Understanding the Complexity Gains of Single-Task RL with a Curriculum

Qiyang Li [*1] **Yuexiang Zhai** [*1] **Yi Ma** [1] **Sergey Levine** [1]

Abstract

Reinforcement learning (RL) problems can be challenging without well-shaped rewards. Prior work on provably efficient RL methods generally proposes to address this issue with dedicated exploration strategies. However, another way to tackle this challenge is to reformulate it as a multi-task RL problem, where the task space contains not only the challenging task of interest but also easier tasks that implicitly function as a curriculum. Such a reformulation opens up the possibility of running existing multi-task RL methods as a more efficient alternative to solving a single challenging task from scratch. In this work, we provide a theoretical framework that reformulates a single-task RL problem as a multi-task RL problem defined by a curriculum. Under mild regularity conditions on the curriculum, we show that sequentially solving each task in the multi-task RL problem is more computationally efficient than solving the original single-task problem, without any explicit exploration bonuses or other exploration strategies. We also show that our theoretical insights can be translated into an effective practical learning algorithm that can accelerate curriculum learning on simulated robotic tasks.

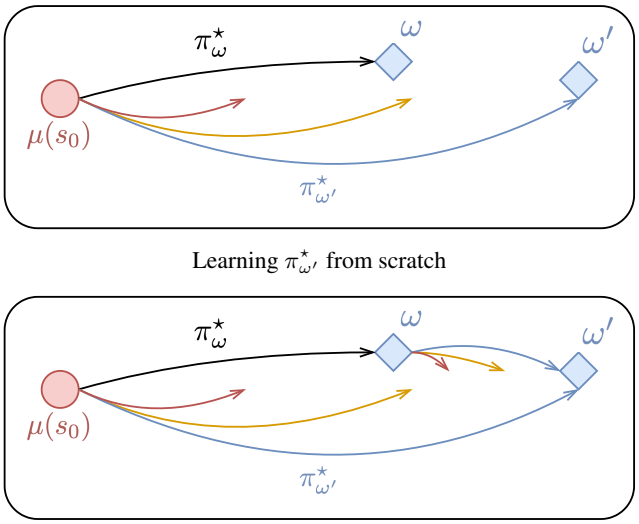


Figure 1: **Illustration of ROLLIN.** **The red circle** represents the initial state distribution. **The dark curve** represents the optimal policy of the preceding task ω. **The blue diamonds** represent the optimal state distributions $d_\mu^{\pi^\star_\omega}, d_\mu^{\pi^\star_{\omega'}}$ of the preceding task ω and the current ω' respectively. ROLLIN runs the optimal policy of the preceding task $\pi^\star_\omega$ to obtain a better initial state distribution for faster learning of the optimal policy of the current task $\pi^\star_{\omega'}$.

1. Introduction

Reinforcement learning (RL) provides an appealing and simple way to formulate control and decision-making problems in terms of reward functions that specify what an agent should do, and then automatically train policies to learn how to do it. However, in practice the specification of the reward function requires great care: if the reward function is well-shaped, then learning can be fast and effective, but if rewards are delayed, sparse, or can only be achieved after extensive explorations, RL problems can be exceptionally difficult (Kakade and Langford, 2002; Andrychowicz et al., 2017; Agarwal et al., 2019). This challenge is often overcome with either reward shaping (Ng et al., 1999; Andrychowicz et al., 2017; 2020; Gupta et al., 2022) or dedicated exploration methods (Tang et al., 2017; Stadie et al., 2015; Bellemare et al., 2016; Burda et al., 2018), but reward shaping can bias the solution away from optimal behavior, while even the best exploration methods, in general, may require covering the entire state space before discovering high-reward regions.

On the other hand, a number of recent works have proposed multi-task learning methods in RL that involve learning contextual policies that simultaneously represent solutions to an entire space of tasks, such as policies that reach any

[1]UC Berkeley. Correspondence to: Qiyang Li <qcli@berkeley.edu>, Yuxiang Zhai <simonzhai@berkeley.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

potential goal (Fu et al., 2018; Eysenbach et al., 2020b; Fujita et al., 2020; Zhai et al., 2022), policies conditioned on language commands (Nair et al., 2022), or even policies conditioned on the parameters of parametric reward functions (Kulkarni et al., 2016; Siriwardhana et al., 2019; Eysenbach et al., 2020a; Yu et al., 2020b). While such methods are often not motivated directly from the standpoint of handling challenging exploration scenarios, but rather directly aim to acquire policies that can perform all tasks in the task space, these multi-task formulations often present a more tractable learning problem than acquiring a solution to a single challenging task in the task space (e.g., the hardest goal, or the most complex language command). We pose the following question:

When do we expect solving the reformulated multi-task RL problem with task-conditioned policies to be more efficient than solving the original single-task problem directly?

In this work, we study this question by analyzing the complexity of learning an optimal policy in the stochastic policy gradient (SPG) setting (Agarwal et al., 2021; Mei et al., 2020; Ding et al., 2021) with a curriculum (learning a list of tasks in sequence). As pointed out by Ding et al. (2021), for learning an optimal policy, SPG requires a polynomial sample complexity if the initialization is near-optimal.[1] In general, there is no guarantee that the initial policy is near-optimal, which could potentially lead to an unbounded density ratio and thus poor sample complexity bound. While there have been a lot of prior works that utilize exploration bonuses to address the sample complexity (Azar et al., 2017; Jin et al., 2018; Agarwal et al., 2020; Zhang et al., 2020d), we take a different approach without the need for exploration bonuses by making use of a curriculum of tasks where adjacent tasks in the curriculum are close in terms of their optimal state visitation distributions. Our algorithm, ROLLIN, works by (1) using the optimal policy parameters of the previous task as an initialization for the current task, and (2) constructing the initial state distribution as a mixture of the optimal state visitation distribution of the previous task and the original initial state distribution of interest. In a nutshell, ROLLIN mixes in the distribution of the optimal policy of the preceding task to the initial distribution to make sure that such distribution is close to the optimal state visitation distribution of the current task, reducing the density mismatch ratio and yielding better sample complexity.

We illustrate the intuition of ROLLIN in Figure 1. We adopt the contextual MDP formulation, where we assume each MDP, $\mathcal{M}_\omega$, is uniquely defined by a context ω in the context space $\mathcal{W}$, and we are given a curriculum $\{\omega_k\}_{k=0}^K$, with the last MDP, $\mathcal{M}_{\omega_K}$, being the MDP of interest. Our main results require a Lipschitz continuity assumption on the context-dependent reward function r_ω and a fixed transition dynamics model across all contexts. We show that learning $\pi_K^\star$ by recursively rolling in with a near-optimal policy for ω_k to construct the initial distribution μ_{k+1} for the next context ω_{k+1}, can have a smaller minimum required sample complexity compared with learning $\pi_{\omega_K}^\star$ from scratch directly. In particular, we show that when an appropriate sequence of contexts is selected, we can reduce the minimum required iteration and sample complexity bounds of entropy-regularized softmax policy gradient (with an inexact stochastic estimation of the gradient) from an original exponential dependency on the state space size, as suggested by Ding et al. (2021), to a polynomial dependency. We also prescribe a practical implementation of ROLLIN.

Our contributions are as follows. We introduce ROLLIN, a simple algorithm that facilitates single-task learning by recasting it as a multi-task problem. Theoretically, we show that under the entropy-regularized softmax policy gradient (PG) setting, our algorithm reduces the exponential complexity bound to a polynomial dependency on S. Empirically, we verify our theory on a tabular MDP and provide a practical implementation of ROLLIN that can accelerate curriculum learning in the tabular environment and a range of simulated robotic tasks.

2. Related Work

Convergence of policy gradient methods. Theoretical analysis of policy gradient methods has a long history (Williams, 1992; Sutton et al., 1999; Konda and Tsitsiklis, 1999; Kakade and Langford, 2002; Peters and Schaal, 2008). Motivated by the recent empirical success (Schulman et al., 2015; 2017) in policy gradient (PG) methods, the theory community has extensively studied the convergence of PG in various settings (Fazel et al., 2018; Agarwal et al., 2021; 2020; Bhandari and Russo, 2019; Mei et al., 2020; Zhang et al., 2020b; Agarwal et al., 2020; Zhang et al., 2020a; Li et al., 2021; Cen et al., 2021; Ding et al., 2021; Yuan et al., 2022; Moskovitz et al., 2022). Agarwal et al. (2021) established the asymptotic global convergence of policy gradient under different policy parameterizations. We extend the result of entropy regularized PG with stochastic gradient (Ding et al., 2021) to the contextual MDP setting. In particular, our contextual MDP setting reduces the exponential state space dependency w.r.t. the iteration number and per iteration sample complexity suggested by Ding et al. (2021) to a polynomial dependency. While there exists convergence analyses on other variants of PG that produce an iteration number that does not suffer from an exponential state space dependency (Agarwal et al., 2021; Mei et al., 2020), they assume access to the *exact* gradient during each update of PG. In contrast, we assume a stochastic estimation of the gradient.

[1] See Definition 4.2 of Section 4.2.

Exploration bonuses. A number of prior works have shown that one can achieve a polynomial complexity of learning an optimal policy with effective exploration methods (Azar et al., 2017; Jin et al., 2018; Du et al., 2019; Misra et al., 2020; Agarwal et al., 2020; Zhang et al., 2020d). The computational efficiency suggested by our work is different from some of the aforementioned prior methods that rely on adding exploration bonuses (Azar et al., 2017; Jin et al., 2018; Agarwal et al., 2020; Zhang et al., 2020d), as we assume access to a "good" curriculum which ensures the optimal policy of the next context is not too different from the optimal policy of the current context while eschewing exploration bonuses entirely.

Contextual MDPs. Contextual MDPs (or MDPs with side information) have been studied extensively in the theoretical RL literature (Abbasi-Yadkori and Neu, 2014; Hallak et al., 2015; Dann et al., 2019; Jiang et al., 2017; Modi et al., 2018; Sun et al., 2019; Dann et al., 2019; Modi et al., 2020). We analyze the iteration complexity and sample complexity of (stochastic) policy gradient methods, which is distinct from these prior works that mainly focus on regret bounds (Abbasi-Yadkori and Neu, 2014; Hallak et al., 2015; Dann et al., 2019) and PAC bounds (Jiang et al., 2017; Modi et al., 2018; Sun et al., 2019; Dann et al., 2019; Modi et al., 2020). Several works assumed linear transition kernel and reward model (or generalized linear model (Abbasi-Yadkori and Neu, 2014)) with respect to the context (Abbasi-Yadkori and Neu, 2014; Modi et al., 2018; Dann et al., 2019; Modi et al., 2020; Belogolovsky et al., 2021). These assumptions share similarity to our assumptions — we have a weaker Lipschitz continuity assumption with respect to the context space (since linear implies Lipschitz) on the reward function and a stronger shared transition kernel assumption.

Curriculum learning in reinforcement learning. Curriculum learning is a powerful idea that has been widely used in RL (Florensa et al., 2017; Kim and Choi, 2018; Omidshafiei et al., 2019; Ivanovic et al., 2019; Akkaya et al., 2019; Portelas et al., 2020; Bassich et al., 2020; Fang et al., 2020; Klink et al., 2020; Dennis et al., 2020; Parker-Holder et al., 2022; Liu et al., 2022) (also see (Narvekar et al., 2020) for a detailed survey). Although curricula formed by well-designed reward functions (Vinyals et al., 2019; OpenAI, 2018; Berner et al., 2019; Ye et al., 2020; Zhai et al., 2022) are usually sufficient given enough domain knowledge, tackling problems with limited domain knowledge requires a more general approach where a suitable curriculum is automatically formed from a task space. In the goal-conditioned reinforcement learning literature, this corresponds to automatic goal proposal mechanisms (Florensa et al., 2018; Warde-Farley et al., 2018; Sukhbaatar et al., 2018; Ren et al., 2019; Ecoffet et al., 2019; Hartikainen et al., 2019; Pitis et al., 2020; Zhang et al., 2020c; OpenAI et al., 2021; Zhang et al., 2021). The practical instantiation of this work is also similar to (Bassich et al., 2020; Liu et al., 2022), where a curriculum is adopted for learning a progression of a set of tasks. Klink et al. (2022) also analyzed the theoretical benefits of curriculum learning in RL, but is primarily concerned with the problem of representations for value functions when utilizing approximate value iteration methods for curriculum learning. This is accomplished by using boosting to increase the effective capacity of the value function estimator. In contrast, our method does not make any prescription in regard to the representation, but is aimed at studying sample complexity and exploration, showing that "rolling in" with the previous policy and then collecting data with the new policy leads to good sample complexity. In principle, we could even imagine in future work combining the representation analysis in Klink et al. (2022) with the discussion of state coverage in our analysis.

Learning conditional policies in multi-task RL. Multi-task RL (Tanaka and Yamamura, 2003) approaches usually learn a task-conditioned policy that is shared across different tasks (Rusu et al., 2015; Rajeswaran et al., 2016; Andreas et al., 2017; Finn et al., 2017; D'Eramo et al., 2020; Yu et al., 2020a; Ghosh et al., 2021; Kalashnikov et al., 2021; Agarwal et al., 2022). Compared to learning each task independently, joint training enjoys the sample efficiency benefits from sharing the learned experience across different tasks as long as the policies generalize well across tasks. To encourage generalization, it is often desirable to condition policies on low dimensional feature representations that are shared across different tasks instead (e.g., using variational auto-encoders (Nair et al., 2018; Pong et al., 2019; Nair et al., 2020) or variational information bottleneck (Goyal et al., 2019; 2020; Mendonca et al., 2021)). The idea of learning contextual policies has also been discussed in classical adaptive control literature (Sastry et al., 1990; Tao, 2003; Landau et al., 2011; Åström and Wittenmark, 2013; Goodwin and Sin, 2014). Different from these prior works which have been mostly focusing on learning policies that can generalize across different tasks, our work focuses on how the near-optimal policy from a learned task could be used to help the learning of a similar task.

3. Preliminaries

We consider the contextual MDP setting, where a contextual MDP, $\mathcal{M}_{\mathcal{W}} = (\mathcal{W}, \mathcal{S}, \mathcal{A}, \boldsymbol{P}, r_\omega, \gamma, \rho)$, consists of a context space $\mathcal{W}$, a state space $\mathcal{S}$, an action space $\mathcal{A}$, a transition dynamic function $\boldsymbol{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ (where $\mathcal{P}(X)$ denotes the set of all probability distributions over set X), a context-conditioned reward function $r : \mathcal{W} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$, a discount factor $\gamma \in (0, 1]$, and an

initial state distribution of interest ρ. For convenience, we use $S = |\mathcal{S}|, A = |\mathcal{A}|$ to denote the number of states and actions. While some contextual MDP formulations (Hallak et al., 2015) have context-conditioned transition dynamics and reward functions, we consider the setting where only the reward function can change across contexts. We denote r_ω as the reward function conditioned on a fixed $\omega \in \mathcal{W}$ and $\mathcal{M}_\omega = (\mathcal{S}, \mathcal{A}, \boldsymbol{P}, r_\omega, \gamma, \rho)$ as the MDP induced by such fixed reward function. We use $\pi(a|s) : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ to denote a policy and we adopt the softmax parameterization: $\pi_\theta(a|s) = \frac{\exp[\theta(s,a)]}{\sum_{a'} \exp[\theta(s,a')]}$, where $\theta : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$. We use $d_\rho^\pi(s) := (1-\gamma)\sum_{t=0}^\infty \gamma^t \mathbb{P}^\pi(s_t = s | s_0 \sim \rho)$ to denote the discounted state visitation distribution and $V_\omega^\pi := \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t r_\omega(s_t, a_t)\right] + \alpha \mathbb{H}(\rho, \pi)$ to denote the entropy regularized discounted return on $\mathcal{M}_\omega$, where $\mathbb{H}(\rho, \pi) := \mathbb{E}_{s_0 \sim \rho, a_h \sim \pi(\cdot|s_h)}\left[\sum_{h=0}^\infty -\gamma^h \log \pi(a_h|s_h)\right]$ is the discounted entropy term. We use $\pi_\omega^\star := \arg\max_\pi V_\omega^\pi$ to denote an optimal policy that maximizes the discounted return under $\mathcal{M}_\omega$. We assume all the contextual reward functions are bounded within $[0,1]$: $r_\omega(s,a) \in [0,1], \ \forall \omega \in \mathcal{W}, \forall (s,a) \in \mathcal{S} \times \mathcal{A}$. Similarly to previous analysis (Agarwal et al., 2021; Mei et al., 2020; Ding et al., 2021), we assume the initial distribution ρ for PG or stochastic PG satisfies $\rho(s) > 0, \forall s \in \mathcal{S}$. Supposing we are given a curriculum $\{\omega_k\}_{k=0}^K$, where the last context ω_K defines $\mathcal{M}_{\omega_K}$ the MDP of interest, our goal is to show that sequentially solving $\pi_{\omega_k}^\star$ for $k = 0, 1, \ldots, K$, enjoys better computational complexity and sample complexity than learning $\pi_{\omega_K}^\star$ problem $\mathcal{M}_{\omega_K}$ from scratch.

3.1. Assumptions

As we will show in Section 4, if there is a curriculum $\{\omega_k\}_{k=0}^K$ where the optimal policies $\pi_{\omega_k}^\star, \pi_{\omega_{k+1}}^\star$ with respect to two consecutive contexts ω_k, ω_{k+1} are close enough to each other in terms of their state visitation distributions, using an ε-optimal policy of ω_k as an initialization allows us to directly start from the near-optimal regime of ω_{k+1}, hence only requiring polynomial complexity to learn $\pi_{\omega_{k+1}}^\star$. We describe our curriculum assumptions as follows.

Assumption 3.1 (Lipschitz reward in the context space)**.** *The reward function is Lipschitz continuous with respect to the context:* $\max_{s,a} |r_\omega(s,a) - r_{\omega'}(s,a)| \leq L_r \|\omega - \omega'\|_2, \forall \omega, \omega' \in \mathcal{W}$.

Intuitively, Assumption 3.1 defines the similarity between two tasks via a Lipschitz continuity in the context space. Similar Lipschitz assumptions also appears in (Abbasi-Yadkori and Neu, 2014; Modi et al., 2018; Dann et al., 2019; Modi et al., 2020; Belogolovsky et al., 2021).

Assumption 3.2 (Similarity of Two Contexts)**.** *The curriculum* $\{\omega_k\}_{k=0}^K$ *satisfies* $\max_{0 \leq k \leq K-1} \|\omega_{k+1} - \omega_k\|_2 \leq O\left(S^{-2}\right)$*, and we have access to a near-optimal initialization* $\theta_0^{(0)}$ *for learning* $\pi_{\omega_0}^\star$ *(formally defined in Section 4.2).*

At first glance, the near-optimal initialization $\theta_0^{(0)}$ for the first task ω_0 in the curriculum (suggested by Assumption 3.2) may seem like a strong assumption, but in many practical settings, it could be quite easy to obtain. For example, if the tasks correspond to reaching different goals, the curriculum might start with a goal right on top of the starting state, and therefore trivially easy to learn. As another example, if the task is a locomotion task and ω contexts correspond to target velocities, ω_0 might correspond to a velocity of zero, corresponding to standing still.

Assumption 3.1 and Assumption 3.2 together quantify the maximum difference between two consecutive tasks $\mathcal{M}_{\omega_{k-1}}, \mathcal{M}_{\omega_k}$, in terms of the maximum difference between their reward function, which plays a crucial role in reducing the exponential complexity to a polynomial one. We will briefly discuss intuition in the next section.

3.2. Prior Analysis on PG with stochastic gradient

Ding et al. (2021) proposed a two-phased PG convergence analysis framework with a stochastic gradient. In particular, the author demonstrates that with high probability, stochastic PG with arbitrary initialization achieves an ε-optimal policy can be achieved with iteration numbers of T_1, T_2 and per iteration sample complexities of B_1, B_2 in two separate phases where $T_1 = \widetilde{\Omega}\left(S^{2S^3}\right), T_2 = \widetilde{\Omega}\left(S^{3/2}\right)$ ($\widetilde{\Omega}(\cdot)$ suppresses the $\log S$ and terms that do not contain S) and $B_1 = \widetilde{\Omega}\left(S^{2S^3}\right), B_2 = \widetilde{\Omega}\left(S^5\right)$, respectively, and PG enters phase 2 only when the updating policy becomes ε_0-optimal, where ε_0 is a term depending on S (formally defined by (19) in Appendix A.3). For completeness, we restate the main theorem of Ding et al. (2021) in Theorem A.2, provide the details of such dependencies on S in Corollary A.3, and describe the two-phase procedure in Algorithm 4. The main implication of the two-phase results is that, when applying SPG to learn an optimal policy from an arbitrary initialization, we suffer from exponential complexities, unless the initialization is ε_0-optimal. We will now discuss how Assumption 3.1 and Assumption 3.2 enable an ε_0-optimal initialization for every ω_k, reducing the exponential complexities to polynomial complexities.

4. Theoretical Analysis

In this section, we first introduce ROLLIN, a simple algorithm that accelerates policy learning under the contextual MDP setup by bootstrapping new context learning with a better initial distribution (Algorithm 1). Then, we provide the total complexity analysis of applying ROLLIN to stochastic PG for achieving an ε-optimal policy. Finally, we validate our theoretical results on a tabular MDP.

