAW}}(\theta)\right)$ with $(s_t, a, s_g) \sim p^{\mathcal{D}}, g \sim p(g)$ [Equation 4 & 12]
end while

D Algorithmic Details

In this section, we provide full details of the value and policy learning objectives for our Sobolev bootstrapping algorithm, which is instantiated on top of Subgoal Advantage-Weighted Policy Bootstrapping (Zhou & Kao, 2025, SAW).

D.1 Goal-conditioned implicit value learning (GCIVL)

To learn a value function $V(s, g)$ that can operate both on single- and multi-step transitions, we use a goal-conditioned variant of implicit Q-learning (IQL) (Kostrikov et al., 2021), which approximates the $\max$ operator with an expectile regression on in-sample state-action pairs:

$$\mathcal{L}_V(\beta_V) = \mathbb{E}_{(s,a,g)\sim D}\left[L_2^\tau(Q_{\bar{\beta}_Q}(s,a,g) - V_{\beta_V}(s,g))\right] \tag{7}$$

$$\mathcal{L}_Q(\beta_Q) = \mathbb{E}_{(s,a,s',g)\sim D}\left[(r(s,g) + \gamma V_{\beta_V}(s',g) - Q_{\beta_Q}(s,a,g))^2\right], \tag{8}$$

where $L_2^\tau(x) = |\tau - \mathbb{1}(x < 0)|x^2$ is the expectile loss and $\tau \in [0.5, 1)$ is the expectile parameter.

Following previous works (Park et al., 2023), we adopt an action-free variant of this objective, which estimates the state-action value function $Q(s, a, g)$ as $r(s, g) + \gamma \bar{V}(s', g)$, leading to

$$\mathcal{L}_{\text{GCIVL}}(\beta) = \mathbb{E}_{(s,s',g)\sim D}\left[L_2^\tau(r(s,g) + \gamma V_{\bar{\beta}}(s',g) - V_\beta(s,g))\right]. \tag{9}$$

Note that using the action-free estimator with expectile regression is optimistically biased in stochastic environments, since it directly regresses towards high-value transitions without using Q-values to marginalize over environment stochasticity. However, it performs well in practice and can be used to measure the relative advantages of intermediate *subgoals* without modification.

D.2 Offline policy extraction

The above value iteration algorithm already performs implicit policy improvement. Then, we simply need to extract the optimal policy from the learned value function. To remain inside the convex hull of dataset actions and avoid querying the value function for out-of-distribution actions, we use a form of weighted behavioral cloning called Advantage-Weighted Regression (Peng et al., 2019, AWR):

$$\mathcal{J}_{\pi_\theta}(\theta) = \mathbb{E}_{(s,a,g)\sim D}\left[\exp(\alpha(\bar{Q}(s,a,g) - V(s,g)))\log \pi_\theta(a \mid s, g)\right], \tag{10}$$

where $\alpha \in [0, \infty)$ is an inverse temperature parameter. Because we do not directly parameterize a critic $Q(s, a, g)$ in GCIVL [Equation 9], we instead estimate the "one-step" advantage $\bar{Q}(s, a, g) - V(s, g)$ as $V(s', g) - V(s, g)$ to reweight the corresponding action in the transition tuple (s, a, s'). This estimate is biased in stochastic environments, since it optimistically assumes that the transition from s to s' was caused by action a.

D.3 Policy bootstrapping algorithms

Policy bootstrapping algorithms are based on the insight that, if an action a is optimal for reaching a subgoal s_g along the optimal trajectory toward goal g, then a should also be optimal for reaching g. To evaluate the optimality of subgoals s_g, we can use a multistep variant of the action-free advantage estimator for GCIVL, where we approximate $A(s, s_g, g)$ as $V(s_g, g) - V(s, g)$.

Reinforcement Learning with Imagined Subgoals [RIS] (Chane-Sane et al., 2021): To produce subgoals, RIS first learns a generative model $\pi^h_\psi(s_g \mid s, g)$ to produce "imagined" subgoals s_g, and then bootstraps a goal-conditioned flat policy $\pi_\theta(a \mid s, g)$ on a short-horizon policy $\pi^{\text{sub}}(a \mid s, s_g)$ conditioned on the imaginary way-points.