4.1. ROLLIN

The theoretical version of ROLLIN is provided in Algorithm 1. The intuition behind ROLLIN is that when two consecutive contexts in the curriculum $\{\omega_k\}_{k=1}^K$ are close, their optimal parameters $\theta^\star_{\omega_{k-1}}, \theta^\star_{\omega_k}$ should be close to each other. Let $\theta_t^{(k)}$ denote the parameters at the t^{th} iteration of stochastic PG for learning $\theta^\star_{\omega_k}$. If we initialize $\theta_0^{(k)}$ as the optimal parameter of the previous context $\theta^\star_{\omega_{k-1}}$ (line 5 in Algorithm 1), and set the initial distribution μ_k as a mixture of the optimal state visitation distribution of the previous context $d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}$ and the original distribution of interest ρ with $\beta \in (0,1)$ (line 6 in Algorithm 1),

$$\mu_k = \beta d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}} + (1-\beta)\rho, \tag{1}$$

then we can show that stochastic PG enjoys a faster convergence rate. This is because setting $\theta_0^{(k)} = \theta^\star_{k-1}$ ensures a near-optimal initialization for learning ω_k, and setting μ_k as the mixture further improves the rate of convergence by decreasing the density mismatch ratio $\left\| d_{\mu_k}^{\pi^\star_{\omega_k}} / \mu_k \right\|_\infty$ (a term with that influences the convergence rate).

4.2. Main Results

We now discuss how to use a sequence of contexts to learn the target context ω_K with provable efficiency given a near-optimal policy $\pi_{\theta_0^{(0)}}$ of the initial context ω_0, without incurring an exponential dependency on S (as mentioned in Section 3.2). Our polynomial complexity comes as a result of enforcing an ε_0-optimal initialization (ε_0 is the same as Section 3.2 and (19)) for running stochastic PG (line 6 of Algorithm 1). Hence, stochastic PG directly enters phase 2, with a polynomial dependency on S.

Our main results consist of two parts. We first show that when two consecutive contexts ω_{k-1}, ω_k are close enough to each other, using ROLLIN for learning $\theta^\star_k$ with initialization $\theta_0^{(k)} = \theta^\star_{\omega_{k-1}}$ and applying an initial distribution $\mu_k = \beta d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}} + (1-\beta)\rho$ improves the convergence rate. Specifically, the iteration number and complexity for learning $\theta^\star_{\omega_k}$ from $\theta^\star_{\omega_{k-1}}$ are stated as follows:

Theorem 4.1 (Complexity of Learning the Next Context)**.** *Consider the context-based stochastic softmax policy gradient (line 7 of Algorithm 1), and suppose Assumption 3.1 and Assumption 3.2 hold, then the iteration number of obtaining an ε-optimal policy for ω_k from $\theta^\star_{\omega_{k-1}}$ is $\widetilde{\Omega}(S)$ and the per iteration sample complexity is $\widetilde{\Omega}\left(\frac{L_r}{\alpha(1-\beta)} S^3\right)$.*

Theorem 4.1 shows that when ω_{k-1}, ω_k are close enough, ROLLIN reduces the minimum required iteration and sample complexity from an exponential dependency of $\widetilde{\Omega}(S^{2S^3})$ to an iteration number of $\widetilde{\Omega}(S)$ and per iteration sample complexity of $\widetilde{\Omega}(S^3)$. It is worth noting that the theorem above only addresses the iteration number and sample complexity for learning $\theta^\star_{\omega_k}$ from $\theta^\star_{\omega_{k-1}}$. Theorem 4.3 provides the total complexity for learning $\theta^\star_{\omega_K}$ from $\theta_0^{(0)}$ via recursively applying the results in Theorem 4.1. Before introducing Theorem 4.3, we first provide a criterion for the desired initialization of $\theta_0^{(0)}$.

Algorithm 1 Provably Efficient Learning via ROLLIN

1: **Input:** ρ, $\{\omega_k\}_{k=0}^K$, $\mathcal{M}_{\mathcal{W}}$, $\beta \in (0,1)$, $\theta_0^{(0)}$.
2: Initialize $\mu_0 = \rho$.
3: Run stochastic PG (Algorithm 4) with initialization $\theta_0^{(0)}$, μ_0, $\mathcal{M}_{\omega_0}$ and obtain $\theta^\star_{\omega_0}$.
4: **for** $k = 1, \ldots, K$ **do**
5: Set $\theta_1^{(k)} = \theta^\star_{\omega_{k-1}}$. ▷ $\pi_{\theta^\star_{\omega_{k-1}}}$ is optimal for ω_{k-1}.
6: Set $\mu_k = \beta d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}} + (1-\beta)\rho$.
7: Run stochastic PG (Algorithm 4) with initialization $\theta_0^{(k)}$, μ_k, $\mathcal{M}_{\omega_k}$ and obtain $\theta^\star_{\omega_k}$.
8: **end for**
9: **Output:** $\theta^\star_{\omega_K}$

Definition 4.2 (Near-optimal Initialization)**.** *We say θ_0 is a near-optimal initialization for learning $\theta^\star_\omega$ if θ_0 satisfies $V_\omega^{\pi^\star_\omega}(\rho) - V_\omega^{\pi_{\theta_0}}(\rho) < \varepsilon_0$ and $\left\| \rho - d_\rho^{\pi^\star_{\omega_0}} \right\|_1 \le \|\omega_1 - \omega_0\|_2$.*

Note that in the above definition, $\pi^\star_{\omega_k}$ represents the optimal policy of ω_k, and $V^\pi_{\omega_k}$ represents value function of context ω_k under policy π. We now introduce the results for the overall complexity:

Theorem 4.3 (Main Results: Total Complexity of ROLLIN)**.** *Suppose Assumption 3.1 and Assumption 3.2 hold, and $\theta_0^{(0)}$ is a near-optimal initialization, then the total number of iterations of learning $\pi^\star_{\omega_K}$ using Algorithm 1 is $\Omega(KS)$ and the per iteration sample complexity is $\widetilde{\Omega}\left(S^3\right)$, with high probability.*

A direct implication of Theorem 4.3 is that, with a curriculum $\{\omega_k\}_{k=0}^K$ satisfying Assumption 3.1 and Assumption 3.2, one can reduce the daunting exponential dependency on S caused by poor initialization to a polynomial dependency. Admittedly the state space S itself is still large in practice, but reducing the state space S itself requires extra assumptions on $\mathcal{S}$, which is beyond the scope of this work. We now provide a sketch proof of Theorem 4.1 and Theorem 4.3 in the next subsection and leave all the details to Appendix A.4 and Appendix A.5 respectively.

4.3. Proof Sketch

Sketch proof of Theorem 4.1. The key insight for proving Theorem 4.1 is to show that in MDP $\mathcal{M}_{\omega_k}$, the value

	Entropy Coeff.	Baseline	ROLLIN
Hard	$\alpha = 0.01$	0.500 ± 0.000	0.562 ± 0.000
	$\alpha = 0.001$	0.856 ± 0.006	1.000 ± 0.000
Easy	$\alpha = 0.01$	0.944 ± 0.003	1.000 ± 0.000
	$\alpha = 0.001$	1.000 ± 0.000	1.000 ± 0.000

	Entropy Coeff.	Baseline	ROLLIN
Hard	$\alpha = 0.01$	0.000 ± 0.000	0.000 ± 0.000
	$\alpha = 0.001$	0.424 ± 0.023	1.067 ± 0.000
Easy	$\alpha = 0.01$	4.093 ± 0.224	7.374 ± 0.216
	$\alpha = 0.001$	10.536 ± 0.002	10.620 ± 0.002

Table 1: **Curriculum progress κ (left) and final return V^π (right) on the four-room navigation with stochastic PG.** Both metrics are reported at the $50,000^{\text{th}}$ gradient step. We use a mixing ratio of $\beta = 0.75$. Across two entropy coefficients and two reward settings (easy and hard), stochastic PG with ROLLIN consistently achieves better curriculum progress and final return. The standard error is computed over 10 random seeds.

function with respect to $\pi^\star_{\omega_k}, \pi^\star_{\omega_{k-1}}$ can be bounded by the ℓ^2-norm between ω_k and ω_{k-1}. In particular, we prove such a relation in Lemma A.5:

$$V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) \le \frac{2L_r \left\|\omega_k - \omega_{k-1}\right\|_2}{(1-\gamma)^2}. \quad (2)$$

By setting $\theta_0^{(k)} = \theta^\star_{\omega_{k-1}}$, Equation (2) directly implies $V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V^{\theta_0^{(k)}}(\rho) \le \frac{2L_r\|\omega_k - \omega_{k-1}\|_2}{(1-\gamma)^2}$. As suggested by Ding et al. (2021) stochastic PG can directly start from stage 2 with polynomial complexity of $T_2 = \widetilde{\Omega}(S), B_2 = \widetilde{\Omega}(S^5)$, if $V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V^{\theta_0^{(k)}}(\rho) \le \varepsilon_0$, where ε_0 (formally defined in Equation (19) in Appendix A.3) is a constant satisfying $\varepsilon_0 = O(S^{-2})$. Hence, by enforcing two consecutive contexts to be close enough $\left\|\omega_k - \omega_{k-1}\right\|_2 \le O(S^{-2})$, we can directly start from a near-optimal initialization with polynomial complexity with respect to S. It is worth highlighting that the per iteration sample complexity B_2 shown by Ding et al. (2021) scales as $\widetilde{\Omega}(S^5)$, while our result in Theorem 4.1 only requires a smaller sample complexity of $\widetilde{\Omega}(S^3)$. Such an improvement in the sample complexity comes from line 6 of ROLLIN: $\mu_k = \beta d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}} + (1-\beta)\rho$. Intuitively, setting μ_k as $\beta d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}} + (1-\beta)\rho$ allows us to provide an upper bound on the density mismatch ratio:

$$\left\| d_{\mu_k}^{\pi^\star_{\mu_k}} / \mu_k \right\|_\infty \le \widetilde{O}\left(\frac{L_r}{\alpha(1-\beta)} \Delta_\omega^k S \right), \quad (3)$$

where $\Delta_\omega^k = \max_{1 \le i \le k} \left\|\omega_i - \omega_{i-1}\right\|_2$. Since the sample complexity B_2 (provided in Corollary A.3) contains one multiplier of $\left\| d_{\mu_k}^{\pi^\star_{\mu_k}} / \mu_k \right\|_\infty$, setting $\Delta_\omega^k = O(S^{-2})$ immediately reduces the complexity by an order of S^2. The proof of the upper bound of the density mismatch ratio (Equation (3)) is provided in Lemma A.1.

Sketch proof of Theorem 4.3. We obtain Theorem 4.3 by recursively applying Theorem 4.1. More precisely, we use induction to show that, if we initialize the parameters of the policy as $\theta_0^{(k)} = \theta^\star_{\omega_{k-1}}$, when $t = \widetilde{\Omega}(S), \forall k \in [K]$, we have $V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi_{\theta_t^{(k-1)}}}(\rho) < \varepsilon_0$. Hence, for any context $\omega_k, k \in [K]$, initializing $\theta_0^{(k)} = \theta_t^{(k-1)}$ from learning

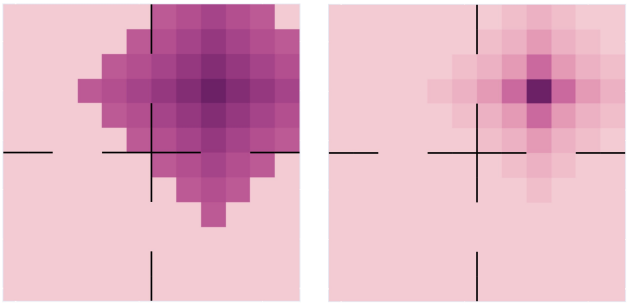


Figure 2: **Visualization of the two reward functions for the four-room navigation environment.** Left: easy; Right: hard. Darker color indicates a higher reward when the agent reaches the state (with the goal state the darkest). The agent receives 0 reward when it is too far from the goal location (5 steps for the easy reward function and 4 steps for the hard reward function). The black line indicates walls in the environment where the agent cannot pass through. The reward function visualization in this figure is for the last context. The reward function for other contexts (other goals) is the same but with the reward function shifted according to the goal state.

$\pi^\star_{\omega_{k-1}}$ via stochastic PG after $t = \Omega(S)$ iteration, $\theta_0^{(k)}$ will directly start from the efficient phase 2 with polynomial complexity. Hence, the total iteration number for learning the $\theta_K^\star$ is $\Omega(KS)$, and the per iteration sample complexity remains the same as Theorem 4.1 ($\widetilde{\Omega}\left(S^3\right)$).

4.4. Numerical Experiments: Four-room Navigation

To provide empirical support for the validity of our theoretical results, we follow the exact setting that is used in our theoretical analysis and implement ROLLIN with stochastic PG and the softmax policy parameterization on a tabular contextual MDP. It is worth highlighting that this is distinct from the practical implementation of ROLLIN in Section 5 – the focus there is to apply the idea of ROLLIN to design a sample-efficient deep RL algorithm for a more practical setting (e.g., continuous state and action space), whereas the focus here is solely to validate our theory in the theoretical setting. The contextual MDP is a grid world consisting of 12×12 grid cells where each cell corresponds to a state in the MDP. The agent always starts from the bottom-left corner of the grid world and navigates around to collect rewards. In particular, the agent receives a positive reward

when it is close to a pre-defined goal cell/state which is defined by the context. We use a curriculum consisting of 17 contexts/goals in sequence, $\{\omega_k\}_{k=0}^{16}$, which form a path from the initial state of the agent to a far away goal location, and we switch from the current context to the next one in the curriculum whenever the current goal location is reached with more than 50% probability. We experiment with two different reward functions (visualized in Figure 2). Table 1 summarizes the results of our experiments. ROLLIN is able to consistently improve upon the vanilla stochastic PG baseline (across two different entropy coefficients and two reward functions with varying difficulties) in terms of the curriculum progress and the final return. This verifies that ROLLIN can indeed improve the sample complexity of stochastic PG in a curriculum learning setting, validating our theory. See more implementation details of the numerical experiments in Appendix F.

5. Practical Implementation of ROLLIN

We have shown empirical evidence through numerical experiments that ROLLIN can lead to sample complexity reduction under our theoretical setting (tabular MDP with discrete action space and state space, softmax parameterization and entropy regularized objective). Now, we introduce a practical implementation of ROLLIN using Soft-Actor-Critic (Haarnoja et al., 2018) such that ROLLIN can be applied to more practical problems with continuous action space and state space. SAC can be seen as a variant of entropy-regularized stochastic PG with the addition of the critics to reduce gradient variance. Recall that in the theoretical analysis, we learn a separate policy for each context that can start from the near-optimal state distribution of the previous context to achieve a good return under the current context. However, in practice, we usually would want to have a policy that can directly start from the initial distribution ρ to obtain a good return for the final context ω_K. In order to learn such a policy, we propose to have two context-conditioned RL agents training in parallel, where the first agent π_{main} is the main agent that eventually will learn to achieve a good return from ρ, and the second agent π_{exp} is an exploration agent that learns to achieve a good return under the current context from the near-optimal state density of the previous context. Another purpose of the exploration agent (as the name suggests) is to provide a better exploration experience for the main agent to learn the current context better. This is made convenient by using an off-policy RL agent where the main agent can learn from the data collected by the exploration agent.

Specifically, for each episode, there is a probability of β where we run the main agent conditioned on the previous context for a random number of steps until we switch to the exploration agent to collect experience for the current context until the episode ends. Otherwise, we directly run the main agent for the entire episode. Both agents are trained to maximize the return under the current context. Whenever the average return of the last 10 episodes exceeds a performance threshold R, we immediately switch to the next context and re-initialize the exploration agent and its replay buffer. A high-level description is available in Algorithm 2 (a more detailed version in Algorithm 8).

Algorithm 2 Practical Implementation of ROLLIN

Input: $\{\omega_k\}_{k=0}^{K}$: input curriculum, R: near-optimal threshold, β: roll-in ratio, H: horizon, γ: discount factor.
Initialize $\mathcal{D} \leftarrow \emptyset, \mathcal{D}_{\text{exp}} \leftarrow \emptyset, k \leftarrow 0$, and two SAC agents π_{main} and π_{exp}.
for each episode **do**
  if average return of the last 10 episodes under context ω_k is greater than R **then**
    $k \leftarrow k+1$, $\mathcal{D}_{\text{exp}} \leftarrow \emptyset$, and re-initialize the exploration agent π_{exp}
  end if
  if $k > 0$ and with probability of β **then**
    $h \sim \text{Geom}(1-\gamma)$ (truncated at H)
    run $\pi_{\text{main}}(a|s, \omega_{k-1})$ from the initial state for h steps and switch to $\pi_{\text{exp}}(a|s, \omega_k)$ until the episode ends to obtain trajectory $\tau_{0:H} = \{s_0, a_0, r_0, s_1, a_1, \cdots, s_H\}$.
    record $\tau_{0:H}$ in $\mathcal{D}$, and $\tau_{h:H}$ in $\mathcal{D}_{\text{exp}}$.
  else
    run $\pi_{\text{main}}(a|s, \omega_k)$ to obtain trajectory $\tau_{0:H}$ and record $\tau_{0:H}$ in $\mathcal{D}$.
  end if
  at each environment step in the episode, update $\pi_{\text{main}}(\cdot|\cdot, \omega_k)$ using $\mathcal{D}$ and $\pi_{\text{exp}}(\cdot|\cdot, \omega_k)$ using $\mathcal{D}_{\text{exp}}$.
end for
Output: π_{main}

6. Experimental Results

While the focus of our work is on developing a provably efficient approach to curriculum learning, we also conduct an experimental evaluation of our practical implementation of ROLLIN with soft actor-critic (SAC) (Haarnoja et al., 2018) as the RL algorithm on several continuous control tasks including a goal reaching task and four non-goal reaching tasks with oracle curricula.

6.1. Goal Reaching with an Oracle Curriculum

We adopt the `antmaze-umaze` environment (Fu et al., 2020) for evaluating the performance of ROLLIN in goal-reaching tasks. We use a hand-crafted path of contexts, where each specifies a goal location (as shown in Ap-

Setting	Method	w/o Geometric Sampling		w/ Geometric Sampling	
		$\Delta = 1/24$	$\Delta = 1/12$	$\Delta = 1/24$	$\Delta = 1/12$
Vanilla	Baseline	0.40 ± 0.02	0.36 ± 0.00	0.82 ± 0.08	0.38 ± 0.03
	ROLLIN	0.49 ± 0.04	0.44 ± 0.01	0.92 ± 0.02	0.55 ± 0.04
Relabeling	Baseline	0.89 ± 0.03	0.66 ± 0.04	0.76 ± 0.02	0.72 ± 0.03
	ROLLIN	0.91 ± 0.03	0.74 ± 0.01	0.78 ± 0.01	0.73 ± 0.00
Go-Explore Noise = 0.1	Baseline	0.37 ± 0.02	0.38 ± 0.01	0.82 ± 0.07	0.42 ± 0.03
	ROLLIN	0.52 ± 0.07	0.38 ± 0.01	0.95 ± 0.02	0.43 ± 0.02

Table 2: Learning progress κ at 3 million environment steps with varying curriculum step size Δ of different settings of goal reaching in `antmaze-umaze`. We pick $\beta = 0.1$ for all experiments using ROLLIN, the results of using other βs, Δs, and exploration noises can be found in Table 8, Table 9, and Table 10 in Appendix G.1. The standard error is computed over 8 random seeds.

pendix E.1, Figure 3). We consider a path of contexts $\omega(\kappa)$ parameterized by $\kappa \in [0, 1]$ where $\omega(0) = \omega_0$ and $\omega(1) = \omega_K$, and step through the contexts along the path with a fixed step size Δ. See Appendix E.1 for more implementation details.

We combine ROLLIN with a variety of prior methods, and we evaluate the following conditions: (1) standard goal reaching; (2) goal reaching with goal relabeling (Andrychowicz et al., 2017); (3) goal reaching with an exploration phase that is similar to Go-Explore (Ecoffet et al., 2019). For goal relabeling, we adopt a similar relabeling technique as Pitis et al. (2020), where each mini-batch contains $1/3$ original transitions, $1/3$ transitions with future state relabeling, and $1/3$ transitions with next state relabeling. We implemented the Go-Explore method by adding an additional standard Gaussian exploration noise in the action to the agent for learning the next goal $\omega(k+1)$, once it reaches the current goal $\omega(k)$. We empirically observed that sampling the replay buffer from a geometric distribution with $p = 10^{-5}$ (more recent transitions are sampled more frequently) improves the overall performance. Hence, in all future experiments, we compare the performance of ROLLIN with classic uniform sampling and the new geometric sampling. We compare the learning speed of ROLLIN with parameter $\beta = 0.1$ on three different step sizes $\Delta = \frac{1}{24}, \frac{1}{18}, \frac{1}{12}$ in Table 2.

Main comparisons. We first provide an overview experiments that compares ROLLIN with a fixed $\beta = 0.1$ on different step sizes Δ in different settings. In each case, we compare the prior method (vanilla, relabeled, or Go-Explore) with and without the addition of ROLLIN. As shown in Table 2, ROLLIN improves the largest value of κ reached by the agent in most presented settings (except Go-Explore with $\Delta = 1/12$). This result suggests that ROLLIN facilitates goal-conditioned RL with a curriculum, as we only update the learning progress κ to $\kappa + \Delta$ when the return of the current policy reaches a certain threshold R (See detailed update of κ in Algorithm 2). Note that $\beta = 0.1$ does not always produce the best result, we will provide more results comparing different βs in different settings later in this section, and we leave all the learning curves and detailed tables to Appendix G.1. Note that we do not include the results of directly learning the last context in the `antmaze-umaze` environment because the agent cannot reach the goal without the aid of a curriculum, which is corroborated by (Pitis et al., 2020).