The generative model $\pi^h_\psi(s_g \mid s, g)$ is optimized to produce subgoals with high advantages while remaining close to the dataset state distribution μ_D, in a manner similar to AWR:

$$\mathcal{J}_{\pi^h_\psi}(\psi) = \mathbb{E}_{(s, s_g, g) \sim D} \left[\exp(\alpha A(s_g \mid s, g)) \log \pi^h_\psi(s_g \mid s, g) \right]. \tag{11}$$

Subgoal Advantage-Weighted Policy Bootstrapping [SAW] (Zhou & Kao, 2025): SAW introduces a unifying view of RIS and a related hierarchical method (Park et al., 2023) under the "RL as inference" framework (Levine, 2018). Rather than modeling the optimal subgoal distribution $p^*(s_g \mid s, g)$. SAW samples in-trajectory waypoints in dataset and uses a similar multistep advantage weight to prioritize bootstrapping on high-advantage subgoals.

$$p^*(s_g \mid s, g) \propto p^D(s_g \mid s) \exp(A(s_g \mid s, g)).$$

This eliminates the need for a parametric subgoal generator and replaces Equation 11 and 14 with

$$\mathcal{J}^{\text{SAW}}_{\pi_\theta}(\theta) = \mathbb{E}_{(s, a, s_g, g) \sim D} \left[e^{\alpha_1 A(a \mid s, g)} \log \pi_\theta(a \mid s, g) - e^{\alpha_2 A(s_g \mid s, g)} D_{\text{KL}}(\pi_\theta \| \pi^{\text{sub}}) \right], \tag{12}$$

where the target subpolicy π^{sub} is extracted with Equation 13, π_θ is a shorthand for $\pi^f_\theta(\cdot \mid s, g)$, and π^{sub} denotes $\pi^{\text{sub}}(\cdot \mid s, s_g)$.

Following Zhou & Kao (2025), we implement the offline variant of RIS and SAW with an independently parameterized short-horizon policy with AWR, instead of using an exponential moving average of the long-horizon policy π^θ as in the original implementation by Chane-Sane et al. (2021):

$$\mathcal{J}_{\pi^{\text{sub}}}(\phi) = \mathbb{E}_{(s, a, s_g) \sim D} \left[\exp(\alpha A(a \mid s, s_g)) \log \pi^{\text{sub}}(a \mid s, s_g) \right]. \tag{13}$$

Finally, we train the long-horizon policy π_θ with "one-step" transitions while regressing it toward the short-horizon policy π^{sub} conditioned on generated subgoals:

$$\mathcal{J}^{\text{RIS}}_{\pi_\theta}(\theta) = \mathbb{E}_{(s, a, g) \sim D, s_g \sim \pi^h_\psi(\cdot \mid s, g)} \left[\exp(\alpha A(a \mid s, g)) \log \pi_\theta(a \mid s, g) - D_{\text{KL}}(\pi_\theta \| \pi^{\text{sub}}) \right], \tag{14}$$

where the "one-step" advantage $A(a \mid s, g)$ is estimated by $V_{\beta_V}(s', g) - V_{\beta_V}(s, g)$, π_θ is a shorthand for $\pi^f_\theta(\cdot \mid s, g)$, and π^{sub} denotes $\pi^{\text{sub}}(\cdot \mid s, s_g)$.

E ADDITIONAL FIGURES

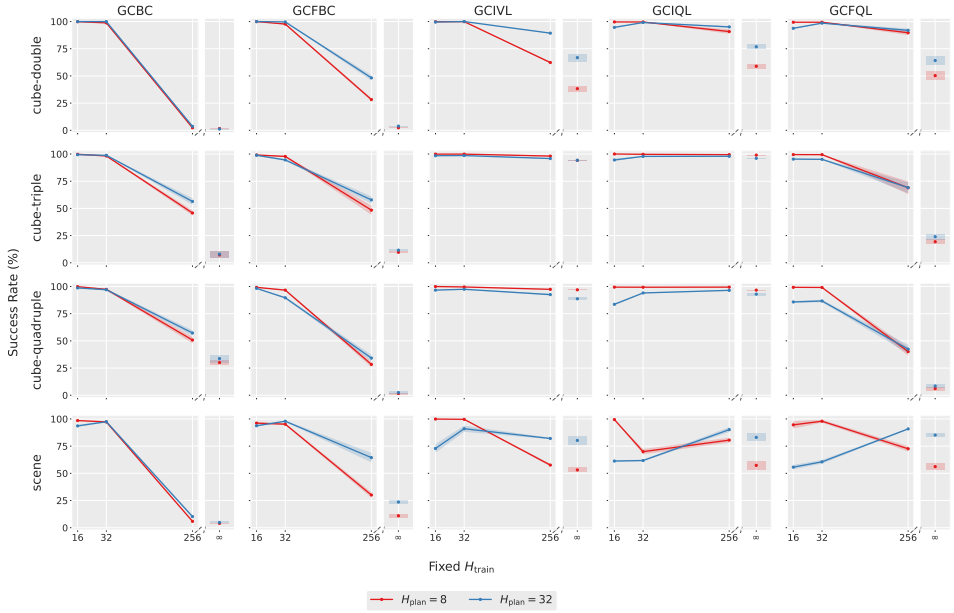

Figure 5: **Training on distant relabeled goals harms *nearby* goal-reaching capabilities**. We record the success rates across learned policies trained on a uniform distribution of relabeling goal offsets $[1, H_{\text{train}}]$, where $H_{\text{train}} = \infty$ corresponds to a uniform distribution over all future in-trajectory states. At evaluation, policies are given optimal H_{plan}-step subgoals from oracle planners. Success rates are averaged over the last three checkpoints and three seeds.

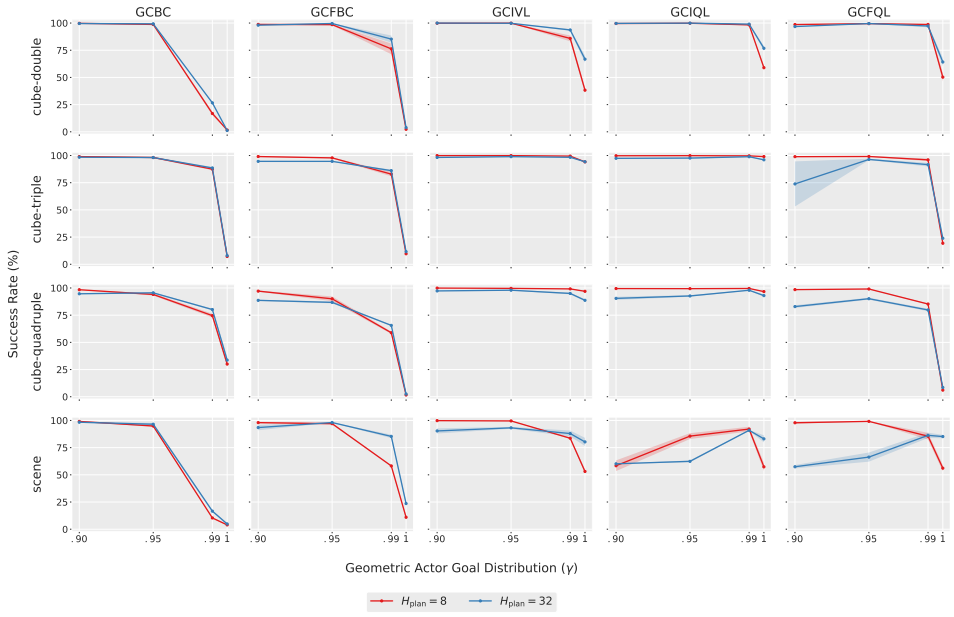

Figure 6: **Training on a larger proportion of distant relabeled goals harms *nearby* goal-reaching capabilities**. We record the success rates across learned policies trained on a geometric distribution of relabeling goal offsets with discount γ, where $\gamma = 1$ corresponds to a uniform distribution over future states. At evaluation, policies are given optimal H_{plan}-step subgoals from oracle planners. Success rates are averaged over the last three checkpoints and three seeds.