6.2. Non-Goal Reaching Tasks

For the non-goal tasks, we consider the tasks of gradually increasing the x-velocity of a locomotion agent in the following environments: `walker2d`, `hopper`, `humanoid`, and `ant` in OpenAI gym (Brockman et al., 2016). More specifically, we set the desired speed range to be $[\lambda\kappa, \lambda(\kappa + 0.1))$, where λ is a parameter depending on the physics of the agent in different environments and we choose a fixed contextual space with ten discrete contexts: $\kappa \in \{0.1, 0.2, \ldots, 1\}$. The agent receives a higher reward when the x-velocity is within the desired speed range and a lower reward otherwise. In each environment, we increase the task difficulty with later curriculum steps (larger κ), by increasing the near-optimal threshold $R(\kappa)$. Detailed parameters of the desired speed range λ, near optimal-threshold $R(\kappa)$, and the reward functions are in Appendix E.2.

Main comparisons. We first compare ROLLIN with a fixed $\beta = 0.1$ at different environment steps: 0.5×10^6, 1×10^6. In each case, we compare the learning progress κ, averaged x-velocity, and averaged return, with and without the addition of ROLLIN. Note that for the case without ROLLIN, we still provide the curriculum to the agent for training. The results in Table 3 show that ROLLIN improves most benchmarks (See detailed update of κ in Algorithm 2). Note that $\beta = 0.1$ does not always produce the best result, and we provide more results comparing different βs in different settings later in this section, with learning curves and more

Env.	Method	Step = 0.5×10^6 κ	x-velocity	return	Step = 1.0×10^6 κ	x-velocity	return
walker	Scratch	n/a	3.07 ± 0.26	3373.1 ± 170.5	n/a	3.30 ± 0.36	4212.3 ± 151.4
	Baseline	0.83 ± 0.03	3.09 ± 0.31	3450.1 ± 307.4	0.92 ± 0.03	3.69 ± 0.27	4032.3 ± 224.3
	ROLLIN	0.79 ± 0.04	2.83 ± 0.31	3350.4 ± 184.6	0.94 ± 0.03	3.62 ± 0.26	4128.8 ± 159.6
hopper	Scratch	n/a	2.50 ± 0.13	2943.6 ± 80.3	n/a	2.55 ± 0.12	3073.2 ± 137.7
	Baseline	0.85 ± 0.02	2.42 ± 0.18	3192.5 ± 80.4	0.88 ± 0.01	2.58 ± 0.16	3386.2 ± 124.7
	ROLLIN	0.82 ± 0.03	2.26 ± 0.22	3148.6 ± 160.7	0.89 ± 0.00	2.65 ± 0.15	3421.9 ± 109.8
humanoid	Scratch	n/a	0.24 ± 0.05	2417.1 ± 188.2	n/a	0.37 ± 0.05	2763.8 ± 96.5
	Baseline	0.32 ± 0.05	0.26 ± 0.05	2910.1 ± 262.9	0.67 ± 0.03	0.39 ± 0.05	3017.2 ± 169.0
	ROLLIN	0.36 ± 0.04	0.32 ± 0.07	2939.7 ± 392.0	0.69 ± 0.06	0.46 ± 0.09	3173.6 ± 238.3
ant	Scratch	n/a	3.60 ± 0.49	2910.7 ± 354.3	n/a	4.55 ± 0.36	4277.9 ± 120.0
	Baseline	0.72 ± 0.02	3.38 ± 0.43	2976.2 ± 252.4	1.00 ± 0.00	4.29 ± 0.51	4248.5 ± 88.6
	ROLLIN	0.82 ± 0.06	3.85 ± 0.41	3593.1 ± 237.8	1.00 ± 0.00	4.66 ± 0.30	4473.0 ± 102.2

Table 3: Learning progress κ, average x-velocity, and average return at 0.5 and 1.0 million environment steps in `walker`, `hopper`, `humanoid`, and `ant`. The average x-velocity and return are estimated using the last 50k environment steps. "Scratch" shows the results of directly training the agent with the last context $\omega(1)$. "Baseline" indicates $\beta = 0$, where we provide the curriculum $\omega(\kappa)$ to the agent without using ROLLIN. We pick $\beta = 0.1$ for all experiments using ROLLIN, the results of using other βs can be found in Table 11, Table 12, and Table 13 in Appendix G.2. The standard error is computed over 8 random seeds.

detailed tables in Appendix G.2.

6.3. Experimental Summary

We empirically showed that ROLLIN improves the performance of one goal-reaching task and four non-goal tasks in different settings. Although ROLLIN introduces an extra parameter β, our experiments show reasonable improvement by simply choosing $\beta = 0.1$ or 0.2. More careful selection of β might lead to further improvements.

7. Discussion and Future Work

We presented ROLLIN, a simple algorithm that accelerates curriculum learning under the contextual MDP setup by rolling in a near-optimal policy to bootstrap the learning of new nearby contexts with provable learning efficiency benefits. Theoretically, we show that ROLLIN attains polynomial sample complexity by utilizing adjacent contexts to initialize each policy. Since the key theoretical insight of ROLLIN suggests that one can reduce the density mismatch ratio by constructing a new initial distribution, it would be interesting to see how ROLLIN can affect other variants of convergence analysis of PG (e.g., NPG (Kakade, 2001; Cen et al., 2021) or PG in a feature space (Agarwal et al., 2021; 2020)). On the empirical side, our experiments demonstrate that ROLLIN improves the empirical performance of various tasks beyond our theoretical assumptions, which reveals the potential of ROLLIN in other practical RL tasks with a curriculum. Our initial practical instantiation of the ROLLIN algorithm has a lot of room for future research. First of all, our implementation requires domain-specific knowledge of a "good" return value as it currently rely on a fixed return threshold R to determine when we are going to switch from the current context to the next context. Another promising direction is to combine our algorithm with context-based meta-RL methods such as learning to generate sub-goal/context to accelerate the learning of the current sub-goal/context. Finally, our method is not specific to the goal-conditioned settings, which opens up the possibility of applying our algorithm to more challenging domains.

8. Acknowledgements

We are thankful to Laura Smith, Dibya Ghosh, Chuer Pan, and other members of the RAIL lab for feedback and suggestions on earlier drafts. QL would like acknowledge the support of the Berkeley Fellowship. YZ would like to thank Jincheng Mei from Google and Yuhao Ding from UC Berkeley for insightful discussions on the related proof. YM would like to acknowledge the support of ONR grants N00014-20-1-2002, N00014-22-1-2102, the joint Simons Foundation-NSF DMS grant # 2031899, and Tsinghua-Berkeley Shenzhen Institute (TBSI) Research Fund. SL would like to acknowledge Air Force Office of Scientific Research AFOSR FA9550-22-1-0273. The research is supported by Savio computational cluster provided by the Berkeley Research Compute program.

References

Yasin Abbasi-Yadkori and Gergely Neu. Online learning in MDPs with side information. *arXiv preprint arXiv:1406.6812*, 2014.

Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 2019.

Alekh Agarwal, Mikael Henaff, Sham Kakade, and Wen Sun. PC-PG: Policy cover directed exploration for provable policy gradient learning. *Advances in Neural Information Processing Systems*, 33:13399–13412, 2020.

Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22(98):1–76, 2021.

Alekh Agarwal, Yuda Song, Wen Sun, Kaiwen Wang, Mengdi Wang, and Xuezhou Zhang. Provable benefits of representational transfer in reinforcement learning. *arXiv preprint arXiv:2205.14571*, 2022.

Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

Jacob Andreas, Dan Klein, and Sergey Levine. Modular multitask reinforcement learning with policy sketches. In *International Conference on Machine Learning*, pages 166–175. PMLR, 2017.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39 (1):3–20, 2020.

Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *International Conference on Machine Learning*, pages 263–272. PMLR, 2017.

Andrea Bassich, Francesco Foglino, Matteo Leonetti, and Daniel Kudenko. Curriculum learning with a progression function. *arXiv preprint arXiv:2008.00511*, 2020.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 1471–1479, 2016.

Stav Belogolovsky, Philip Korsunsky, Shie Mannor, Chen Tessler, and Tom Zahavy. Inverse reinforcement learning in contextual MDPs. *Machine Learning*, 110(9):2295–2334, 2021.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Jalaj Bhandari and Daniel Russo. Global optimality guarantees for policy gradient methods. *arXiv preprint arXiv:1906.01786*, 2019.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Shicong Cen, Chen Cheng, Yuxin Chen, Yuting Wei, and Yuejie Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *Operations Research*, 2021.

Christoph Dann, Lihong Li, Wei Wei, and Emma Brunskill. Policy certificates: Towards accountable reinforcement learning. In *International Conference on Machine Learning*, pages 1507–1516. PMLR, 2019.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. *Advances in Neural Information Processing Systems*, 33:13049–13061, 2020.

Carlo D'Eramo, Davide Tateo, Andrea Bonarini, Marcello Restelli, and Jan Peters. Sharing knowledge in multi-task deep reinforcement learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgpv2VFvr.

Yuhao Ding, Junzi Zhang, and Javad Lavaei. Beyond exact gradients: Convergence of stochastic soft-max policy gradient methods with entropy regularization. *arXiv preprint arXiv:2110.10117*, 2021.

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

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Go-Explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*, 2019.

Ben Eysenbach, Xinyang Geng, Sergey Levine, and Russ R Salakhutdinov. Rewriting history with inverse rl: Hindsight inference for policy improvement. *Advances in neural information processing systems*, 33:14783–14795, 2020a.

Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. C-learning: Learning to achieve goals via recursive classification. *arXiv preprint arXiv:2011.08909*, 2020b.

Kuan Fang, Yuke Zhu, Silvio Savarese, and Li Fei-Fei. Adaptive procedural task generation for hard-exploration problems. *arXiv preprint arXiv:2007.00350*, 2020.

Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *International Conference on Machine Learning*, pages 1467–1476. PMLR, 2018.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *Conference on robot learning*, pages 482–495. PMLR, 2017.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, pages 1515–1528. PMLR, 2018.

Justin Fu, Avi Singh, Dibya Ghosh, Larry Yang, and Sergey Levine. Variational inverse control with events: A general framework for data-driven reward definition. *arXiv preprint arXiv:1805.11686*, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Yasuhiro Fujita, Kota Uenishi, Avinash Ummadisingu, Prabhat Nagarajan, Shimpei Masuda, and Mario Ynocente Castro. Distributed reinforcement learning of targeted grasping with active vision for mobile manipulators. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9712–9719. IEEE, 2020.

Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Manon Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=rALA0Xo6yNJ.

Graham C Goodwin and Kwai Sang Sin. *Adaptive filtering prediction and control*. Courier Corporation, 2014.

Anirudh Goyal, Riashat Islam, DJ Strouse, Zafarali Ahmed, Hugo Larochelle, Matthew Botvinick, Sergey Levine, and Yoshua Bengio. Transfer and exploration via the information bottleneck. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJg8yhAqKm.

Anirudh Goyal, Yoshua Bengio, Matthew Botvinick, and Sergey Levine. The variational bandwidth bottleneck: Stochastic evaluation on an information budget. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Hye1kTVFDS.

Abhishek Gupta, Aldo Pacchiano, Yuexiang Zhai, Sham M Kakade, and Sergey Levine. Unpacking reward shaping: Understanding the benefits of reward engineering on sample complexity. *arXiv preprint arXiv:2210.09579*, 2022.

Tuomas Haarnoja. *Acquiring diverse robot skills via maximum entropy deep reinforcement learning*. University of California, Berkeley, 2018.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

Kristian Hartikainen, Xinyang Geng, Tuomas Haarnoja, and Sergey Levine. Dynamical distance learning for semi-supervised and unsupervised skill discovery. *arXiv preprint arXiv:1907.08225*, 2019.

Boris Ivanovic, James Harrison, Apoorva Sharma, Mo Chen, and Marco Pavone. BARC: Backward reachability curriculum for robotic reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 15–21. IEEE, 2019.

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

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is Q-learning provably efficient? *Advances in neural information processing systems*, 31, 2018.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *In Proc. 19th International Conference on Machine Learning*. Citeseer, 2002.

Sham M Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14, 2001.

Dmitry Kalashnikov, Jacob Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine, and Karol Hausman. MT-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.

Tae-Hoon Kim and Jonghyun Choi. Screenernet: Learning self-paced curriculum for deep neural networks. *arXiv preprint arXiv:1801.00904*, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Pascal Klink, Carlo D'Eramo, Jan R Peters, and Joni Pajarinen. Self-paced deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 9216–9227, 2020.

Pascal Klink, Carlo D'Eramo, Jan Peters, and Joni Pajarinen. Boosted curriculum reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=anbBFlX1tJ1.

Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

Ilya Kostrikov. JAXRL: Implementations of Reinforcement Learning algorithms in JAX, 10 2021. URL https://github.com/ikostrikov/jaxrl.

Tejas D Kulkarni, Ardavan Saeedi, Simanta Gautam, and Samuel J Gershman. Deep successor reinforcement learning. *arXiv preprint arXiv:1606.02396*, 2016.

Ioan Doré Landau, Rogelio Lozano, Mohammed M'Saad, and Alireza Karimi. *Adaptive control: algorithms, analysis and applications*. Springer Science & Business Media, 2011.

Gen Li, Yuting Wei, Yuejie Chi, Yuantao Gu, and Yuxin Chen. Softmax policy gradient methods can take exponential time to converge. In *Conference on Learning Theory*, pages 3107–3110. PMLR, 2021.

Xingyu Liu, Deepak Pathak, and Kris M Kitani. Revolver: Continuous evolutionary models for robot-to-robot policy transfer. *arXiv preprint arXiv:2202.05244*, 2022.

Jincheng Mei, Chenjun Xiao, Csaba Szepesvari, and Dale Schuurmans. On the global convergence rates of softmax policy gradient methods. In *International Conference on Machine Learning*, pages 6820–6829. PMLR, 2020.

Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and achieving goals via world models. *Advances in Neural Information Processing Systems*, 34, 2021.

Dipendra Misra, Mikael Henaff, Akshay Krishnamurthy, and John Langford. Kinematic state abstraction and provably efficient rich-observation reinforcement learning. In *International conference on machine learning*, pages 6961–6971. PMLR, 2020.

Aditya Modi, Nan Jiang, Satinder Singh, and Ambuj Tewari. Markov decision processes with continuous side information. In *Algorithmic Learning Theory*, pages 597–618. PMLR, 2018.

Aditya Modi, Nan Jiang, Ambuj Tewari, and Satinder Singh. Sample complexity of reinforcement learning using linearly combined model ensembles. In *International Conference on Artificial Intelligence and Statistics*, pages 2010–2020. PMLR, 2020.

Ted Moskovitz, Michael Arbel, Jack Parker-Holder, and Aldo Pacchiano. Towards an understanding of default policies in multitask policy optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 10661–10686. PMLR, 2022.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Ashvin Nair, Shikhar Bahl, Alexander Khazatsky, Vitchyr Pong, Glen Berseth, and Sergey Levine. Contextual imagined goals for self-supervised robotic learning. In *Conference on Robot Learning*, pages 530–539. PMLR, 2020.

Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems*, 31, 2018.

Suraj Nair, Eric Mitchell, Kevin Chen, Silvio Savarese, Chelsea Finn, et al. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning*, pages 1303–1315. PMLR, 2022.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *arXiv preprint arXiv:2003.04960*, 2020.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, volume 99, pages 278–287, 1999.

Shayegan Omidshafiei, Dong-Ki Kim, Miao Liu, Gerald Tesauro, Matthew Riemer, Christopher Amato, Murray Campbell, and Jonathan P How. Learning to teach in cooperative multiagent reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33 issue 01, pages 6128–6136, 2019.

OpenAI. OpenAI Five. https://blog.openai.com/openai-five/, 2018.

OpenAI OpenAI, Matthias Plappert, Raul Sampedro, Tao Xu, Ilge Akkaya, Vineet Kosaraju, Peter Welinder, Ruben D'Sa, Arthur Petron, Henrique P d O Pinto, et al. Asymmetric self-play for automatic goal discovery in robotic manipulation. *arXiv preprint arXiv:2101.04882*, 2021.

Jack Parker-Holder, Minqi Jiang, Michael Dennis, Mikayel Samvelyan, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Evolving curricula with regret-based environment design. *arXiv preprint arXiv:2203.01302*, 2022.

Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

Silviu Pitis, Harris Chan, Stephen Zhao, Bradly Stadie, and Jimmy Ba. Maximum entropy gain exploration for long horizon multi-goal reinforcement learning. In *International Conference on Machine Learning*, pages 7750–7761. PMLR, 2020.

Vitchyr H Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *Conference on Robot Learning*, pages 835–853. PMLR, 2020.

Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*, 2016.

Zhizhou Ren, Kefan Dong, Yuan Zhou, Qiang Liu, and Jian Peng. Exploration via hindsight goal generation. *Advances in Neural Information Processing Systems*, 32, 2019.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

Shankar Sastry, Marc Bodson, and James F Bartram. Adaptive control: stability, convergence, and robustness, 1990.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shamane Siriwardhana, Rivindu Weerasakera, Denys JC Matthies, and Suranga Nanayakkara. VUSFA: Variational universal successor features approximator to improve transfer DRL for target driven visual navigation. *arXiv preprint arXiv:1908.06376*, 2019.

Bradly C Stadie, Sergey Levine, and Pieter Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*, 2015.

Sainbayar Sukhbaatar, Emily Denton, Arthur Szlam, and Rob Fergus. Learning goal embeddings via self-play for hierarchical reinforcement learning. *arXiv preprint arXiv:1811.09083*, 2018.

Wen Sun, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Model-based RL in contextual

decision processes: PAC bounds and exponential improvements over model-free approaches. In *Conference on learning theory*, pages 2898–2933. PMLR, 2019.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Fumihide Tanaka and Masayuki Yamamura. Multitask reinforcement learning on the distribution of MDPs. In *Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation. Computational Intelligence in Robotics and Automation for the New Millennium (Cat. No. 03EX694)*, volume 3, pages 1108–1113. IEEE, 2003.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in neural information processing systems*, pages 2753–2762, 2017.

Gang Tao. *Adaptive control design and analysis*, volume 37. John Wiley & Sons, 2003.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354, 2019.

David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Deheng Ye, Guibin Chen, Wen Zhang, Sheng Chen, Bo Yuan, Bo Liu, Jia Chen, Zhao Liu, Fuhao Qiu, Hongsheng Yu, Yinyuting Yin, Bei Shi, Liang Wang, Tengfei Shi, Qiang Fu, Wei Yang, Lanxiao Huang, and Wei Liu. Towards playing full moba games with deep reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 621–632. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/06d5ae105ea1bea4d800bc96491876e9-Paper.pdf.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020a.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020b.

Rui Yuan, Robert M Gower, and Alessandro Lazaric. A general sample complexity analysis of vanilla policy gradient. In *International Conference on Artificial Intelligence and Statistics*, pages 3332–3380. PMLR, 2022.

Yuexiang Zhai, Christina Baek, Zhengyuan Zhou, Jiantao Jiao, and Yi Ma. Computational benefits of intermediate rewards for goal-reaching policy learning. *Journal of Artificial Intelligence Research*, 73:847–896, 2022.

Junzi Zhang, Jongho Kim, Brendan O'Donoghue, and Stephen Boyd. Sample efficient reinforcement learning with reinforce. *arXiv preprint arXiv:2010.11364*, page 97, 2020a.

Kaiqing Zhang, Alec Koppel, Hao Zhu, and Tamer Basar. Global convergence of policy gradient methods to (almost) locally optimal policies. *SIAM Journal on Control and Optimization*, 58(6):3586–3612, 2020b.

Tianjun Zhang, Benjamin Eysenbach, Ruslan Salakhutdinov, Sergey Levine, and Joseph E Gonzalez. C-planning: An automatic curriculum for learning goal-reaching tasks. *arXiv preprint arXiv:2110.12080*, 2021.

Yunzhi Zhang, Pieter Abbeel, and Lerrel Pinto. Automatic curriculum learning through value disagreement. *Advances in Neural Information Processing Systems*, 33: 7648–7659, 2020c.

Zihan Zhang, Yuan Zhou, and Xiangyang Ji. Almost optimal model-free reinforcement learningvia reference-advantage decomposition. *Advances in Neural Information Processing Systems*, 33:15198–15207, 2020d.

A. Generalization Between Different Tasks in the Context Space

A.1. Summaries of Notations and Assumptions

1. The maximum entropy RL (MaxEnt RL) objective with initial state distribution ρ in reinforcement aims at maximizing (Equation 15 & 16 of (Mei et al., 2020))

$$V^{\pi}(\rho) := \sum_{h=0}^{\infty} \gamma^h \mathbb{E}_{s_0 \sim \rho, a_h \sim \pi(a_h|s_h)} \left[r(s_h, a_h) \right] + \alpha \mathbb{H}(\rho, \pi) \tag{4}$$

and $\mathbb{H}(\pi(a_h|s_h))$ is the discounted entropy term

$$\mathbb{H}(\rho, \pi) := \mathop{\mathbb{E}}_{s_0 \sim \rho, a_h \sim \pi(\cdot|s_h)} \left[\sum_{h=0}^{\infty} -\gamma^h \log \pi(a_h|s_h) \right], \tag{5}$$

and α is the penalty term. For simplicity, we denote the optimization objective function in (4) as α-MaxEnt RL. Similar to Equation 18 & 19 of (Mei et al., 2020), we also define the advantage and Q-functions and for MaxEnt RL as

$$\begin{aligned} A^{\pi}(s, a) &:= Q^{\pi}(s, a) - \alpha \log \pi(s, a) - V^{\pi}(s), \\ Q^{\pi}(s, a) &:= r(s, a) + \gamma \sum_{s'} P(s'|s, a) V^{\pi}(s). \end{aligned} \tag{6}$$

2. We let

$$d_{s_0}^{\pi}(s) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}^{\pi}(s_t = s | s_0), \tag{7}$$

to denote the discounted state visitation of policy π starting at state s_0, and let

$$d_{\rho}^{\pi}(s) = \mathbb{E}_{s \sim \rho}[d_s^{\pi}(s)] \tag{8}$$

denote the initial state visitation distribution under **initial state distribution** ρ.