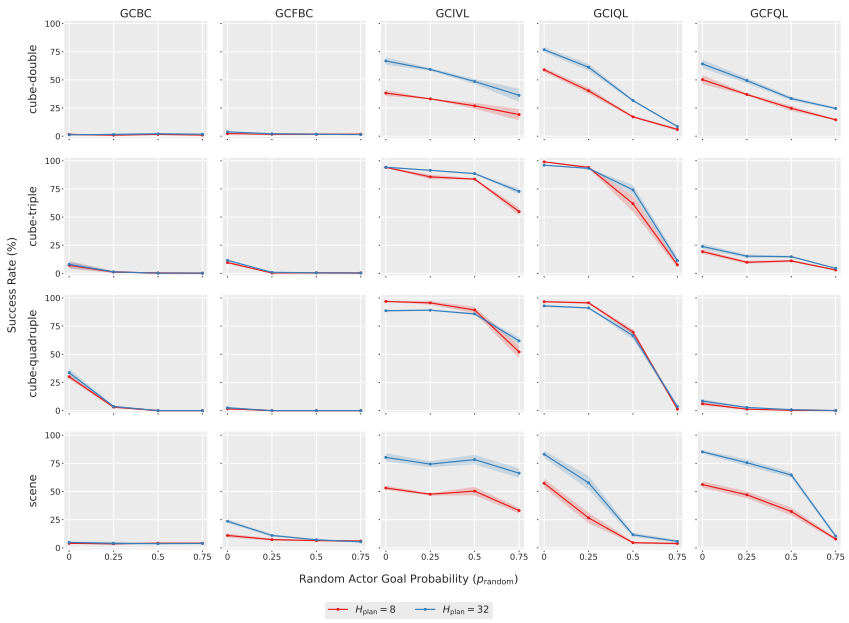

Figure 7: **Training on more random relabeled goals harms *nearby* goal-reaching capabilities**. We record the success rates across learned policies trained on a mixture of goals either sampled from a uniform distribution over future states or from an arbitrary dataset state with probability $p_{\text{rand}} \in \{0.25, 0.50, 0.75\}$. At evaluation, policies are given optimal H_{plan}-step subgoals from oracle planners. Success rates are averaged over the last three checkpoints and three seeds.

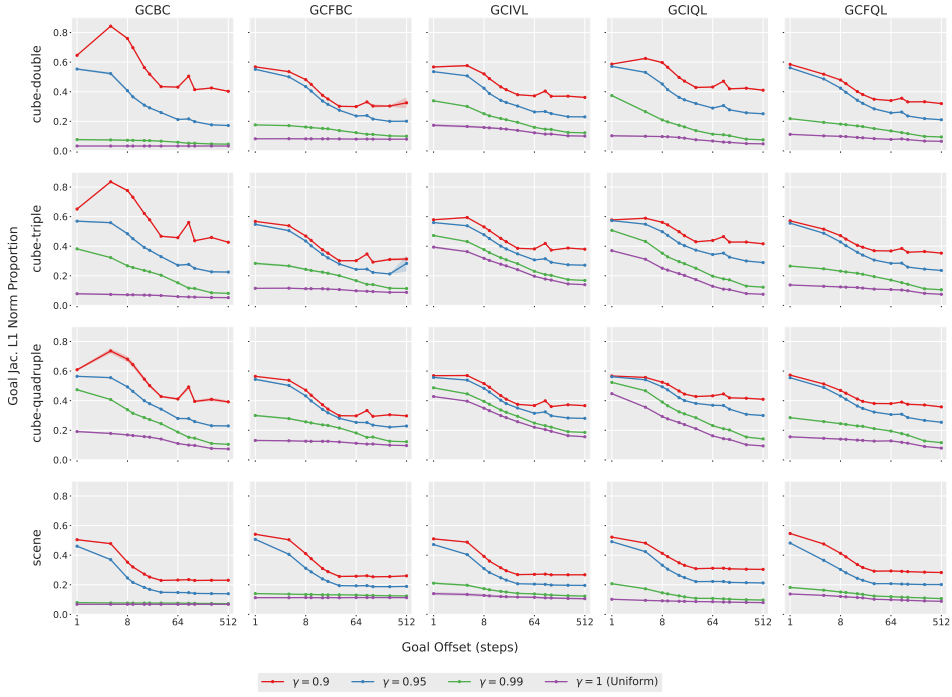

Figure 8: **Increasing training goal horizon *globally* decreases policy goal sensitivity**. We measure the proportion of the input Jacobian norm corresponding to the goal at fixed state-goal offsets, plotted with $\log_2$ scaling. Results are averaged over the last three training checkpoints, three seeds, and 1024 randomly sampled state-goal pairs per horizon.

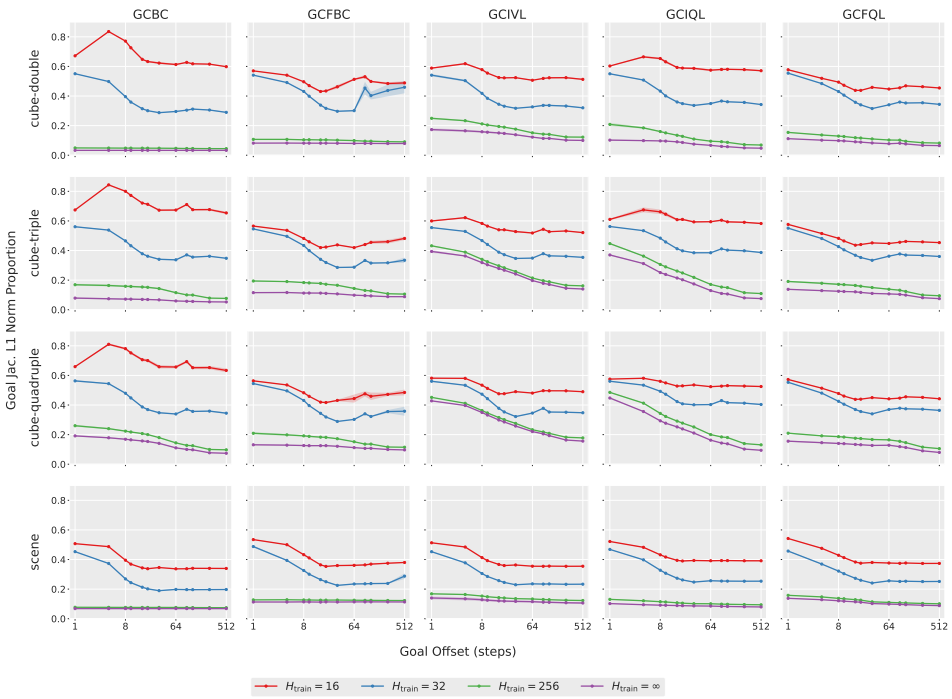

Figure 9: **Training on distant relabeled goals *globally* decreases policy goal sensitivity**. We measure the proportion of the input Jacobian norm corresponding to the goal at fixed state-goal offsets, plotted with $\log_2$ scaling. Results are averaged over the last three training checkpoints, three seeds, and 1024 randomly sampled state-goal pairs per horizon.

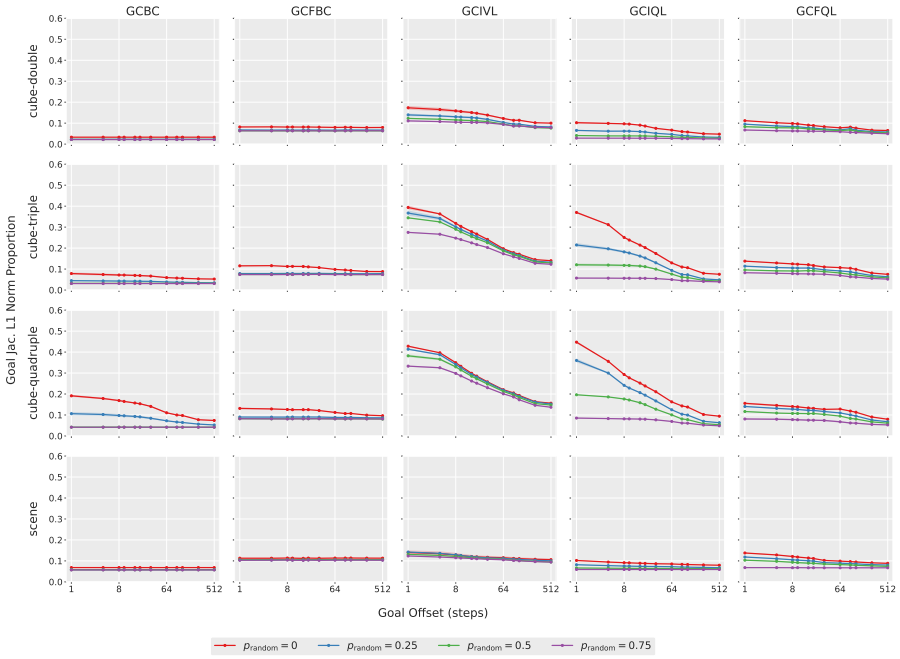

Figure 10: **Random goal sampling *globally* decreases policy goal sensitivity.** We directly reduce the action-goal MI by replacing in-trajectory goals with randomly sampled dataset states with probability $p_{\text{random}} \in \{0.25, 0.5, 0.75\}$, and measure the goal Jacobian norm proportion at fixed state-goal offsets.