3. We assume the reward functions under all context are bounded within $[0, 1]$:

$$r_{\omega}(s, a) \in [0, 1], \ \forall \omega \in \Omega, \forall (s, a) \in \mathcal{S} \times \mathcal{A}. \tag{9}$$

4. Similar to previous analysis in (Agarwal et al., 2021; Mei et al., 2020; Ding et al., 2021), we assume the initial distribution μ for PG/stochastic PG satisfies $\rho(s) > 0, \forall s \in \mathcal{S}$.

A.2. Main Results: Mismatch Coefficient Upper Bound

Lemma A.1 (Density Mismatch Ratio via ROLLIN). *Assuming $\rho = \mathrm{Unif}(\mathcal{S})$, and $\mu_k = \beta d_{\mu_{k-1}}^{\pi^{\star}_{\omega_{k-1}}} + (1 - \beta)\rho$ (using (1) from ROLLIN), the density mismatch ratio $\left\| d_{\mu_k}^{\pi^{\star}_{\omega_k}} / \mu_k \right\|_{\infty}$ satisfies*

$$\left\| \frac{d_{\mu_k}^{\pi^{\star}_{\omega_k}}}{\mu_k} \right\|_{\infty} \leq \widetilde{O}\left(\frac{L_r}{\alpha(1 - \beta)} \Delta_{\omega}^k S \right), \tag{10}$$

where $\Delta_{\omega}^k = \max_{1 \leq i \leq k} \|\omega_i - \omega_{i-1}\|_2$.

Proof. By (1) from ROLLIN, we have

$$\begin{aligned}\left\|\frac{d_{\mu_k}^{\pi^\star_{\omega_k}}}{\mu_k}\right\|_\infty &= \left\|\frac{d_{\mu_k}^{\pi^\star_{\omega_k}} - d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}} + d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}}{\mu_k}\right\|_\infty \\ &\overset{(i)}{\leq} \frac{\left\|d_{\mu_k}^{\pi^\star_{\omega_k}} - d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}\right\|_1}{\min \mu_k} + \left\|\frac{d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}}{\beta d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}} + (1-\beta)\rho}\right\|_\infty \\ &\overset{(ii)}{\leq} \frac{\left\|d_{\mu_k}^{\pi^\star_{\omega_k}} - d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}\right\|_1}{\min \mu_k} + \frac{1}{\beta}\end{aligned} \tag{11}$$

where inequality (i) holds because of (1), and inequality (ii) holds because $\rho(s) \geq 0, \forall s \in \mathcal{S}$. Now it remains to bound $\left\|d_{\mu_{k+1}}^{\pi^\star_{\omega_{k+1}}} - d_{\mu_k}^{\pi^\star_{\omega_k}}\right\|_1$ using the difference $\|\omega_{k+1} - \omega_k\|_2$. Let $\mathbb{P}_h^k = \mathbb{P}_h^{\pi^\star_{\omega_k}}(s'|s_0 \sim \mu_k)$ denote the state visitation distribution resulting from $\pi^\star_{\omega_k}$ probability starting at μ_k, then we have

$$\begin{aligned}&\mathbb{P}_h^k(s') - \mathbb{P}_h^{k-1}(s') = \sum_{s,a}\left(\mathbb{P}_{h-1}^k(s)\pi^\star_{\omega_k}(a|s) - \mathbb{P}_{h-1}^{k-1}(s)\pi^\star_{\omega_{k-1}}(a|s)\right)P(s'|s,a) \\ &= \sum_{s,a}\left(\mathbb{P}_{h-1}^k(s)\pi^\star_{\omega_k}(a|s) - \mathbb{P}_{h-1}^k(s)\pi^\star_{\omega_{k-1}}(a|s) + \mathbb{P}_{h-1}^{k-1}(s)\pi^\star_{\omega_{k-1}}(a|s) - \mathbb{P}_{h-1}^{k-1}(s)\pi^\star_{\omega_{k-1}}(a|s)\right)P(s'|s,a) \\ &= \sum_s \mathbb{P}_{h-1}^k(s)\left[\sum_a\left(\pi^\star_{\omega_k}(a|s) - \pi^\star_{\omega_{k-1}}(a|s)\right)P(s'|s,a)\right] \\ &\quad + \sum_s\left(\mathbb{P}_{h-1}^k(s) - \mathbb{P}_{h-1}^{k-1}(s)\right)\left[\sum_a \pi^\star_{\omega_{k-1}}(a|s)P(s'|s,a)\right].\end{aligned} \tag{12}$$

Taking absolute value on both side, yields

$$\begin{aligned}&\left\|\mathbb{P}_h^k - \mathbb{P}_h^{k-1}\right\|_1 = \sum_{s'}\left|\mathbb{P}_h^k(s') - \mathbb{P}_h^{k-1}(s')\right| \\ &\leq \sum_s \mathbb{P}_{h-1}^k(s)\sum_a \underbrace{\left|\pi^\star_{\omega_k}(a|s) - \pi^\star_{\omega_{k-1}}(a|s)\right|}_{\leq c_1\|\omega_k - \omega_{k-1}\|_2}\sum_{s'}P(s'|s,a) \\ &\quad + \sum_s\left|\mathbb{P}_{h-1}^k(s) - \mathbb{P}_{h-1}^{k-1}(s)\right|\left[\sum_{s'}\sum_a \pi^\star_{\omega_{k-1}}(a|s)P(s'|s,a)\right] \\ &\overset{(i)}{\leq} c_1\|\omega_k - \omega_{k-1}\|_2 + \left\|\mathbb{P}_{h-1}^k - \mathbb{P}_{h-1}^{k-1}\right\|_1 \leq \cdots \leq c_1 h\|\omega_k - \omega_{k-1}\|_2 + \left\|\mathbb{P}_0^k - \mathbb{P}_0^{k-1}\right\|_1 \\ &\overset{(ii)}{=} c_1 h\|\omega_k - \omega_{k-1}\|_2 + \|\mu_k - \mu_{k-1}\|_1,\end{aligned} \tag{13}$$

where inequality (i) holds by applying Lemma B.2 with $c_1 = L_r/\alpha(1-\gamma)$ and equality (ii) holds because the initial distribution of $\mathbb{P}_h^k$ is μ_k. By the definition of d_μ^π, we have

$$d_{\mu_k}^{\pi^\star_{\omega_k}}(s) - d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}(s) \overset{(i)}{=} d_k(s) - d_{k-1}(s) = (1-\gamma)\sum_{h=0}^\infty \gamma^h\left(\mathbb{P}_h^k(s) - \mathbb{P}_h^{k-1}(s)\right), \ \forall s \in \mathcal{S}. \tag{14}$$

where in equality (i), we use d_k to denote $d_{\mu_k}^{\pi^\star_{\omega_k}}$. Adding ℓ^1 norm on both sides of (14) and applying (13), yields

$$\begin{aligned}&\|d_k - d_{k-1}\|_1 \leq (1-\gamma)\sum_{h=0}^\infty \gamma^h\left(c_1 h\|\omega_k - \omega_{k-1}\|_2 + \|\mu_k - \mu_{k-1}\|_1\right) \\ &\overset{(i)}{=} \frac{\gamma c_1}{1-\gamma}\|\omega_k - \omega_{k-1}\|_2 + \|\mu_k - \mu_{k-1}\|_1 \overset{(ii)}{=} \frac{\gamma c_1}{1-\gamma}\|\omega_k - \omega_{k-1}\|_2 + \beta\|d_{k-1} - d_{k-2}\|_1,\end{aligned} \tag{15}$$

where equality (i) holds because $\sum_{h=0}^{\infty}\gamma^h h=\gamma/(1-\gamma)^2$ and equality (ii) holds because of (1). Hence, we know that

$$\begin{aligned}\|d_k-d_{k-1}\|_1 \leq& \frac{\gamma c_1}{1-\gamma}\|\omega_k-\omega_{k-1}\|_2+\beta\|d_{k-1}-d_{k-2}\|_1\\ \leq& \frac{\gamma c_1}{1-\gamma}\sum_{i=0}^{k-1}\left[\|\omega_{i+1}-\omega_i\|_2\,\beta^{k-i}\right]+\beta^{k-1}\|d_1-d_0\|_1\\ \leq& \frac{\gamma c_1}{1-\gamma}\cdot\frac{1}{1-\beta}\Delta_\omega^k+\beta^{k-1}\|d_1-d_0\|_1\leq\left(\frac{\gamma c_1}{(1-\gamma)(1-\beta)}+1\right)\Delta_\omega^k\end{aligned} \tag{16}$$

where $\Delta_\omega^k=\max_{1\leq i\leq k}\|\omega_i-\omega_{i-1}\|_2$ and the last inequality holds due to the near optimality definition (Definition 4.2). Therefore, applying (16) back to (11), we know that

$$\begin{aligned}\left\|\frac{d_{\mu_k}^{\pi^\star_{\omega_k}}}{\mu_k}\right\|_\infty \leq& \frac{\left\|d_{\mu_k}^{\pi^\star_{\omega_k}}-d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}\right\|_1}{\min\mu_k}+\frac{1}{\beta}\\ \overset{(i)}{\leq}& \frac{1}{\min\mu_k}\left(\frac{\gamma c_1}{(1-\gamma)(1-\beta)}+1\right)\Delta_\omega^k+\frac{1}{\beta}=\widetilde{O}\left(\frac{L_r}{\alpha(1-\beta)}\Delta_\omega^k S\right),\end{aligned} \tag{17}$$

where inequality (i) holds since (1) Lemma B.2 implies $c_1=L_r/\alpha(1-\gamma)$, and we omit the $1/(1-\gamma)^6$ and $\log$ in the $\widetilde{O}$; (2) $1/\min\mu_k\leq S/(1-\beta)$ according to $\mu_k=\beta d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}+(1-\beta)\rho$. Note that we can only achieve the final bound $\widetilde{O}\left(\frac{L_r}{\alpha(1-\beta)}\Delta_\omega^k S\right)$ by setting β as a constant. If we pick an arbitrarily small β, then the $1/\beta$ term will dominate the complexity and we will not have the final bound of $\widetilde{O}\left(\frac{L_r}{\alpha(1-\beta)}\Delta_\omega^k S\right)$. □

A.3. Complexity of Vanilla Stochastic PG

Theorem A.2 (Complexity of Stochastic PG (Theorem 5.1 of (Ding et al., 2021))). *Consider an arbitrary tolerance level $\delta>0$ and a small enough tolerance level $\varepsilon>0$. For every initial point θ_0, if θ_{T+1} is generated by SPG (Algorithm 4) with*

$$\begin{aligned}&T_1\geq\left(\frac{6D(\theta_0)}{\delta\varepsilon_0}\right)^{\frac{8L}{C_\delta^0\ln 2}},\ T_2\geq\left(\frac{\varepsilon_0}{6\delta\varepsilon}-1\right)t_0,\ T=T_1+T_2,\\ &B_1\geq\max\left\{\frac{30\sigma^2}{C_\delta^0\varepsilon_0\delta},\frac{6\sigma T_1\log T_1}{\bar{\Delta}L}\right\},\ B_2\geq\frac{\sigma^2\ln(T_2+t_0)}{6C_\zeta\delta\varepsilon},\\ &\eta_t=\eta\leq\min\left\{\frac{\log T_1}{T_1L},\frac{8}{C_\delta^0},\frac{1}{2L}\right\}\ \forall 1\leq t\leq T_1,\ \eta_t=\frac{1}{t-T_1+t_0}\ \forall t>T_1,\end{aligned} \tag{18}$$

where

$$\begin{aligned}&D(\theta_t)=V^{\pi^\star}(\rho)-V^{\pi_{\theta_t}}(\rho),\ \varepsilon_0=\min\left\{\left(\frac{\alpha\min_{s\in\mathcal{S}}\rho(s)}{6\ln 2}\right)^2\left[\zeta\exp\left(-\frac{1}{(1-\gamma)\alpha}\right)\right]^4,1\right\},\\ &t_0\geq\sqrt{\frac{3\sigma^2}{2\delta\varepsilon_0}},\ C_\delta^0=\frac{2\alpha}{S}\left\|\frac{d_\rho^{\pi^\star}}{\rho}\right\|_\infty^{-1}\min_{s\in\mathcal{S}}\rho(s)\min_{\theta\in\mathcal{G}_\delta^0}\min_{s,a}\pi_\theta(a|s)^2,\\ &C_\zeta=\frac{2\alpha}{S}\left\|\frac{d_\rho^{\pi^\star}}{\rho}\right\|_\infty^{-1}\min_{s\in\mathcal{S}}\rho(s)(1-\zeta)^2\min_{s,a}\pi^\star(a|s)^2,\\ &\mathcal{G}_\delta^0:=\left\{\theta\in\mathbb{R}^{S\times A}:\min_{\theta^\star\in\Theta^\star}\|\theta-\theta^\star\|_2\leq(1+1/\delta)\bar{\Delta}\right\},\ \bar{\Delta}=\left\|\log c_{\bar{\theta}_1,\eta}-\log\pi^\star\right\|_2,\\ &c_{\bar{\theta}_1,\eta}=\inf_{t\geq 1}\min_{s,a}\pi_{\theta_t}(a|s),\ \sigma^2=\frac{8}{(1-\gamma)^2}\left(\frac{1+(\alpha\log A)^2}{(1-\gamma^{1/2})^2}\right),\ L=\frac{8+\alpha(4+8\log A)}{(1-\gamma)^3},\end{aligned} \tag{19}$$

then we have $\mathbb{P}(D(\theta_{T+1})\leq\varepsilon)\geq 1-\delta$.[2]

[2]Note that the ζ here is an optimization constant that appears in ε_0 and C_ζ.

Corollary A.3 (Iteration Complexity and Sample Complexity for ε-Optimal Policies)**.** *Suppose we set the tolerance level $\varepsilon, \delta = O(S^{-1})$, the iteration complexity and sample complexity of obtaining an ε-optimal policy using stochastic softmax policy gradient (Algorithm 4) in phase 1 and phase 2 satisfies:*

- *Phase 1:* $T_1 = \widetilde{\Omega}\left(S^{2S^3}\right)$, $B_1 = \widetilde{\Omega}\left(S^{2S^3}\right)$,
- *Phase 2:* $T_2 = \widetilde{\Omega}\left(S\right)$, $B_2 = \widetilde{\Omega}\left(S^5\right)$,

with probability at least $1-\delta$.

Proof. We first check the dependency of (19) on S. Notice that

- ε_0:
$$\frac{1}{\varepsilon_0} = \max\left\{\left(\frac{6\ln 2}{\alpha \min_{s\in\mathcal{S}}\rho(s)}\right)^2\left[\zeta\exp\left(-\frac{1}{(1-\gamma)\alpha}\right)\right]^{-4}, 1\right\} = \widetilde{\Omega}(S^2); \tag{20}$$
- t_0:
$$t_0 \geq \sqrt{\frac{3\sigma^2}{2\delta\varepsilon_0}} = \widetilde{\Omega}(S); \tag{21}$$
- C_δ^0:
$$\frac{1}{C_\delta^0} = \frac{S}{2\alpha}\left\|\frac{d_\rho^{\pi^\star}}{\rho}\right\|_\infty \max_{s\in\mathcal{S}}\rho(s)^{-1}\frac{1}{\min_{\theta\in\mathcal{G}_\delta^0}\min_{s,a}\pi_\theta(a|s)^2} = \widetilde{\Omega}(S^3); \tag{22}$$
- C_ζ:
$$\frac{1}{C_\zeta} = \frac{S}{2\alpha}\left\|\frac{d_\rho^{\pi^\star}}{\rho}\right\|_\infty \max_{s\in\mathcal{S}}\rho(s)^{-1}(1-\zeta)^{-2}\max_{s,a}\pi^\star(a|s)^{-2} = \widetilde{\Omega}(S^3). \tag{23}$$

Hence, the complexities in phase 1 scales at

$$T_1 \geq \left(\frac{6D(\theta_0)}{\delta\varepsilon_0}\right)^{\frac{8L}{C_\delta^0\ln 2}} = \widetilde{\Omega}\left(S^{2S^3}\right),\ B_1 \geq \max\left\{\frac{30\sigma^2}{C_\delta^0\varepsilon_0\delta}, \frac{6\sigma T_1\log T_1}{\bar{\Delta}L}\right\} = \widetilde{\Omega}\left(S^{2S^3}\right). \tag{24}$$

To enforce a positive T_2, the tolerance level ε, δ should satisfy $\frac{\varepsilon_0}{6\delta\varepsilon} \geq 1$, which implies $\frac{1}{\delta\varepsilon} = \Omega(S^2)$. Hence, assuming $\frac{\varepsilon_0}{\delta\varepsilon} = o(S)$, $\varepsilon, \delta = O(S^{-1})$, then the complexities in phase 2 scales at

$$T_2 \geq \left(\frac{\varepsilon_0}{6\delta\varepsilon} - 1\right)t_0 = \widetilde{\Omega}\left(S\right),\ B_2 \geq \frac{\sigma^2\ln(T_2+t_0)}{6C_\zeta\delta\varepsilon} = \widetilde{\Omega}\left(S^5\right). \tag{25}$$

□

A.4. Complexity of Learning the Next Context

Theorem A.4 (Theorem 4.1: Complexity of Learning the Next Context)**.** *Consider the context-based stochastic softmax policy gradient (line 7 of Algorithm 1), suppose Assumption 3.1 and Assumption 3.2 hold, then the iteration number of obtaining an ε-optimal policy for ω_k from $\theta^\star_{\omega_{k-1}}$ is $\widetilde{\Omega}\left(S\right)$ and the per iteration sample complexity is $\widetilde{\Omega}\left(\frac{L_r}{\alpha(1-\beta)}S^3\right)$.*

We first introduce the following lemma to aid the proof of Theorem A.4.

Lemma A.5 (Bounded Optimal Values Between two Adjacent Contexts)**.** *Under the same conditions as Theorem A.4, we have*

$$V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) \leq \frac{2L_r\left\|\omega_k - \omega_{k-1}\right\|_2}{(1-\gamma)^2}. \tag{26}$$

Proof. Let V_ω^π denote the value function of policy π with reward function r_ω. From (65) of Lemma B.3, we know that for any initial distribution ρ, we have

$$V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) = \frac{1}{1-\gamma}\sum_s \left[d_\rho^{\pi^\star_{\omega_{k-1}}}(s) \cdot \alpha \cdot D_{\mathrm{KL}}\left(\pi^\star_{\omega_{k-1}}(\cdot|s)||\pi^\star_{\omega_k}(\cdot|s)\right)\right]. \tag{27}$$

From (47) of Lemma B.1, we know that

$$\begin{aligned}
\pi^\star_{\omega_{k-1}}(a|s) &= \left[\mathrm{softmax}(Q^{\pi^\star_{\omega_{k-1}}}(\cdot,s)/\alpha)\right]_a := \frac{\exp\left[Q^{\pi^\star_{\omega_{k-1}}}(s,a)/\alpha\right]}{\sum_{a'}\exp\left[Q^{\pi^\star_{\omega_{k-1}}}(s,a')/\alpha\right]} \\
\pi^\star_{\omega_k}(a|s) &= \left[\mathrm{softmax}(Q^{\pi^\star_{\omega}}(\cdot,s)/\alpha)\right]_a := \frac{\exp\left[Q^{\pi^\star_{\omega_k}}(s,a)/\alpha\right]}{\sum_{a'}\exp\left[Q^{\pi^\star_{\omega_k}}(s,a')/\alpha\right]},
\end{aligned} \tag{28}$$

hence, we have

$$\begin{aligned}
&D_{\mathrm{KL}}\left(\pi^\star_{\omega_{k-1}}(\cdot|s)||\pi^\star_{\omega_k}(\cdot|s)\right) \\
&= \sum_a \pi^\star_{\omega_{k-1}}(a|s)\left\{\log\left(\left[\mathrm{softmax}(Q^{\pi^\star_{\omega_{k-1}}}(a,s)/\alpha)\right]_a\right) - \log\left(\left[\mathrm{softmax}(Q^{\pi^\star_{\omega_k}}(a,s)/\alpha)\right]_a\right)\right\}.
\end{aligned} \tag{29}$$

Let $f(\boldsymbol{x})$ denote the log soft max function for an input vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_A]^\top$ such that $x_i \geq 0$, then for a small perturbation $\boldsymbol{\Delta} \in \mathbb{R}^A$, the intermediate value theorem implies

$$|[f(\boldsymbol{x}+\boldsymbol{\Delta})]_i - [f(\boldsymbol{x})]_i| = \left|\boldsymbol{\Delta}^\top \nabla_{\boldsymbol{z}}\left[f(\boldsymbol{z})\right]_i\right|, \tag{30}$$

for some vector $\boldsymbol{z}$ on the segment $[\boldsymbol{x}, \boldsymbol{x}+\boldsymbol{\Delta}]$. Now consider the Jacobian of the log softmax function $\partial[\nabla_{\boldsymbol{z}} f(\boldsymbol{z})]_i/\partial z_j$:

$$\frac{\partial[\nabla_{\boldsymbol{z}} f(\boldsymbol{z})]_i}{\partial z_j} = \begin{cases} 1 - p_i(\boldsymbol{z}) \in (0,1) & \text{if } i = j, \\ -p_j(\boldsymbol{z}) \in (-1,0) & \text{otherwise,} \end{cases} \tag{31}$$

where $p_i(\boldsymbol{z}) = \exp(z_i)/\sum_{k=1}^A \exp(z_k)$. hence, we know that

$$\begin{aligned}
|[f(\boldsymbol{x}+\boldsymbol{\Delta})]_i - [f(\boldsymbol{x})]_i| &= \left|\boldsymbol{\Delta}^\top \nabla_{\boldsymbol{z}}\left[f(\boldsymbol{z})\right]_i\right| \leq \|\boldsymbol{\Delta}\|_\infty \sum_{k=1}^A \left|\frac{\partial[f(\boldsymbol{z})]_i}{\partial z_k}\right| \\
&= \|\boldsymbol{\Delta}\|_\infty \left(1 - p_i(\boldsymbol{z}) + \sum_{j\neq i} p_j(\boldsymbol{z})\right) \leq 2\|\boldsymbol{\Delta}\|_\infty.
\end{aligned} \tag{32}$$

Now let

$$\begin{aligned}
\boldsymbol{x} &= \frac{1}{\alpha}[Q^{\pi^\star_{\omega_{k-1}}}(s,a_1), Q^{\pi^\star_{\omega_{k-1}}}(s,a_2), \ldots, Q^{\pi^\star_{\omega_{k-1}}}(s,a_A)], \\
\boldsymbol{x}+\boldsymbol{\Delta} &= \frac{1}{\alpha}[Q^{\pi^\star_{\omega_k}}(s,a_1), Q^{\pi^\star_{\omega_k}}(s,a_2), \ldots, Q^{\pi^\star_{\omega_k}}(s,a_A)],
\end{aligned} \tag{33}$$

(57) from Lemma B.2 implies that

$$\frac{1}{\alpha}\left\|Q^{\pi^\star_{\omega_k}} - Q^{\pi^\star_{\omega_{k-1}}}\right\|_\infty \leq \frac{L_r\|\omega_k - \omega_{k-1}\|_2}{\alpha(1-\gamma)}, \tag{34}$$

substituting (34) and (32) into (29), yields

$$D_{\mathrm{KL}}\left(\pi^\star_{\omega_{k-1}}(\cdot|s)||\pi^\star_{\omega_k}(\cdot|s)\right) \leq \sum_a 2\pi^\star_{\omega_{k-1}}(a|s)\|\boldsymbol{\Delta}\|_\infty \leq 2\|\boldsymbol{\Delta}\|_\infty \leq \frac{2L_r\|\omega_k - \omega_{k-1}\|_2}{\alpha(1-\gamma)}. \tag{35}$$

Combine (35) with (27), we have

$$\begin{aligned} V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) =& \frac{1}{1-\gamma}\sum_s \left[d_\rho^{\pi^\star_{\omega_{k-1}}}(s) \cdot \alpha \cdot D_{\mathrm{KL}}\left(\pi^\star_{\omega_{k-1}}(\cdot|s) || \pi^\star_{\omega_k}(\cdot|s)\right)\right] \\ \leq& \frac{2L_r \left\|\omega_k - \omega_{k-1}\right\|_2}{(1-\gamma)^2}, \end{aligned} \tag{36}$$

which completes the proof. □

Now we are ready to proceed to the proof of Theorem A.4.

Proof. From (19) we know that

$$\varepsilon_0 = \min\left\{ \left(\frac{\alpha \min_{s\in\mathcal{S}} \rho(s)}{6\ln 2} \right)^2 \left[\zeta \exp\left(-\frac{1}{(1-\gamma)\alpha} \right) \right]^4, 1 \right\} = O\left(\frac{1}{S^2} \right). \tag{37}$$

And from Section 6.2 of (Ding et al., 2021), we can directly enter phase 2 of the stochastic PG when

$$V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) \leq \varepsilon_0. \tag{38}$$

Hence, when $\Delta_\omega^k = \max_{1\leq i\leq k} \left\|\omega_i - \omega_{i-1}\right\|_2 = O(1/S^2)$, we have

$$V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) \leq \frac{2L_r \Delta_\omega}{(1-\gamma)^2} \leq \frac{\varepsilon_0}{2}, \tag{39}$$

which implies we can directly enter phase 2 and enjoys the faster iteration complexity of $T_2 = \Omega(S)$ (by choosing $\delta = O(S^{-1})$) and the smaller batch size of

$$B_2 \geq \frac{\sigma^2 \ln(T_2 + t_0)}{6 C_\zeta \delta \varepsilon} \overset{(i)}{=} \widetilde{\Omega}\left(\frac{L_r}{\alpha(1-\beta)} \Delta_\omega^k S^5 \right) \overset{(ii)}{=} \widetilde{\Omega}\left(\frac{L_r}{\alpha(1-\beta)} S^3 \right), \tag{40}$$

where equation (i) holds by applying Lemma A.1 to (23):

$$\frac{\sigma^2 \ln(T_2 + t_0)}{6 C_\zeta \delta \varepsilon} = \widetilde{\Omega}\left(S^4 \cdot \left\| d_{\mu_k}^{\pi^\star_{\omega_k}} / \mu_k \right\|_\infty \right) = \widetilde{\Omega}\left(\frac{L_r}{\alpha(1-\beta)} \Delta_\omega^k S^5 \right),$$

and equality (ii) holds by the assumption that $\Delta_\omega^k = O(S^{-2})$ and we omit the log term and components not related to S in $\widetilde{\Omega}$. □

A.5. Total Complexity of ROLLIN

Theorem A.6 (Theorem 4.3: Total Complexity of Learning the Target Context)**.** *Suppose Assumption 3.1 and Assumption 3.2 hold, and $\theta_0^{(0)}$ is an near-optimal initialization, then the total number of iteration of learning $\pi^\star_{\omega_K}$ using Algorithm 1 is $\Omega(KS)$ and the per iteration is $\widetilde{\Omega}\left(S^3\right)$, with high probability.*

Proof. From Lemma A.5, we know that

$$V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) \leq \frac{2L_r \left\|\omega_k - \omega_{k-1}\right\|_2}{(1-\gamma)^2}. \tag{41}$$

Suppose for each context ω_k, we initialize the parameters of the policy as $\theta_0^{(k)} = \theta^\star_{\omega_{k-1}}$, and let $\theta_t^{(k)}$ denote the parameters at the t^{th} iteration of SPG. We will use induction to show that when $t = \widetilde{\Omega}(S)$, $\forall k \in [K]$, we have

$$V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi_{\theta_t^{(k-1)}}}(\rho) < \varepsilon_0, \tag{42}$$

this implies that for any context $\omega_k, k \in [K]$, we can always find a good initialization by setting $\theta_0^{(k)} = \theta_t^{(k-1)}$ from learning $\pi^\star_{\omega_{k-1}}$ using SPG after $t = \Omega(S)$ iteration. This result guarantees that every initialization $\theta_0^{(k)}$ for learning the optimal contextual policy $\pi^\star_{\omega_k}$ will directly start from the efficient phase 2.

Induction: $k = 0$**.** When $k = 0$, Assumption 3.2 and the near-optimal initialization (Definition 4.2) of $\theta_0^{(0)}$ implies that

$$V_{\omega_0}^{\pi^\star_{\omega_0}}(\rho) - V_{\omega_0}^{\pi_{\theta_0^{(0)}}}(\rho) < \varepsilon_0. \tag{43}$$

This result implies that a near-optimal initialization allows the initialization to directly start from phase 2 of SPG.

Induction: from $k-1$ **to** k**.** Suppose the result in (42) holds for $k-1$, then we know that

$$V_{\omega_{k-1}}^{\pi^\star_{\omega_{k-1}}}(\rho) - V_{\omega_{k-1}}^{\pi_{\theta_0^{(k-1)}}}(\rho) = V_{\omega_{k-1}}^{\pi^\star_{\omega_{k-1}}}(\rho) - V_{\omega_{k-1}}^{\pi_{\theta_t^{(k-2)}}}(\rho) < \varepsilon_0. \tag{44}$$

Select ε such that $\varepsilon \le \varepsilon_0/2$. Theorem A.4 suggests that when $t' = \tilde{\Omega}(S)$, with high probability, we have

$$V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi_{\theta_{t'}^{(k-1)}}}(\rho) < \varepsilon \le \frac{\varepsilon_0}{2}. \tag{45}$$

Hence, if we initialize $\theta_0^{(k)} = \theta_t^{(k-1)}$, with high probability when $t' = \widetilde{\Omega}(S)$, we have

$$\begin{aligned} &V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi_{\theta_{t'}^{(k-1)}}}(\rho) = V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) + V_{\omega_k}^{\pi^\star_{\omega_{k-1}}}(\rho) - V_{\omega_k}^{\pi_{\theta_{t'}^{(k-1)}}}(\rho) \\ &\overset{(i)}{\le} \frac{\varepsilon_0}{2} + V_{\omega_k}^{\pi^\star_{\omega_k}}(\rho) - V_{\omega_k}^{\pi_{\theta_{t'}^{(k-1)}}}(\rho) \overset{(ii)}{<} \varepsilon_0, \end{aligned} \tag{46}$$

where inequality (i) holds by equation (39) in Theorem A.4, inequality (ii) holds because of the induction assumption in (45).

Therefore, we have shown (42) holds for $t = \widetilde{\Omega}(S)$, $\forall k \in [K]$. Since we have K contexts in total, we know that Algorithm 1 can enforce a good initialization $\theta_0^{(k)}$ that directly starts from phase 2 for learning all $\pi^\star_{\omega_k}$, and for each $k \in [K]$, the iteration complexity is $\widetilde{\Omega}(S)$. Hence the total iteration complexity of obtaining an ε-optimal policy for the final context ω_K is $\widetilde{\Omega}(KS)$, with per iteration sample complexity of $\widetilde{\Omega}(S^3)$. □

B. Key Lemmas

B.1. Optimal Policy of Maximum Entropy RL (Nachum et al., 2017)

Lemma B.1. *The optimal policy $\pi^\star$ that maximizes the α-MaxEnt RL objective (4) with penalty term α satisfies:*

$$\pi^\star(a|s) = \exp\left[\left(Q^{\pi^\star}(s,a) - V^{\pi^\star}(s)\right)/\alpha\right] = \frac{\exp\left(Q^{\pi^\star}(s,a)/\alpha\right)}{\sum_a \exp\left(Q^{\pi^\star}(s,a)/\alpha\right)} \tag{47}$$

for all $h \in \mathbb{N}$, where

$$\begin{aligned} Q^{\pi^\star}(s,a) &:= r(s,a) + \gamma \mathbb{E}_{s' \sim P(s'|s,a)} V(s') \\ V^{\pi^\star}(s) &:= \alpha \log\left(\sum_a \exp\left(Q^{\pi^\star}(s,a)/\alpha\right)\right). \end{aligned} \tag{48}$$

Proof. Similar proof appears in (Nachum et al., 2017), we provide the proof for completeness. At the optimal policy $\pi_\theta = \pi^\star$, take the gradient of (4) w.r.t. $p \in \Delta(\mathcal{A})$ and set it to 0, we have

$$\frac{\partial}{\partial p(a)}\left[\sum_{a\in\mathcal{A}} p(a)\left(Q^{\pi^\star}(s,a) - \alpha \ln p(a)\right)\right] = Q^{\pi^\star}(s,a) - \alpha \ln p(a) - \alpha = 0, \tag{49}$$

which implies

$$p(a) = \exp\left(\frac{Q^{\pi^\star}(s,a)}{\alpha} - 1\right) \propto \exp\left(\frac{Q^{\pi^\star}(s,a)}{\alpha}\right). \tag{50}$$

Hence, we conclude that $\pi^\star(a|s) \propto \exp(Q^\star(s,a)/\alpha)$. □

B.2. Bounding the Difference between Optimal Policies

Lemma B.2. *Suppose Assumption 3.1 holds, let $\pi^\star_\omega(a|s), \pi^\star_{\omega'}(a|s)$ denote the optimal policy for α-MaxEnt RL (47), then $\forall (s,a) \in \mathcal{S} \times \mathcal{A}$, the optimal policies of α-MaxEnt RL under context ω, ω' satisfy:*

$$|\pi^\star_\omega(a|s) - \pi^\star_{\omega'}(a|s)| \leq \frac{L_r \|\omega - \omega'\|_2}{\alpha(1-\gamma)}. \tag{51}$$

Proof. From Lemma C.1, we know that the soft value iteration

$$\mathcal{T}Q(s,a) = r(s,a) + \gamma\alpha \mathbb{E}_{s'}\left[\log \sum_{a'} \exp Q(s',a')/\alpha\right] \tag{52}$$

is a contraction. Let $Q^t_\omega, Q^t_{\omega'}$ denote the Q functions at the t^{th} value iteration under context ω, ω' respectively, we know $Q^\infty_\omega = Q^{\pi^\star}_\omega$ and $Q^\infty_{\omega'} = Q^{\pi^\star}_{\omega'}$. Let $\varepsilon_t = \|Q^t_\omega - Q^t_{\omega'}\|_\infty$, then we have

$$\begin{aligned}
\varepsilon_{t+1} &= \left\|Q^{t+1}_\omega - Q^{t+1}_{\omega'}\right\|_\infty \\
&= \left\| r_\omega(s,a) - r_{\omega'}(s,a) + \gamma\alpha\mathbb{E}_{s'}\left[\log\sum_{a'}\exp\frac{Q^t_{\omega'}(s',a')}{\alpha}\right] - \gamma\alpha\mathbb{E}_{s'}\left[\log\sum_{a'}\exp\frac{Q^t_{\omega'}(s',a')}{\alpha}\right]\right\|_\infty \\
&\leq \|r_\omega - r_{\omega'}\|_\infty + \gamma\alpha\left\|\mathbb{E}_{s'}\log\sum_{s'}\exp Q^t_\omega(s',a')/\alpha - \mathbb{E}_{s'}\log\sum_{s'}\exp Q^t_{\omega'}(s',a')/\alpha\right\|_\infty \\
&\leq \|r_\omega - r_{\omega'}\|_\infty + \gamma\left\|Q^t_\omega - Q^t_{\omega'}\right\|_\infty = \|r_\omega - r_{\omega'}\|_\infty + \gamma\varepsilon_t,
\end{aligned} \tag{53}$$

where the last inequality holds because $f(\boldsymbol{x}) = \log\sum_{i=1}^n \exp(x_i)$ is a contraction. From (53), we have

$$\varepsilon_{t+1} \leq \|r_\omega - r_{\omega'}\|_\infty + \gamma\varepsilon_t \leq (1+\gamma)\|r_\omega - r_{\omega'}\|_\infty + \gamma^2\varepsilon_{t-1} \leq \cdots \leq \|r_\omega - r_{\omega'}\|_\infty \sum_{i=0}^t \gamma^i + \gamma^t\varepsilon_1, \tag{54}$$

which implies

$$\left\|Q^{\pi^\star}_\omega - Q^{\pi^\star}_{\omega'}\right\|_\infty = \varepsilon_\infty \leq \frac{\|r_\omega - r_{\omega'}\|_\infty}{1-\gamma} \leq \frac{L_r\|\omega-\omega'\|_2}{1-\gamma}, \tag{55}$$

where the last inequality holds by Assumption 3.1. Hence, we have

$$\frac{1}{\alpha}\left|Q^{\pi^\star_\omega}(s,a) - Q^{\pi^\star_{\omega'}}(s,a)\right| \leq \frac{L_r\|\omega-\omega'\|_2}{\alpha(1-\gamma)},\ \forall s,a \in \mathcal{S}\times\mathcal{A} \tag{56}$$

which implies

$$\frac{1}{\alpha}\left\|Q^{\pi^\star_\omega} - Q^{\pi^\star_{\omega'}}\right\|_\infty \leq \frac{L_r\|\omega-\omega'\|_2}{\alpha(1-\gamma)}. \tag{57}$$

Next, let $\pi^\star_\omega, \pi^\star_{\omega'}$ denote the maximum entropy policy RL under context ω, ω' respectively. Then for a fixed state action pair $(s,a) \in \mathcal{S}\times\mathcal{A}$, we have

$$\begin{aligned}
\pi^\star_\omega(a|s) &= \left[\text{softmax}(Q^{\pi^\star_\omega}(\cdot,s)/\alpha)\right]_a := \frac{\exp\left[Q^{\pi^\star_\omega}(s,a)/\alpha\right]}{\sum_{a'}\exp\left[Q^{\pi^\star_\omega}(s,a')/\alpha\right]}, \\
\pi^\star_{\omega'}(a|s) &= \left[\text{softmax}(Q^{\pi^\star_{\omega'}}(\cdot,s)/\alpha)\right]_a := \frac{\exp\left[Q^{\pi^\star_{\omega'}}(s,a)/\alpha\right]}{\sum_{a'}\exp\left[Q^{\pi^\star_{\omega'}}(s,a')/\alpha\right]},
\end{aligned} \tag{58}$$

where $Q^{\pi^\star_\omega}(\cdot,s), Q^{\pi^\star_{\omega'}}(\cdot,s) \in \mathbb{R}^A$, and we want to bound $|\pi^\star_\omega(a|s) - \pi^\star_{\omega'}(a|s)|$. Next we will use (57) to bound $|\pi^\star_\omega(a|s) - \pi^\star_{\omega'}(a|s)|$, where the last inequality holds by (56). Let $f(\boldsymbol{x})$ denote the softmax function for an input vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_A]^\top$ such that $x_i \geq 0$, then for a small perturbation $\boldsymbol{\Delta} \in \mathbb{R}^A$, the intermediate value theorem implies

$$|[f(\boldsymbol{x}+\boldsymbol{\Delta})]_i - [f(\boldsymbol{x})]_i| = \left|\boldsymbol{\Delta}^\top\nabla_{\boldsymbol{x}}[f(\boldsymbol{z})]_i\right|, \tag{59}$$

for some vector $\boldsymbol{z}$ on the segment $[\boldsymbol{x}, \boldsymbol{x}+\boldsymbol{\Delta}]$. Hence

$$\begin{aligned} &|[f(\boldsymbol{x}+\boldsymbol{\Delta})]_i - [f(\boldsymbol{x})]_i| = \left|\boldsymbol{\Delta}^\top [\nabla_{\boldsymbol{x}} f(\boldsymbol{z})]_i\right| \leq \|\boldsymbol{\Delta}\|_\infty \sum_{k=1}^{A} \left|\frac{\partial [f(\boldsymbol{z})]_i}{\partial z_k}\right| \\ &\leq \|\boldsymbol{\Delta}\|_\infty \left(p_i(\boldsymbol{z})(1-p_i(\boldsymbol{z})) + \sum_{j\neq i} p_i(\boldsymbol{z}) p_j(\boldsymbol{z})\right) < \|\boldsymbol{\Delta}\|_\infty \left(p_i(\boldsymbol{z}) + \sum_{j\neq i} p_j(\boldsymbol{z})\right) = \|\boldsymbol{\Delta}\|_\infty , \end{aligned} \tag{60}$$

where the Jacobian of the softmax function $\partial \left[\nabla_{\boldsymbol{x}} f(\boldsymbol{z})\right]_i / \partial z_j$ satisfies:

$$\frac{\partial \left[\nabla_{\boldsymbol{x}} f(\boldsymbol{z})\right]_i}{\partial z_j} = \begin{cases} p_i(\boldsymbol{z})(1-p_i(\boldsymbol{z})) & \text{if } i=j, \\ p_i(\boldsymbol{z}) p_j(\boldsymbol{z}) & \text{otherwise,} \end{cases} \tag{61}$$

and $p_i(\boldsymbol{z}) = \exp(z_i)/\sum_{k=1}^{A}\exp(z_k)$. Now let

$$\begin{aligned} \boldsymbol{x} &= \frac{1}{\alpha}[Q^{\pi^\star_\omega}(s,a_1), Q^{\pi^\star_\omega}(s,a_2), \ldots, Q^{\pi^\star_\omega}(s,a_A)], \\ \boldsymbol{x}+\boldsymbol{\Delta} &= \frac{1}{\alpha}[Q^{\pi^\star_{\omega'}}(s,a_1), Q^{\pi^\star_{\omega'}}(s,a_2), \ldots, Q^{\pi^\star_{\omega'}}(s,a_A)]. \end{aligned} \tag{62}$$

We know that $f(\boldsymbol{x}) = \pi^\star_\omega(a|s)$ and $f(\boldsymbol{x}+\boldsymbol{\Delta}) = \pi^\star_{\omega'}(a|s)$. Then (57) implies that

$$\|\boldsymbol{\Delta}\|_\infty \leq \frac{L_r \|\omega - \omega'\|_2}{\alpha(1-\gamma)}, \tag{63}$$

substituting this bound on $\|\boldsymbol{\Delta}\|_\infty$ into (60), we have

$$|\pi^\star_\omega(a|s) - \pi^\star_{\omega'}(a|s)| = |f(\boldsymbol{x}) - f(\boldsymbol{x}+\boldsymbol{\Delta})| \leq \|\boldsymbol{\Delta}\|_\infty \leq \frac{L_r \|\omega - \omega'\|_2}{\alpha(1-\gamma)}, \tag{64}$$

which completes the proof. □

B.3. Soft Sub-Optimality lemma (Lemma 25 & 26 of (Mei et al., 2020))

Lemma B.3. *For any policy π and any initial distribution ρ, the value function $V^\pi(\rho)$ of the α-MaxEnt RL (48) satisfies:*

$$V^{\pi^\star}(\rho) - V^\pi(\rho) = \frac{1}{1-\gamma}\sum_s \left[d^\pi_\rho(s)\cdot\alpha\cdot D_{\mathrm{KL}}\left(\pi(\cdot|s)||\pi^\star(\cdot|s)\right)\right], \tag{65}$$

where $\pi^\star$ is the optimal policy of the α-MaxEnt RL (4).

Proof. Similar proof appears in Lemma 25 & 26 of (Mei et al., 2020), we provide the proof here for completeness.