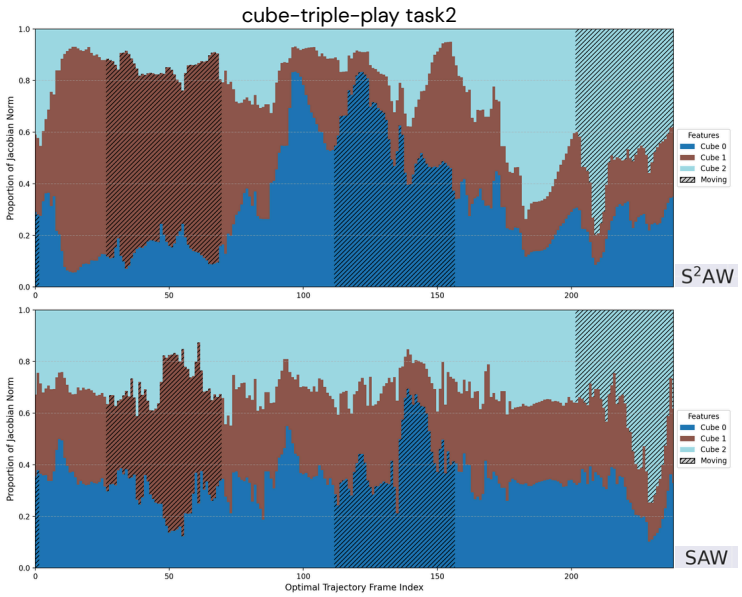

Figure 11: **Sobolev bootstrapping pays attention to the correct object during optimal trajectories.** To enable fair comparisons across later task stages, we plot the input Jacobian norms with respect to each object over an independent scripted optimal trajectory Ahn et al. (2025) for the `cube-triple` triple pick-and-place task. S^2AW (top) allocates a higher proportion of the Jacobian norm to the focal cube both during approach and grasp, whereas SAW's (bottom) Jacobian norm only changes after the cube is already grasped. The shaded area denotes when the cube is actively grasped.

F HYPERPARAMETERS

We provide hyperparameters related to our Sobolev bootstrapping objective in Table 3. We use value learning hyperparameters from Zhou & Kao (2025) for HIQL, SAW, and S^{2}AW in `cube-single`, which significantly improves performance. If not listed, all other hyperparameters follow the default values from Park et al. (2024) and Zhou & Kao (2025).

Dataset	Expectile τ	AWR α	KLD β	Subgoal steps k	Sobolev weight λ
`pointmaze-medium-navigate-v0`	0.7	3.0	3.0	25	0.03
`pointmaze-large-navigate-v0`	0.7	3.0	3.0	25	1.00
`pointmaze-giant-navigate-v0`	0.7	3.0	3.0	25	1.00
`antmaze-medium-navigate-v0`	0.7	3.0	3.0	25	0.03
`antmaze-large-navigate-v0`	0.7	3.0	3.0	25	0.03
`antmaze-giant-navigate-v0`	0.7	3.0	3.0	25	0.30
`humanoidmaze-medium-navigate-v0`	0.7	3.0	3.0	100	0.03
`humanoidmaze-large-navigate-v0`	0.7	3.0	3.0	100	0.03
`humanoidmaze-giant-navigate-v0`	0.7	3.0	3.0	100	0.03
`cube-single-play-v0`	0.9	3.0	0.3	10	0.02
`cube-double-play-v0`	0.7	3.0	1.0	10	0.02
`cube-triple-play-v0`	0.7	3.0	1.0	10	0.03
`cube-quadruple-play-v0`	0.7	3.0	1.0	10	0.03
`scene-play-v0`	0.7	3.0	1.0	10	0.02

Table 3: **Hyperparameters for SAW with Sobolev bootstrapping.**

F.1 SENSITIVITY ANALYSIS

We examine sensitivity to the choice of Sobolev weight λ in Table 4. We do not perform additional tuning of the expectile τ, AWR α, subgoal steps k, and KLD β, which are common hyperparameters shared by HIQL and SAW. Our results show that smaller weights can be insufficient to realize the benefits of Sobolev training, while larger weights may "oversmooth" in some manipulation environments. Nevertheless, we observe consistently substantial gains in `cube-triple` and `antmaze-giant` across all values of λ.