Soft performance difference. We first show a soft performance difference result for the MaxEnt value function (Lemma 25 of (Mei et al., 2020)). By the definition of MaxEnt value function and Q-function (4), (6), $\forall \pi, \pi'$, we have

$$
\begin{aligned}
&V^{\pi'}(s) - V^{\pi}(s) \\
=&\sum_a \pi'(a|s) \cdot \left[Q^{\pi'}(s,a) - \alpha \log \pi'(a|s)\right] - \sum_a \pi(a|s) \cdot \left[Q^{\pi}(s,a) - \alpha \log \pi(a|s)\right] \\
=&\sum_a \left(\pi'(a|s) - \pi(a|s)\right) \cdot \left[Q^{\pi'}(a|s) - \alpha \log \pi'(a|s)\right] \\
&+ \sum_a \pi(a|s) \cdot \left[Q^{\pi'}(s,a) - \alpha \log \pi'(a|s) - Q^{\pi}(s,a) + \alpha \log \pi(a|s)\right] \\
=&\sum_a \left(\pi'(a|s) - \pi(a|s)\right) \cdot \left[Q^{\pi'}(a|s) - \alpha \log \pi'(a|s)\right] + \alpha D_{\mathrm{KL}}\left(\pi(\cdot|s)||\pi'(\cdot|s)\right) \\
&+ \gamma \sum_a \pi(a|s) \sum_{s'} P(s'|s,a) \cdot \left[V^{\pi'}(s') - V^{\pi}(s')\right] \\
=&\frac{1}{1-\gamma} \sum_{s'} d_s^{\pi}(s') \Bigg[\sum_{a'} \left(\pi'(a'|s') - \pi(a'|s')\right) \left[Q^{\pi'}(s',a') - \alpha \log \pi'(a'|s')\right] \\
&+ \alpha D_{\mathrm{KL}}\left(\pi(\cdot|s')||\pi'(\cdot|s')\right) \Bigg],
\end{aligned}
\tag{66}
$$

where the last equality holds because by the definition of state visitation distribution

$$
d_{s_0}^{\pi}(s) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}^{\pi}(s_t = s | s_0), \tag{67}
$$

taking expectation of s with respect to $s \sim \rho$, yields

$$
\begin{aligned}
&V^{\pi'}(\rho) - V^{\pi}(\rho) \\
=&\frac{1}{1-\gamma} \sum_{s'} d_{\rho}^{\pi}(s') \Bigg[\sum_{a'} \left(\pi'(a'|s') - \pi(a'|s')\right) \cdot \left[Q^{\pi'}(s',a') - \alpha \log \pi'(a'|s')\right] \\
&+ \alpha D_{\mathrm{KL}}\left(\pi(\cdot|s')||\pi'(\cdot|s')\right) \Bigg],
\end{aligned}
\tag{68}
$$

and (68) is known as the soft performance difference lemma (Lemma 25 in (Mei et al., 2020)).

Soft sub-optimality. Next we will show the soft sub-optimality result. By the definition of the optimal policy of α-MaxEnt RL (47), we have

$$
\alpha \log \pi^{\star}(a|s) = Q^{\pi^{\star}}(s,a) - V^{\pi^{\star}}(s). \tag{69}
$$

Substituting $\pi^\star$ into the performance difference lemma (68), we have

$$\begin{aligned}
&V^{\pi^\star}(s) - V^{\pi}(s) \\
=&\frac{1}{1-\gamma}\sum_{s'} d_s^{\pi}(s') \cdot \Bigg[\sum_{a'}(\pi^\star(a'|s') - \pi(a'|s')) \cdot \underbrace{\left[Q^{\pi^\star}(s',a') - \alpha \log \pi^\star(a'|s')\right]}_{=V^{\pi^\star}(s')} \\
&+ \alpha D_{\mathrm{KL}}\left(\pi(\cdot|s')||\pi^\star(\cdot|s')\right)\Bigg] \\
=&\frac{1}{1-\gamma}\sum_{s'} d_s^{\pi}(s') \cdot \Bigg[\underbrace{\sum_{a'}(\pi^\star(a'|s') - \pi(a'|s'))}_{=0} \cdot V^{\pi^\star}(s') + \alpha D_{\mathrm{KL}}\left(\pi(\cdot|s')||\pi^\star(\cdot|s')\right)\Bigg] \\
=&\frac{1}{1-\gamma}\sum_{s'} \left[d_s^{\pi}(s') \cdot \alpha D_{\mathrm{KL}}\left(\pi(\cdot|s')||\pi^\star(\cdot|s')\right)\right],
\end{aligned} \tag{70}$$

taking expectation $s \sim \rho$ yields

$$V^{\pi^\star}(\rho) - V^{\pi}(\rho) = \frac{1}{1-\gamma}\sum_{s}\left[d_\rho^{\pi}(s) \cdot \alpha \cdot D_{\mathrm{KL}}\left(\pi(\cdot|s)||\pi^\star(\cdot|s)\right)\right], \tag{71}$$

which completes the proof. □

C. Supporting Lemmas

C.1. Bellman Consistency Equation of MaxEnt RL

Lemma C.1 (Contraction of Soft Value Iteration)**.** *From (48) and (6), the soft value iteration operator $\mathcal{T}$ defined as*

$$\mathcal{T}Q(s,a) := r(s,a) + \gamma\alpha\mathbb{E}_{s'}\left[\log\sum_{a'}\exp\left(Q(s',a')/\alpha\right)\right] \tag{72}$$

is a contraction.

Proof. A similar proof appears in (Haarnoja, 2018), we provide the proof for completeness. To see (72) is a contraction, for each $(s,a) \in \mathcal{S}\times\mathcal{A}$, we have

$$\begin{aligned}
&\mathcal{T}Q_1(s,a) = r(s,a) + \gamma\alpha\log\sum_{a'}\exp\left(\frac{Q_1(s,a)}{\alpha}\right) \\
\leq& r(s,a) + \gamma\alpha\log\sum_{a'}\exp\left(\frac{Q_2(s,a) + \|Q_1 - Q_2\|_\infty}{\alpha}\right) \\
\leq& r(s,a) + \gamma\alpha\log\left\{\exp\left(\frac{\|Q_1 - Q_2\|_\infty}{\alpha}\right)\sum_{a'}\exp\left(\frac{Q_2(s,a)}{\alpha}\right)\right\} \\
=&\gamma\|Q_1 - Q_2\|_\infty + r(s,a) + \gamma\alpha\log\sum_{a'}\exp\left(\frac{Q_2(s,a)}{\alpha}\right) = \gamma\|Q_1 - Q_2\|_\infty + \mathcal{T}Q_2(s,a),
\end{aligned} \tag{73}$$

which implies $\mathcal{T}Q_1(s,a) - \mathcal{T}Q_2(s,a) \leq \gamma\|Q_1 - Q_2\|_\infty$. Similarly, we also have $\mathcal{T}Q_2(s,a) - \mathcal{T}Q_1(s,a) \leq \gamma\|Q_1 - Q_2\|_\infty$, hence we conclude that

$$|Q_1(s,a) - Q_2(s,a)| \leq \gamma\|\mathcal{T}Q_1 - \mathcal{T}Q_2\|_\infty, \ \forall (s,a) \in \mathcal{S}\times\mathcal{A}, \tag{74}$$

which implies $\|Q_1 - Q_2\|_\infty \leq \gamma\|\mathcal{T}Q_1 - \mathcal{T}Q_2\|_\infty$. Hence $\mathcal{T}$ is a γ-contraction and the optimal policy $\pi^\star$ of it is unique. □

C.2. Constant Minimum Policy Probability

Lemma C.2 (Lemma 16 of (Mei et al., 2020)). *Using the policy gradient method (Algorithm 3) with an initial distribution ρ such that $\rho(s) > 0, \forall S$, we have*

$$c := \inf_{t \geq 1} \min_{s,a} \pi_{\theta_t}(a|s) > 0 \tag{75}$$

is a constant that does not depend on t.

Remark C.3 (State Space Dependency of constant c, C_δ^0). *For the exact PG case, c in Lemma C.2 could also depend on S, similarly for the constant C_δ^0 in the stochastic PG case. As pointed out by Li et al. (2021) (Table 1), the constant c (or C_δ^0 in Theorem A.2 of the SPG case) may depend on the structure of the MDP. The ROLLIN technique only improves the mismatch coefficient $\left\| d_\mu^{\pi^\star} / \mu \right\|_\infty$, instead of the constant c (or C_δ^0). Still, in the exact PG case, if one replaces the constant c with other S dependent function $f(S)$, one still can apply a similar proof technique for Theorem 4.1 to show that* ROLLIN *reduces the iteration complexity, and the final iteration complexity bound in Theorem 4.1 will include an additional $f(S)$. In addition, omitting the factor C_δ^0,* ROLLIN *can improve the exponential complexity dependency incurred by the stochastic optimization to a polynomial dependency.*

D. Supporting Algorithms

Algorithm 3 PG for α-MaxEnt RL (Algorithm 1 in (Mei et al., 2020))

Input: ρ, θ_0, $\eta > 0$.
for $t = 0, \ldots, T$ **do**
  $\theta_{t+1} \leftarrow \theta_t + \eta \cdot \frac{\partial V^{\pi_{\theta_t}}(\rho)}{\partial \theta_t}$
end for

Algorithm 4 Two-Phase SPG for α-MaxEnt RL (Algorithm 5.1 in (Ding et al., 2021))

Input: $\rho, \theta_0, \alpha, B_1, B_2, T_1, T, \{\eta_t\}_{t=0}^T$
for $t = 0, 1, \ldots, T$ **do**
  if $t \leq T_1$ **then**
    $B = B_1$ ▷ Phase 1
  else
    $B = B_2$ ▷ Phase 2
  end if
  Run random horizon SPG with $\rho, \alpha, \theta_t, B, t, \eta_t$ ▷ Algorithm 5
end for

Algorithm 5 Random-horizon SPG for α-MaxEnt RL Update (Algorithm 3.2 in (Ding et al., 2021))

Input: $\rho, \alpha, \theta_0, B, t, \eta_t$
for $i = 1, 2, ..., B$ **do**
  $s_{H_t}^i, a_{H_t}^i \leftarrow \mathrm{SamSA}(\rho, \theta_t, \gamma)$ ▷ Algorithm 6
  $\hat{Q}^{\pi_{\theta_t}, i} \leftarrow \mathrm{EstEntQ}(s_{H_t}^i, a_{H_t}^i, \theta_t, \gamma, \alpha)$ ▷ Algorithm 7
end for
$\theta_{t+1} \leftarrow \theta_t + \frac{\eta_t}{(1-\gamma)B} \sum_{i=1}^B \left[\nabla_\theta \log \pi_{\theta_t}(a_{H_t}^i | s_{H_t}^i) \left(\hat{Q}^{\pi_{\theta_t}, i} - \alpha \log \pi_{\theta_t} \right) (a_{H_t}^i | s_{H_t}^i) \right]$

Remark D.1. *Lemma 3.4 in (Ding et al., 2021) implies that the estimator*

$$\frac{1}{(1-\gamma)} \left[\nabla_\theta \log \pi_{\theta_t}(a_{H_t}^i | s_{H_t}^i) \left(\hat{Q}^{\pi_{\theta_t}, i} - \alpha \log \pi_{\theta_t} \right) (a_{H_t}^i | s_{H_t}^i) \right] \tag{76}$$

in line 6 of Algorithm 6 is an unbiased estimator of the gradient $\nabla_\theta V^{\pi_\theta}(\rho)$.

Algorithm 6 SamSA: Sample s, a for SPG (Algorithm 8.1 in (Ding et al., 2021))

Input: ρ, θ, γ
Draw $H \sim \mathrm{Geom}(1-\gamma)$ ▷ $\mathrm{Geom}(1-\gamma)$ geometric distribution with parameter $1-\gamma$
Draw $s_0 \sim \rho, a_0 \sim \pi_\theta(\cdot|s_0)$
for $h = 1, 2, \ldots, H-1$ **do**
    Draw $s_{h+1} \sim \mathbb{P}(\cdot|s_h, a_h), a_{h+1} \sim \pi_{\theta_t}(\cdot|s_{h+1})$
end for
Output: s_H, a_H

Algorithm 7 EstEntQ: Unbiased Estimation of MaxEnt Q (Algorithm 8.2 in (Ding et al., 2021))

Input: $s, a, \theta, \gamma, \alpha$
Initialize $s_0 \leftarrow s, a_0 \leftarrow a, \hat{Q} \leftarrow r(s_0, a_0)$
Draw $H \sim \mathrm{Geom}(1-\gamma)$
for $h = 0, 1, \ldots, H-1$ **do**
    $s_{h+1} \sim \mathbb{P}(\cdot|s_h, a_h), a_{h+1} \sim \pi_\theta(\cdot|s_{h+1})$
    $\hat{Q} \leftarrow \hat{Q} + \gamma^{h+1}/2\left[r(s_{h+1}, a_{h+1}) - \alpha \log \pi_\theta(a_{h+1}|s_{h+1})\right]$
end for
Output: $\hat{Q}$

E. Experimental Details

We use the SAC implementation from `https://github.com/ikostrikov/jaxrl` (Kostrikov, 2021) for all our experiments in the paper.

E.1. Goal Reaching with an Oracle Curriculum

For our `antmaze-umaze` experiments with oracle curriculum, we use a sparse reward function where the reward is 0 when the distance D between the ant and the goal is greater than 0.5 and $r = \exp(-5D)$ when the distance is smaller than or equal to 0.5. The performance threshold is set to be $R = 200$. Exceeding such threshold means that the ant stays on top of the desired location for at least 200 out of 500 steps, where 500 is the maximum episode length of the `antmaze-umaze` environment. We use the average return of the last 10 episodes and compare it to the performance threshold R. For both of the SAC agents, we use the same set of hyperparameters shown in Table 4. See Algorithm 8, for a more detailed pseudocode.

Algorithm 8 Practical Implementation of ROLLIN

Input: $\{\omega_k\}_{k=0}^{K}$: input curriculum, ρ: initial state distribution, R: near-optimal threshold, β: roll-in ratio, discount factor γ.
Initialize $\mathcal{D} \leftarrow \emptyset, \mathcal{D}_{\text{exp}} \leftarrow \emptyset, k \leftarrow 0$, and two off-policy RL agents π_{main} and π_{exp}.
for each environment step **do**
  if episode terminating or beginning of training **then**
    if average return of the last 10 episodes under context ω_k is greater than R **then**
      $k \leftarrow k + 1, \mathcal{D}_{\text{exp}} \leftarrow \emptyset$
      Re-initialize the exploration agent π_{exp}
    end if
    Start a new episode under context ω_k with $s_0 \sim \rho, t \leftarrow 0$
    if $k > 0$ and with probability of β **then**
      enable **Rollin** for the current episode.
    else
      disable **Rollin** for the current episode.
    end if
  end if
  if Rollin is enabled for the current episode **then**
    if Rollin is stopped for the current episode **then**
      $a_t \sim \pi_{\text{exp}}(a_t|s_t, \omega_k)$
    else
      $a_t \sim \pi_{\text{main}}(a_t|s_t, \omega_{k-1})$
      with probability of $1 - \gamma$, stop **Rollin** for the current episode
    end if
  else
    $a_t \sim \pi_{\text{main}}(a_t|s_t, \omega_k)$
  end if
  take action a_t in the environment and receives s_{t+1} and $r_t = r_{\omega_k}(s_t, a_t)$
  add (s_t, a_t, s_{t+1}, r_t) in replay buffer $\mathcal{D}$
  if Rollin is disabled for the current episode **then**
    update π_{main} using $\mathcal{D}$.
  end if
  if π_{exp} was used to produce a_t **then**
    add (s_t, a_t, s_{t+1}, r_t) in replay buffer $\mathcal{D}_{\text{exp}}$
    update π_{exp} using $\mathcal{D}_{\text{exp}}$.
  end if
  $t \leftarrow t + 1$
end for
Output: π_{main}

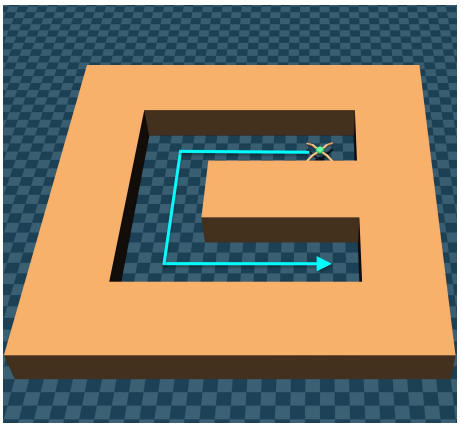


Figure 3: **Oracle curriculum of desired goals on** `antmaze-umaze`. The ant starts from the right top corner and the farthest goal is located at the bottom right corner.

Initial Temperature		1.0
Target Update Rate	update rate of target networks	0.005
Learning Rate	learning rate for the Adam optimizer	0.0003
Discount Factor		0.99
Batch Size		256
Warmup Period	number of steps of initial random exploration (random actions)	10000
Network Size		$(256, 256)$

Table 4: Hyperparameters used for the SAC algorithm (Haarnoja et al., 2018)

E.2. Non-Goal Reaching

For the non goal reaching tasks in `walker2d`, `hopper`, `humanoid`, and `ant` experiments, the desired x-velocity range $[\lambda\kappa, \lambda(\kappa + 0.1))$, the near-optimal threshold $R(\kappa)$, and the `healthy_reward` all depend on the environments. The maximum episode length 1000. Details are provided in Table 5.

			`healthy_reward`		
`Env.`	λ	$R(\kappa)$	original	`high`	`low`
`walker`	5	$500 + 4500\kappa$	1.0	1.5	0.5
`hopper`	3	$500 + 4500\kappa$	1.0	1.5	0.5
`humanoid`	1	$2500 + 2500\kappa$	5.0	7.5	2.5
`ant`	6	$500 + 4500\kappa$	1.0	1.5	0.25

Table 5: **Learning progress κ, average x-velocity, and average return at the 0.75 and 1.0 million environment steps in `walker`, `hopper`, `humanoid`, and `ant`.** The average x-velocity and return are estimated using the last 50k time steps. We pick $\beta = 0.1$ for all experiments using ROLLIN, the results of using other βs can be found in Table 11, Table 12, and Table 13 in Appendix G.2. The standard error is computed over 8 random seeds.

F. Numerical Experiments: The Four Room Navigation

F.1. MDP Setup

The grid world consists of 12×12 grid cells where each cell corresponds to a state in the MDP. The agent can take four different actions to move itself in four directions to a different cell or take a fifth action to receive reward (positive if close to the goal, 0 otherwise). Each context in the context space represents a distinct goal state in the grid-world. The agent (when taking the fifth action) receives higher reward the closer it is to the goal state and receives 0 reward when it is too far (4 steps away for easy, and 5 steps away for hard). We also include 100 additional dummy actions in the action space (taking these actions do not result in reward nor state changes) to make the exploration problem challenging. See Figure 2 for a visualization of the environment and the two reward functions we use. More concretely, let $D(s, g)$ be the number of action it takes to go from state s to state g (the current goal) if the walls did not exist (the Manhattan distance), the reward received when taking the fifth action at state s is

$$r_{\text{four\_room}}(s) = \begin{cases} \gamma_{\text{reward}}^{D(s,g)}, & D(s,g) \leq D_{\text{threshold}} \\ 0, & D(s,g) > D_{\text{threshold}} \end{cases}$$

For the easy reward function, $\gamma_{\text{reward}} = 0.9$, $D_{\text{threshold}} = 5$. For the hard reward function, $\gamma_{\text{reward}} = 0.5$, $D_{\text{threshold}} = 4$.

F.2. Pre-defined curriculum.

Our curriculum contains 16 contexts in sequence, $\{\omega_k\}_{k=0}^{16}$, which form a continuous path from the start location of the agent $(0, 0)$ to the goal location of the agent at $(8, 8)$. We use a fixed success rate threshold (an episode is considered to be successful if the agent reaches the goal state and perform the fifth action at that goal state) to determine convergence of the stochastic PG algorithm. We switch to the next context/curriculum step whenever the success rate exceeds 50%. We use $\kappa \in [0, 1]$ to denote a normalized curriculum progress which is computed as the current curriculum step index divided by the total number of curriculum steps.

F.3. Stochastic PG description

We follow Algorithm 1 closely for our implementation. In particular, we adopt the softmax parameterization of π that is parameterized by $\theta \in \mathbb{R}^{S \times A}$ as $\pi_\theta(a = j|s = i) = \frac{\exp(\theta_{ij})}{\sum_{j'} \exp(\theta_{ij'})}$ with $i \in [S]$ and $j \in [A]$ (in this MDP, $S = 144$ as there are $12 \times 12 = 144$ cells in the grid world and $A = 105$ due to the dummy actions). To sample from $d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}$ (Line 6), we rollout the policy from the previous context $\pi_{\theta_{k-1}}$ in the MDP for h steps (where h being sampled from $\text{Geom}(1 - \gamma)$, the geometric distribution with a rate of $1 - \gamma$) and take the resulting state as a sample from $d_{\mu_{k-1}}^{\pi^\star_{\omega_{k-1}}}$. We implement the stochastic PG using Adam optimizer (Kingma and Ba, 2015) on θ with a constant learning rate of 0.001. Every gradient step is computed over 2000 trajectories with the trajectory length capped at 50 for each. The empirical policy gradient for π_θ is computed as over $B = 2000$ trajectories ($\{(s_0^b, a_0^b, s_1^b, \cdots, s_T^b\}_{b=1}^B$) and $T = 50$ time steps collected by rolling out the current policy π_θ: $\frac{1}{BT} \sum_b \sum_t \nabla_\theta \log \pi_\theta(a_t^b|s_t^b) R_t^b$ with $R_t^b = -\sum_{t'=t}^T \gamma^{t'-t} r_{\text{ent},t'}^b$, $r_{\text{ent},t}^b = r(s_t^b, a_t^b) - \alpha \log \pi(a_t^b|s_t^b)$ where R_t^b is the Monte-Carlo estimate of the discounted, entropy-regularized cumulative reward.