Dataset	$\lambda = 0$	0.001	0.01	0.1
`cube-double-play-v0`	40 ± 7	40 ± 11	55 ± 7	35 ± 9
`cube-triple-play-v0`	4 ± 2	9 ± 2	30 ± 7	23 ± 10
`scene-play-v0`	63 ± 6	54 ± 10	71 ± 5	62 ± 5
`antmaze-giant-navigate-v0`	73 ± 4	76 ± 4	82 ± 4	85 ± 4

Table 4: **Sobolev loss weight sensitivity analysis.** Sobolev bootstrapping is robust to different weighting factors within a reasonable scale (around 0.01 for `cube-double` and `scene`). Notably, improvements on `antmaze-giant` and `cube-triple` are robust across various weighting scales. Experimental results are averaged across four seeds.

G COMPUTATIONAL RESOURCES

All experiments were run on an internal cluster of NVIDIA 5090 and NVIDIA PRO 6000 Blackwell GPUs. In Table 5, We select the highest-dimensional environments and report the hours required to complete a standard 1M-step training run on an NVIDIA 5090 GPU.

hours / 1M iters	GCIQL	SAW	S^{2}AW
`cube-quadruple-play`	0.26	0.46	0.75
`scene-play`	0.25	0.45	0.74
`humanoidmaze-giant-navigate`	0.26	0.46	1.75

Table 5: **Training time comparison.** We select the highest-dimensional environments and report the hours required to complete a standard 1M-step training run on an NVIDIA 5090 GPU.

H Additional Experiments

H.1 Architectural ablations

While we have made the case that differences in state-goal distributions are responsible for the improved properties of the target policy π^{sub}, we note that the experiments in Table 1 use different architectures for each policy. The full goal-conditioned policy π consists of a three-layer MLP, followed by a length-normalized bottleneck of dimension 10, and another three-layer MLP; the target policy consists only of a three-layer MLP. Then, our results may be explained by a natural simplicity bias introduced by smaller network sizes. To control for this potential confound, we train Sobolev SAW using identical architectures for π^{sub} and π, which we denote as $\mathbf{S^2AW}_\ell$.

Environment	Dataset	GCBC	GCIVL	GCIQL	QRL	CRL	HIQL	SAW	S^2AW	S^2AW_ℓ
maze-giant	antmaze-giant-navigate-v0	0 ±0	0 ±0	0 ±0	14 ±3	16 ±3	65 ±5	73 ±4	**87** ±3	80 ±5
	humanoidmaze-giant-navigate-v0	0 ±0	0 ±0	0 ±0	1 ±0	3 ±2	12 ±4	35 ±4	**38** ±6	35 ±5
cube	cube-single-play-v0	6 ±2	53 ±4	68 ±6	5 ±1	19 ±2	44 ±9	72 ±5	**86** ±5	79 ±8
	cube-double-play-v0	1 ±1	36 ±3	40 ±5	1 ±0	10 ±2	6 ±2	40 ±7	50 ±6	**55** ±8
	cube-triple-play-v0	1 ±1	1 ±0	3 ±1	0 ±0	**4** ±1	3 ±1	4 ±2	**30** ±5	**30** ±4
	cube-quadruple-play-v0	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0	**3** ±2	2 ±1
scene	scene-play-v0	5 ±1	42 ±4	51 ±4	5 ±1	19 ±2	38 ±3	63 ±6	**73** ±6	**72** ±5

Table 6: **Architectural ablations to the Sobolev SAW target actor.** We report the performance of Sobolev SAW where the target actor has the same architecture as the full policy, averaged over 4 seeds with standard deviations after the ± sign. Numbers within 5% of the best value in the row are in **bold**.

We find that results remain consistent across all manipulation environments where combinatorial generalization is most important, supporting the primary conclusion that the difference in training curricula, and not architectural differences, produces desirable properties of input gradients. Performance does drop slightly in locomotion environments compared to Sobolev SAW with the smaller target actor, suggesting that a smaller actor network does provide some beneficial regularization. Indeed, we observe that Sobolev bootstrapping with a smaller target policy achieves lower SAW validation loss than its counterpart.

H.2 Extending Sobolev bootstrapping to RIS

Adding Sobolev bootstrapping to RIS (Chane-Sane et al., 2021) also yields significant gains, albeit less consistently. In our implementation of RIS, subgoals generated by π^h_ψ lie in a lower-dimensional latent space rather than the raw observation space, making it impractical to compute advantages directly from the learned value function V_{β_V}. Thus, we adopt a constant weighting scheme for the Sobolev bootstrapping loss, under the assumption that all generated subgoals are near-optimal. If the subgoal generator proposes suboptimal waypoints, then the uniformly weighted Sobolev term in RIS can force the flat policy to match noisy input gradients, which can degrade its performance.