F.4. Results

Setting	Entropy Coefficient	$\beta = 0.0$ (Baseline)	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.3$	$\beta = 0.5$	$\beta = 0.75$	$\beta = 0.9$
Hard	$\alpha = 0.01$	0.500 ± 0.000	0.506 ± 0.001	0.512 ± 0.001	0.525 ± 0.000	0.562 ± 0.000	0.562 ± 0.000	0.562 ± 0.000
	$\alpha = 0.001$	0.856 ± 0.006	0.981 ± 0.003	0.981 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
Easy	$\alpha = 0.01$	0.944 ± 0.003	0.994 ± 0.001	0.994 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	$\alpha = 0.001$	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

Table 6: **Curriculum progress κ on the four-room navigation with stochastic PG at step 50,000.** We tested with two different entropy coefficients and seven different β's. The standard error is computed over 10 random seeds.

Setting	Entropy Coefficient	$\beta = 0.0$ (Baseline)	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.3$	$\beta = 0.5$	$\beta = 0.75$	$\beta = 0.9$
Hard	$\alpha = 0.01$	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
	$\alpha = 0.001$	0.424 ± 0.023	0.939 ± 0.0014	0.710 ± 0.021	1.010 ± 0.005	1.062 ± 0.000	1.067 ± 0.000	1.060 ± 0.000
Easy	$\alpha = 0.01$	4.093 ± 0.224	5.136 ± 0.230	4.156 ± 0.228	1.040 ± 0.140	3.913 ± 0.218	7.374 ± 0.216	4.227 ± 0.232
	$\alpha = 0.001$	10.536 ± 0.002	10.566 ± 0.003	10.602 ± 0.003	10.593 ± 0.002	10.611 ± 0.002	10.620 ± 0.002	10.575 ± 0.002

Table 7: **Final return V^π on the four-room navigation with stochastic PG at step 50,000.** We tested with two different entropy coefficients and seven different β's. The standard error is computed over 10 random seeds.

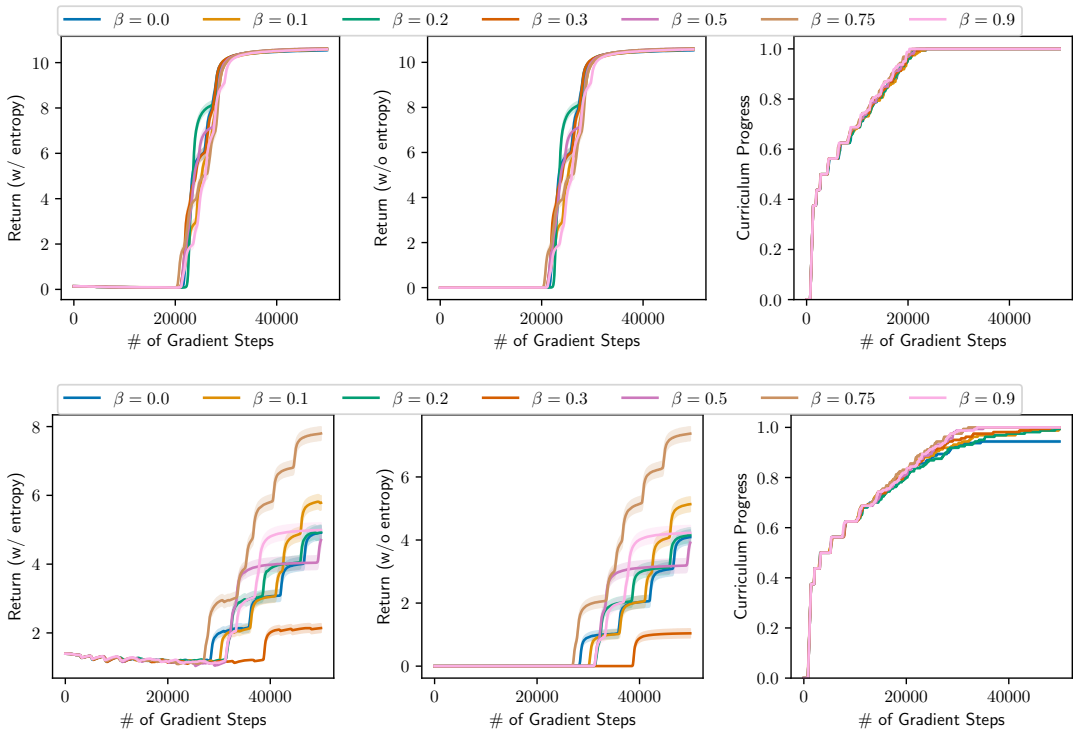


Figure 4: **Learning curves for the numerical experiments on the easy curriculum.**

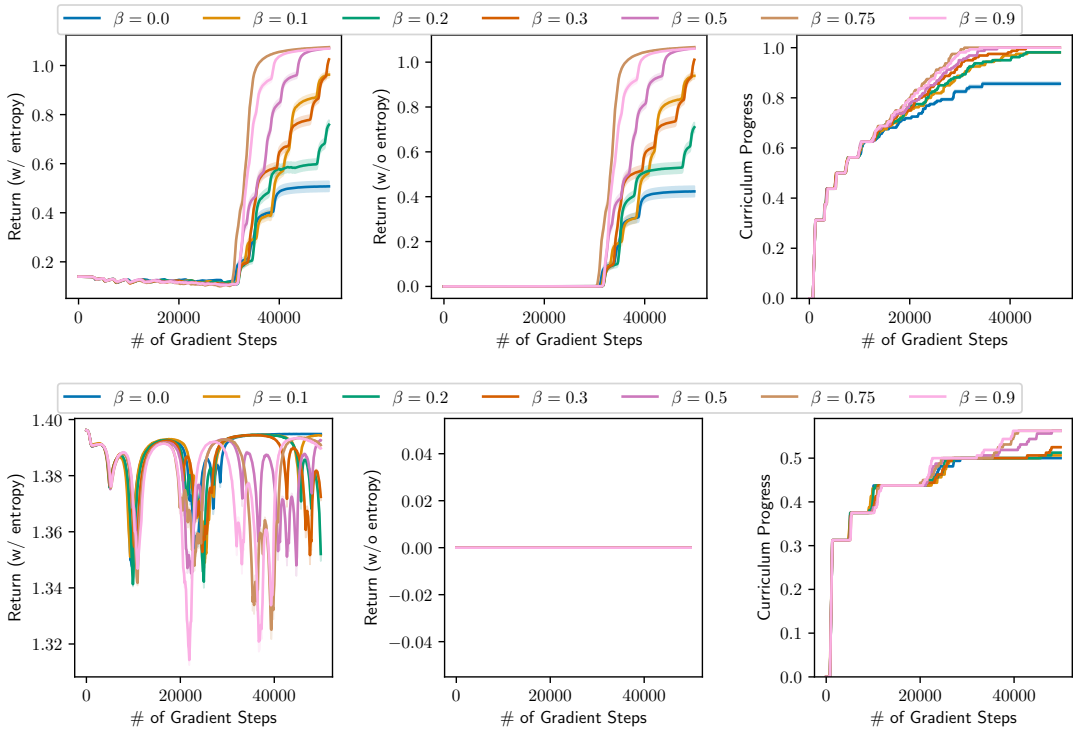
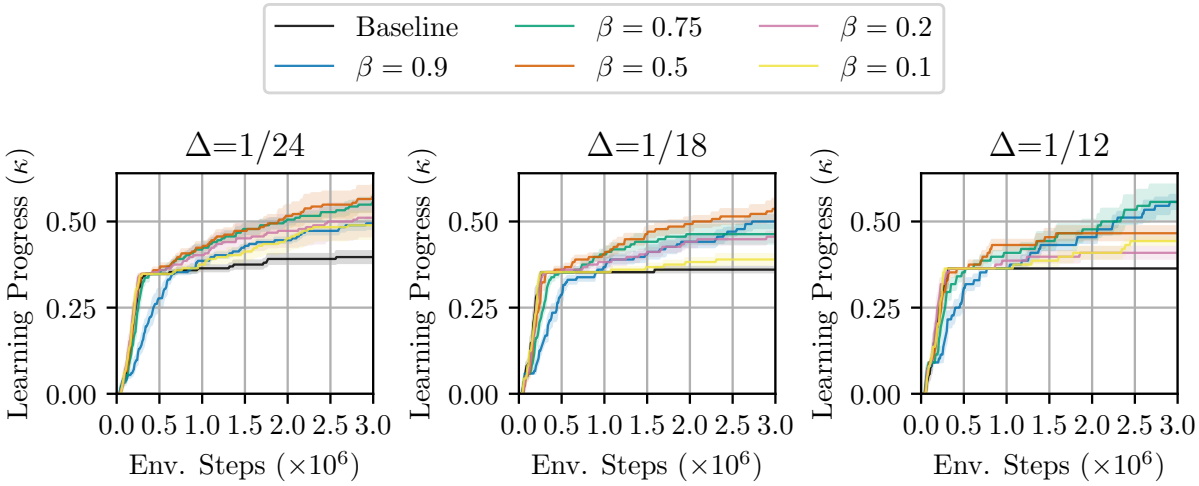


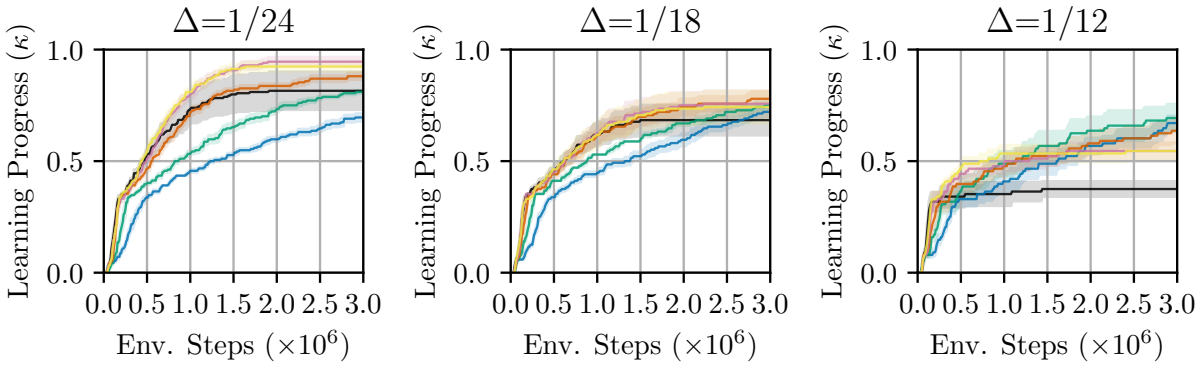
Figure 5: **Learning curves for the numerical experiments on the hard curriculum.**

G. Additional Learning Curves and Tables

G.1. Goal Reaching

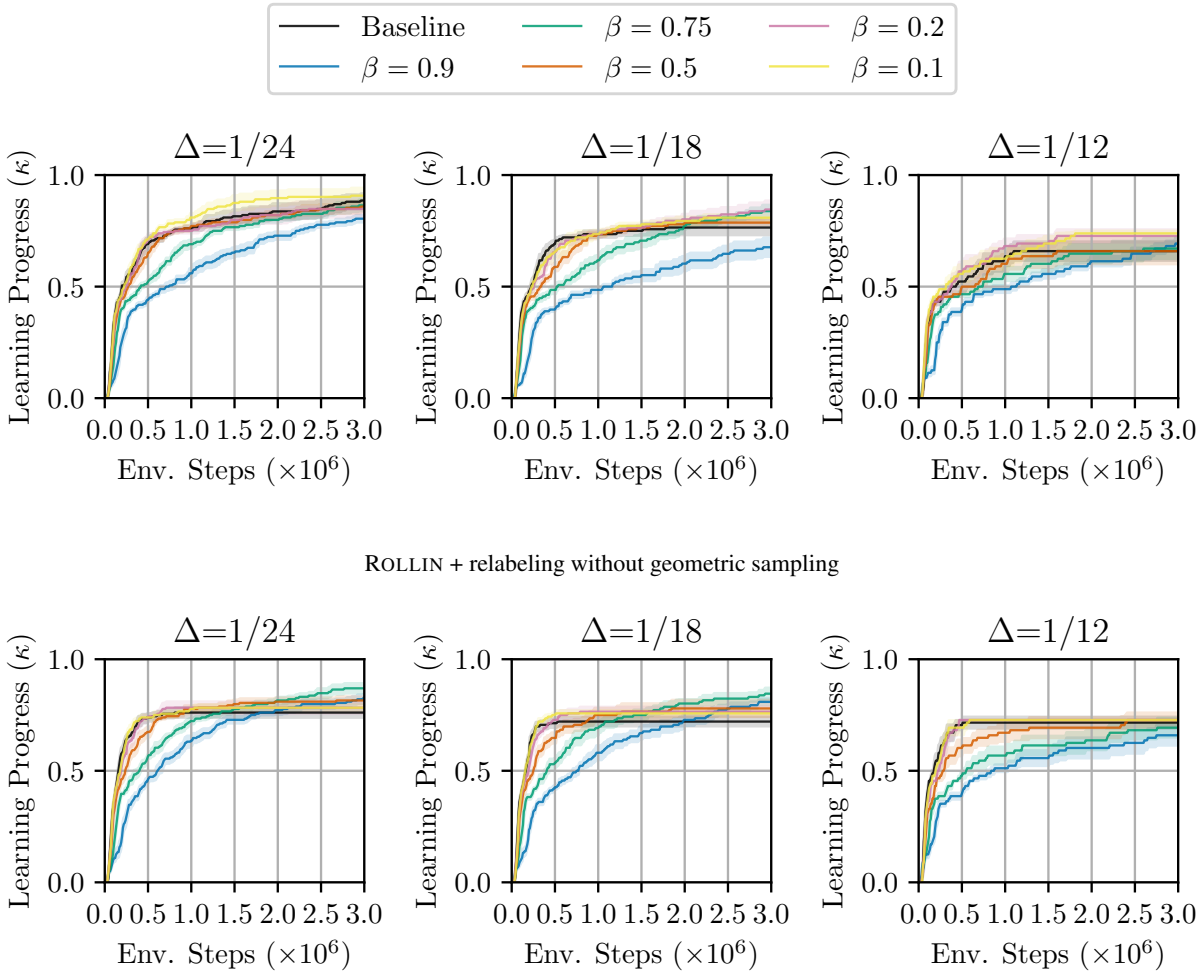


Vanilla ROLLIN without geometric sampling



Vanilla ROLLIN with geometric sampling

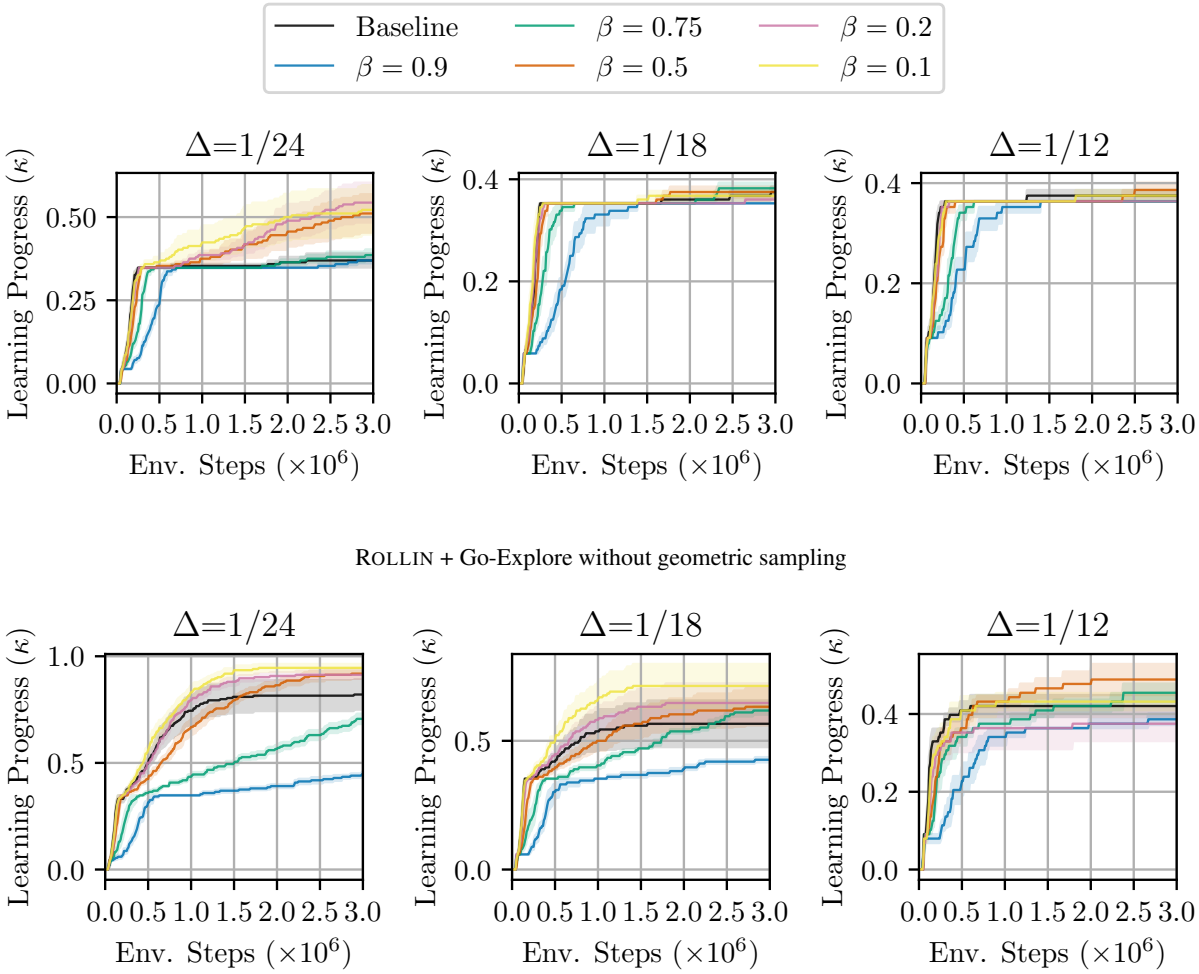
Figure 6: **Vanilla Goal reaching.** Accelerating learning on `antmaze-umaze` with ROLLIN on an oracle curriculum in Figure 3. The confidence interval represents the standard error computed over 8 random seeds.



ROLLIN + relabeling without geometric sampling

ROLLIN + relabeling with geometric sampling

Figure 7: **Goal relabeling.** Accelerating learning on `antmaze-umaze` with ROLLIN on an oracle curriculum in Figure 3. The confidence interval represents the standard error computed over 8 random seeds.



ROLLIN + Go-Explore without geometric sampling

ROLLIN + Go-Explore with geometric sampling

Figure 8: **Go-Explore (exploration noise = 0.1).** Accelerating learning on `antmaze-umaze` with ROLLIN on an oracle curriculum in Figure 3. The confidence interval represents the standard error computed over 8 random seeds.