Furthermore, it is well-known that planning with a latent subgoal representation extracted from an intermediate layer of the value function, as done in Park et al. (2023) and our implementation of RIS, is suboptimal and can significantly hamper performance in larger environments such as `antmaze-giant` and `humanoidmaze-giant` (Zhou & Kao, 2025; Ahn et al., 2025). Although not a focus of our work, we expect that improvements to the high-level policy, such as modern generative models and new representation learning objectives, may significantly improve RIS's performance.

H.3 Additional horizon reduction baselines

Existing horizon-reduction approaches address one of two issues: compounding bias in off-policy value learning, e.g., n-step TD (Park et al., 2025), QC (Li et al., 2025; 2026b), OTA (Ahn et al., 2025); or noisy advantage estimates during policy extraction, e.g., HIQL (Park et al., 2023), SAW (Zhou & Kao, 2025). In contrast, our work identifies an additional bottleneck within the actor itself, related to decreased policy sensitivity to goal information scaling with horizon.

For completeness, we compare our method against OTA (Ahn et al., 2025), DQC (Li et al., 2026b), and SHARSA (Park et al., 2025). Because the original OTA paper did not report results on state-based manipulation tasks, we use the official implementation and sweep the abstraction factor

Dataset	RIS	Sobolev RIS
pointmaze-medium-navigate-v0	88 ±6	**96** ±1
pointmaze-large-navigate-v0	63 ±13	**75** ±3
pointmaze-giant-navigate-v0	57 ±12	**67** ±5
antmaze-medium-navigate-v0	**96** ±1	**97** ±1
antmaze-large-navigate-v0	**89** ±3	**90** ±1
antmaze-giant-navigate-v0	**65** ±4	**69** ±0
humanoidmaze-medium-navigate-v0	73 ±5	**84** ±7
humanoidmaze-large-navigate-v0	21 ±7	**47** ±2
humanoidmaze-giant-navigate-v0	3 ±2	**13** ±3
cube-single-play-v0	**81** ±6	59 ±3
cube-double-play-v0	36 ±4	**41** ±2
cube-triple-play-v0	**3** ±2	**3** ±0
cube-quadruple-play-v0	**0** ±0	**0** ±0
scene-play-v0	**64** ±7	**66** ±1

Table 7: **Evaluating RIS with Sobolev bootstrapping.** We evaluate the effectiveness of Sobolev bootstrapping on RIS by comparing it against the numbers reported in Zhou & Kao (2025). Results are averaged over 4 seeds, and other settings are identical to Table 1.

$n \in \{5, 10, 20\}$. Furthermore, adapting DQC and SHARSA to our standard dataset is non-trivial, as they were evaluated using larger datasets with oracle representations. To ensure a fair comparison, we adopt the documented hyperparameters and use the standard batch and model sizes consistent with the OGBench suite. Our results demonstrate that OTA, DQC, and SHARSA underperform with the standard datasets.

Environment	OTA n=5	OTA n=10	OTA n=20	SHARSA	DQC	S^2AW
antmaze-giant-navigate	77±4	-	-	72±1	25±6	87±3
cube-double-play	2±0	3±1	2±1	5±1	24±4	57±6
cube-triple-play	1±0	0±0	1±0	2±1	11±4	30±5

Table 8: **Additional horizon reduction baselines**. The additional baselines do not readily generalize to the standard datasets and perform poorly. All results are averaged across 4 seeds.

H.4 Uniform Jacobian regularization

	SAW	SAW$_{\ell_1}$	SAW$_{\ell_2}$	S^2AW
antmaze-giant-navigate	73±4	72±3	74±2	87±3
cube-double-play	40±7	47±10	37±7	57±6
cube-triple-play	4±2	5±4	4±1	30±5
scene-play	63±6	69±8	62±2	74±6

Table 9: **Uniform Jacobian regularization.** We apply ℓ_1 and ℓ_2 norm penalization with coefficient 0.3 to the SAW baseline. We observe that ℓ_1 penalization is more effective, leading to notable performance gains in `cube-double` and `scene` environments. Results are averaged across four seeds and five evaluation tasks.