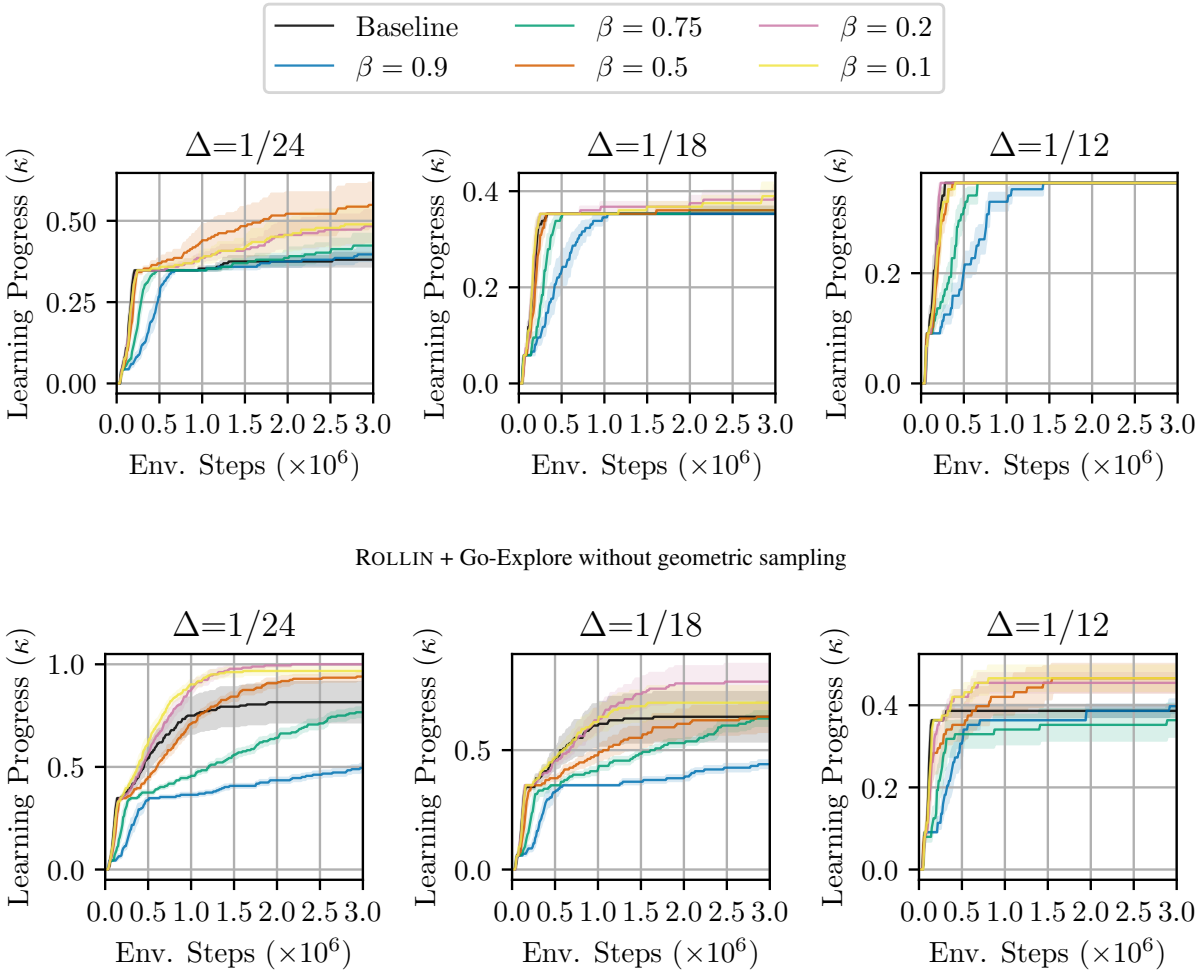
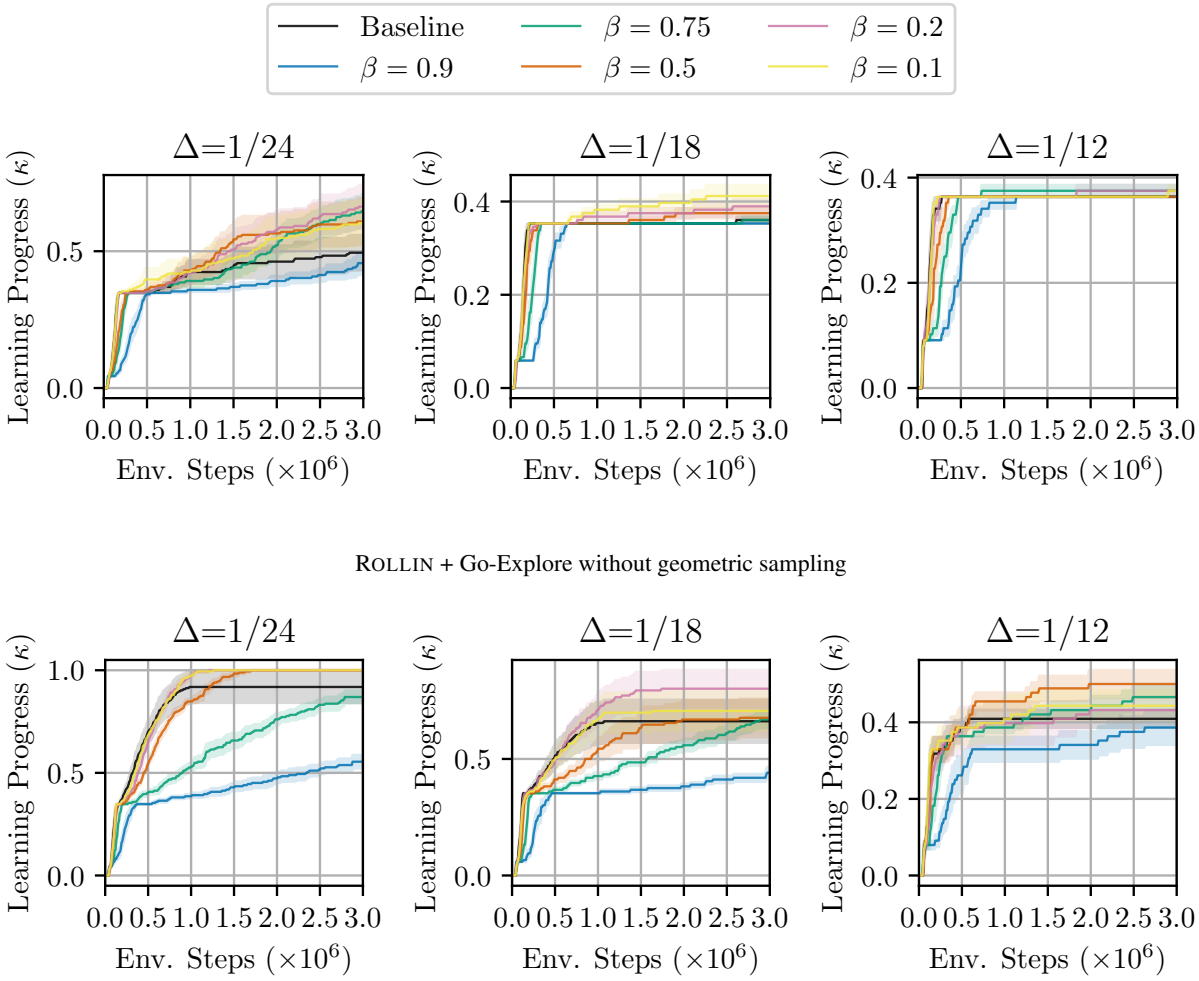


Figure 9: **Go-Explore (exploration noise = 0.25).** Accelerating learning on `antmaze-umaze` with ROLLIN on an oracle curriculum in Figure 3. The confidence interval represents the standard error computed over 8 random seeds.



ROLLIN + Go-Explore without geometric sampling

ROLLIN + Go-Explore with geometric sampling

Figure 10: **Go-Explore (exploration noise = 0.5).** Accelerating learning on `antmaze-umaze` with ROLLIN on an oracle curriculum in Figure 3. The confidence interval represents the standard error computed over 8 random seeds.

Geo	Δ	Baseline	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.5$	$\beta = 0.75$	$\beta = 0.9$
✗	1/24	0.40 ± 0.02	0.49 ± 0.04	0.51 ± 0.05	$\mathbf{0.57 \pm 0.04}$	0.55 ± 0.02	0.49 ± 0.02
✗	1/18	0.36 ± 0.01	0.39 ± 0.01	0.46 ± 0.01	$\mathbf{0.54 \pm 0.03}$	0.46 ± 0.03	0.50 ± 0.02
✗	1/12	0.36 ± 0.00	0.44 ± 0.01	0.41 ± 0.02	0.47 ± 0.02	$\mathbf{0.56 \pm 0.05}$	$\mathbf{0.56 \pm 0.02}$
✓	1/24	0.82 ± 0.08	0.92 ± 0.02	$\mathbf{0.95 \pm 0.02}$	0.88 ± 0.01	0.81 ± 0.01	0.70 ± 0.02
✓	1/18	0.68 ± 0.07	0.74 ± 0.07	0.76 ± 0.06	$\mathbf{0.78 \pm 0.03}$	0.75 ± 0.02	0.72 ± 0.02
✓	1/12	0.38 ± 0.03	0.55 ± 0.04	0.55 ± 0.04	0.64 ± 0.06	$\mathbf{0.69 \pm 0.06}$	0.67 ± 0.03

Table 8: **Vanilla Goal reaching.** Learning progress κ at 3 million environment steps with varying β and curriculum step size Δ of vanilla goal reaching task. Geo indicates the usage of geometric sampling. Baseline corresponds to $\beta = 0$, where no ROLLIN is used. The standard error is computed over 8 random seeds. We highlight the values that are larger than the baseline ($\beta = 0$) in purple, and the largest value in **bold font**.

Geo	Δ	$\beta = 0$	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.5$	$\beta = 0.75$	$\beta = 0.9$
✗	1/24	0.89 ± 0.03	$\mathbf{0.91 \pm 0.03}$	0.85 ± 0.04	0.86 ± 0.02	0.86 ± 0.02	0.80 ± 0.02
✗	1/18	0.76 ± 0.03	0.81 ± 0.01	$\mathbf{0.85 \pm 0.04}$	0.79 ± 0.01	0.84 ± 0.03	0.68 ± 0.04
✗	1/12	0.66 ± 0.04	$\mathbf{0.74 \pm 0.01}$	0.73 ± 0.03	0.66 ± 0.06	0.67 ± 0.05	0.69 ± 0.03
✓	1/24	0.76 ± 0.02	0.78 ± 0.01	0.78 ± 0.03	0.82 ± 0.02	$\mathbf{0.87 \pm 0.02}$	0.83 ± 0.02
✓	1/18	0.72 ± 0.02	0.76 ± 0.01	0.76 ± 0.02	0.78 ± 0.04	$\mathbf{0.85 \pm 0.03}$	0.81 ± 0.01
✓	1/12	0.72 ± 0.03	$\mathbf{0.73 \pm 0.00}$	$\mathbf{0.73 \pm 0.00}$	$\mathbf{0.73 \pm 0.03}$	0.69 ± 0.04	0.66 ± 0.04

Table 9: **Goal relabeling.** All other settings are the same as Table 8.

EN	Geo	Δ	$\beta = 0$	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.5$	$\beta = 0.75$	$\beta = 0.9$
0.1	✗	1/24	0.37 ± 0.02	0.52 ± 0.07	$\mathbf{0.54 \pm 0.06}$	0.51 ± 0.06	0.39 ± 0.02	0.37 ± 0.01
0.1	✗	1/18	0.38 ± 0.01	0.37 ± 0.01	0.36 ± 0.01	0.38 ± 0.01	0.38 ± 0.01	0.35 ± 0.00
0.1	✗	1/12	0.38 ± 0.01	0.38 ± 0.01	0.36 ± 0.00	$\mathbf{0.39 \pm 0.01}$	0.36 ± 0.00	0.36 ± 0.00
0.1	✓	1/24	0.82 ± 0.07	$\mathbf{0.95 \pm 0.02}$	0.91 ± 0.02	0.92 ± 0.02	0.71 ± 0.02	0.45 ± 0.01
0.1	✓	1/18	0.57 ± 0.09	$\mathbf{0.71 \pm 0.08}$	0.65 ± 0.07	0.63 ± 0.07	0.62 ± 0.02	0.43 ± 0.01
0.1	✓	1/12	0.42 ± 0.03	0.43 ± 0.02	0.38 ± 0.04	$\mathbf{0.49 \pm 0.04}$	0.45 ± 0.02	0.39 ± 0.01
0.25	✗	1/24	0.38 ± 0.02	0.49 ± 0.06	0.48 ± 0.05	$\mathbf{0.55 \pm 0.07}$	0.43 ± 0.04	0.40 ± 0.02
0.25	✗	1/18	0.35 ± 0.00	$\mathbf{0.39 \pm 0.03}$	$\mathbf{0.39 \pm 0.02}$	0.36 ± 0.01	0.36 ± 0.01	0.35 ± 0.00
0.25	✗	1/12	0.36 ± 0.00	0.36 ± 0.00	0.36 ± 0.00	0.36 ± 0.00	0.36 ± 0.00	0.36 ± 0.00
0.25	✓	1/24	0.82 ± 0.10	0.97 ± 0.02	$\mathbf{1.00 \pm 0.00}$	0.94 ± 0.02	0.77 ± 0.02	0.49 ± 0.02
0.25	✓	1/18	0.64 ± 0.10	0.70 ± 0.07	$\mathbf{0.79 \pm 0.07}$	0.64 ± 0.06	0.63 ± 0.03	0.44 ± 0.01
0.25	✓	1/12	0.39 ± 0.01	$\mathbf{0.47 \pm 0.03}$	0.45 ± 0.02	$\mathbf{0.47 \pm 0.03}$	0.36 ± 0.04	0.40 ± 0.02
0.5	✗	1/24	0.49 ± 0.06	0.60 ± 0.08	$\mathbf{0.66 \pm 0.08}$	0.61 ± 0.08	0.65 ± 0.06	0.46 ± 0.04
0.5	✗	1/18	0.36 ± 0.01	$\mathbf{0.41 \pm 0.02}$	0.39 ± 0.01	0.38 ± 0.01	0.37 ± 0.01	0.35 ± 0.00
0.5	✗	1/12	0.36 ± 0.00	$\mathbf{0.38 \pm 0.01}$	$\mathbf{0.38 \pm 0.01}$	0.36 ± 0.00	$\mathbf{0.38 \pm 0.01}$	0.36 ± 0.00
0.5	✓	1/24	0.92 ± 0.08	$\mathbf{1.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$	0.87 ± 0.03	0.55 ± 0.03
0.5	✓	1/18	0.66 ± 0.09	0.71 ± 0.08	$\mathbf{0.80 \pm 0.08}$	0.68 ± 0.08	0.67 ± 0.04	0.44 ± 0.02
0.5	✓	1/12	0.41 ± 0.02	0.44 ± 0.04	0.43 ± 0.03	$\mathbf{0.50 \pm 0.04}$	0.47 ± 0.03	0.39 ± 0.04

Table 10: **Go-Explore with different exploration noise.** EN represents the multiplier for the Gaussian exploration noise. All other settings are the same as Table 8.

G.2. Non Goal Reaching Tasks

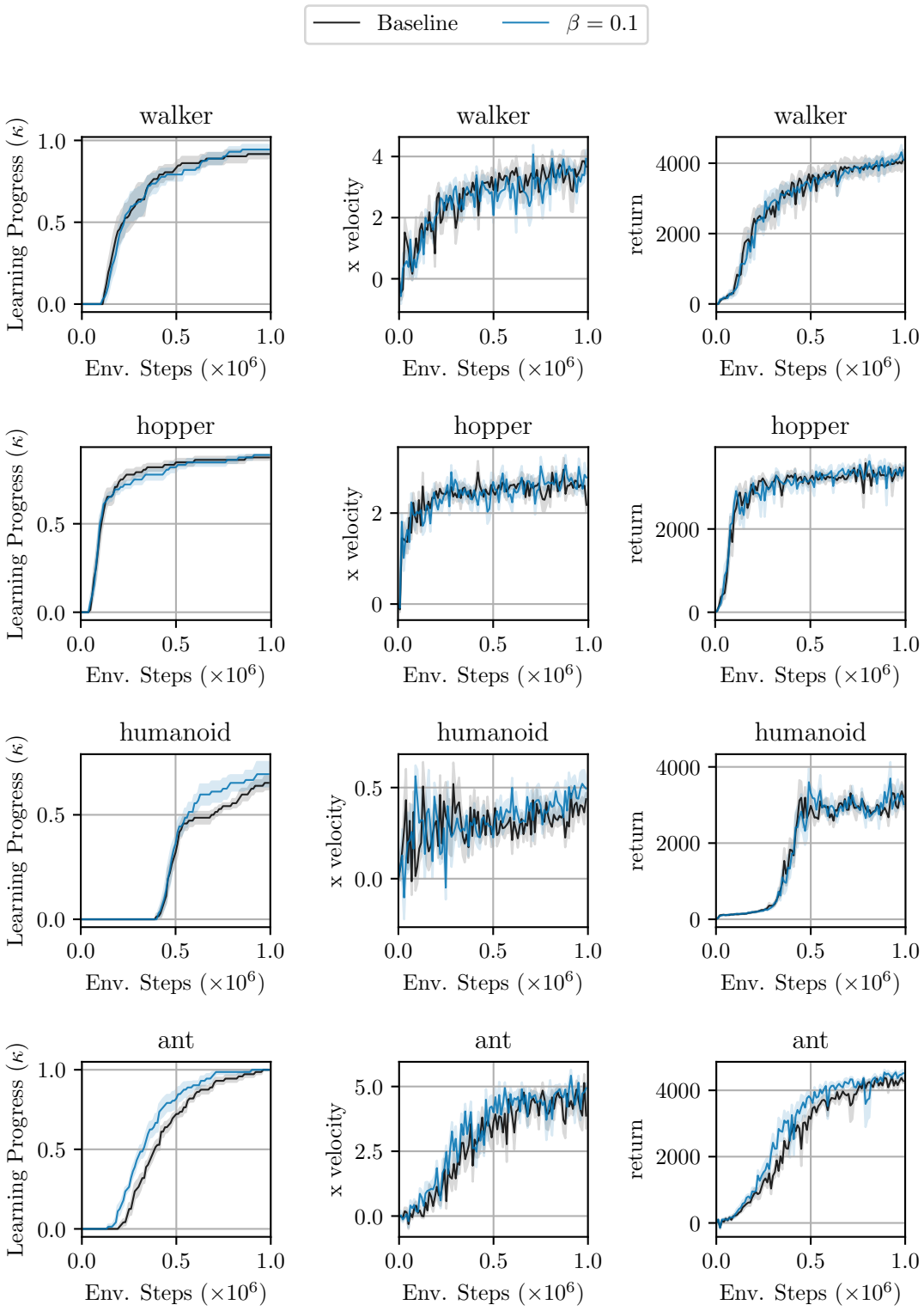


Figure 11: **Accelerating learning on several non goal-reaching tasks.** The confidence interval represents the standard error computed over 8 random seeds, for $\beta = 0.1$.

Env.	Step	$\beta = 0$	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.5$	$\beta = 0.75$
walker	0.5m	0.83 ± 0.03	0.79 ± 0.04	0.75 ± 0.04	0.78 ± 0.05	0.76 ± 0.05
	1m	0.92 ± 0.03	$\mathbf{0.94 \pm 0.03}$	0.90 ± 0.01	0.92 ± 0.04	0.92 ± 0.03
hopper	0.5m	0.85 ± 0.02	0.82 ± 0.03	0.83 ± 0.02	0.78 ± 0.02	0.75 ± 0.02
	1m	0.88 ± 0.01	$\mathbf{0.89 \pm 0.00}$	$\mathbf{0.89 \pm 0.03}$	0.82 ± 0.02	0.81 ± 0.02
humanoid	0.5m	0.32 ± 0.05	$\mathbf{0.36 \pm 0.04}$	0.21 ± 0.07	0.33 ± 0.04	0.14 ± 0.06
	1m	0.67 ± 0.03	0.69 ± 0.06	0.62 ± 0.02	$\mathbf{0.76 \pm 0.03}$	0.71 ± 0.06
ant	0.5m	0.72 ± 0.02	$\mathbf{0.82 \pm 0.06}$	0.68 ± 0.08	0.64 ± 0.05	0.47 ± 0.05
	1m	1.00 ± 0.00	1.00 ± 0.00	0.83 ± 0.08	0.86 ± 0.06	0.71 ± 0.07

Table 11: **Learning progress κ at 0.5 and 1.0 million environment steps with varying β of non goal reaching tasks.** Baseline corresponds to $\beta = 0$, where no ROLLIN is used. The standard error is computed over 8 random seeds. We highlight the values that are larger than the baseline ($\beta = 0$) in purple, and the largest value in **bold font**.

Env.	Step	$\beta = 0$	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.5$	$\beta = 0.75$
walker	0.5m	3.09 ± 0.31	2.83 ± 0.31	2.41 ± 0.33	2.77 ± 0.31	2.88 ± 0.32
	1m	3.69 ± 0.27	3.62 ± 0.26	3.09 ± 0.28	3.48 ± 0.27	3.14 ± 0.34
hopper	0.5m	2.42 ± 0.18	2.26 ± 0.22	$\mathbf{2.45 \pm 0.14}$	2.34 ± 0.16	2.34 ± 0.16
	1m	2.58 ± 0.16	$\mathbf{2.65 \pm 0.15}$	$\mathbf{2.65 \pm 0.17}$	2.39 ± 0.18	2.52 ± 0.19
humanoid	0.5m	0.26 ± 0.05	0.32 ± 0.07	0.27 ± 0.05	0.34 ± 0.05	$\mathbf{0.38 \pm 0.07}$
	1m	0.39 ± 0.05	0.46 ± 0.09	0.41 ± 0.05	0.41 ± 0.06	$\mathbf{0.49 \pm 0.10}$
ant	0.5m	3.38 ± 0.43	$\mathbf{3.85 \pm 0.41}$	3.43 ± 0.53	3.15 ± 0.45	2.38 ± 0.46
	1m	4.29 ± 0.51	$\mathbf{4.66 \pm 0.30}$	3.93 ± 0.45	3.99 ± 0.48	3.50 ± 0.49

Table 12: **Average x-direction velocity of the last 50k time steps, at 0.5 and 1.0 million environment steps with varying β of non goal reaching tasks.** Baseline corresponds to $\beta = 0$, where no ROLLIN is used. The standard error is computed over 8 random seeds. We highlight the values that are larger than the baseline ($\beta = 0$) in purple, and the largest value in **bold font**.

Env.	Step	$\beta = 0$	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.5$	$\beta = 0.75$
walker	0.5m	3450.1 ± 307.4	3350.4 ± 184.6	2897.4 ± 276.5	3255.9 ± 203.8	3185.8 ± 341.5
	1m	4032.3 ± 224.3	$\mathbf{4128.8 \pm 159.6}$	3685.5 ± 135.6	4028.8 ± 164.2	3895.4 ± 265.4
hopper	0.5m	3192.5 ± 80.4	3148.6 ± 160.7	$\mathbf{3241.5 \pm 130.8}$	3116.5 ± 141.8	3059.6 ± 153.8
	1m	3386.2 ± 124.7	$\mathbf{3421.9 \pm 109.8}$	3262.3 ± 98.1	3170.7 ± 180.6	3394.5 ± 126.5
humanoid	0.5m	2910.1 ± 262.9	2939.7 ± 392.0	2598.9 ± 309.8	$\mathbf{3137.3 \pm 305.6}$	2259.6 ± 245.4
	1m	3017.2 ± 169.0	3173.6 ± 238.3	2935.8 ± 181.1	2905.5 ± 125.9	$\mathbf{3290.7 \pm 275.9}$
ant	0.5m	2976.2 ± 252.4	$\mathbf{3593.1 \pm 237.8}$	3071.8 ± 340.0	2818.3 ± 265.2	2188.3 ± 256.2
	1m	4248.5 ± 88.6	$\mathbf{4473.0 \pm 102.2}$	3683.1 ± 345.0	3708.7 ± 290.5	3250.1 ± 316.2

Table 13: **Average return of the last 50k time steps, at the 0.5 and 1.0 million environment steps with varying β of non goal reaching tasks.** Baseline corresponds to $\beta = 0$, where no ROLLIN is used. The standard error is computed over 8 random seeds. We highlight the values that are larger than the baseline ($\beta = 0$) in purple, and the largest value in **bold font**.