Langevin Policy for Safe Reinforcement Learning

Fenghao Lei [1] **Long Yang** [2] **Shiting Wen** [3] **Zhixiong Huang** [1] **Zhiwang Zhang** [3] **Chaoyi Pang** [3]

Abstract

Optimization and sampling based algorithms are two branches of methods in machine learning, while existing safe reinforcement learning (RL) algorithms are mainly based on optimization, it is still unclear whether sampling based methods can lead to desirable performance with safe policy. This paper formulates the Langevin policy for safe RL, and proposes Langevin Actor-Critic (LAC) to accelerate the process of policy inference. Concretely, instead of parametric policy, the proposed Langevin policy provides a stochastic process that directly infers actions, which is the numerical solver to the Langevin dynamic of actions on the continuous time. Furthermore, to make Langevin policy practical on RL tasks, the proposed LAC accumulates the transitions induced by Langevin policy and reproduces them with a generator. Finally, extensive empirical results show the effectiveness and superiority of LAC on the MuJoCo-based and Safety Gym tasks. Our implementation is available at https://github.com/Lfh404/LAC.

1. Introduction

Reinforcement learning (RL) has significantly boosted its utilization in many challenging tasks, e.g., playing games (Silver et al., 2017; Mnih et al., 2015), robot control (Schulman et al., 2015), and discovering algorithms (Mankowitz et al., 2023; Fawzi et al., 2022). However, when considering the deployment of RL in the real world such as autonomous driving and surgical robots, ensuring safety is a critical factor. These applications require careful consideration of constraints to mitigate the risk of causing harm to humans.

[1]College of Computer Science and Technology, Zhejiang University, Hangzhou, China [2]School of Artificial Intelligence, Peking University, Beijing, China [3]School of Computer and Data Engneering, NingboTech University, Ningbo, China. Correspondence to: Shiting Wen <wensht@nit.zju.edu.cn>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

Existing safe RL algorithms are classified into the following categories: surrogate function methods (e.g, CPO (Achiam et al., 2017), FOCOPS(Zhang et al., 2020), CUP(Yang et al., 2022)), primal-dual methods (e.g., RCPO (Tessler et al., 2018), PDO (Chow et al., 2017), RPD-PG(Ding et al., 2023)), Lyapunov-based method (Chow et al., 2018), and Barrier function method (Luo & Ma, 2021). All of the above methods are based on optimization, which formulates the policy as a parametric function. Optimization and Monte Carlo sampling are the two main classes of algorithms to solve machine learning problems, but it is still unknown whether Monte Carlo sampling works well for RL or not. More seriously, the folk wisdom is that Monte Carlo sampling is necessarily inefficient than optimization based methods and is only warranted in situations where estimates of uncertainty are needed (Ma et al., 2019). However, recent sampling based methods (e.g., diffusion model (Ho et al., 2020), noise conditional score network (NCSN) (Song & Ermon, 2019)) show its great success in a variety of domains (e.g., image synthesis (Ramesh et al., 2022), video generation (Ramesh et al., 2022)). It is still unclear whether Monte Carlo sampling based methods can lead to desirable performance in exploring safe policy, which is the main focus of this paper.

This paper considers the safe RL under the background of constrained policy optimization (CPO) (Achiam et al., 2017). Zhang et al. (2020) shows the optimal update policy of CPO as an energy-based model. In this work, we seek to solve safe RL problem from a sampling perspective, which introduces a new form of policy learning that approximates safe optimal policy via Langevin dynamics, **Langevin Policy**. As all the issues in RL, there exists a classical trade-off between exploitation and exploration in Langevin policy. Langevin policy injects noise during the sampling process, which naturally exhibits the capability of exploration. However, it is not trivial to naively apply Langevin policy to RL tasks, due to the well-known limitation of Langevin MCMC with its inherent inefficiency caused by the iterative sampling process (Xie et al., 2018), which we consider an exploitation problem since the valuable information obtained through Langevin policy is conventionally discarded.

To address this issue, we propose Langevin Actor-Critic (**LAC**), which is a practical implementation to tackle above exploitation-exploration trade-off dilemma. Concretely,

LAC accumulates transitions generated by the Langevin policy and subsequently reproducing them using a generator. This approach enhances exploitation while preserving the ability to explore. Thus, the role of the generator in LAC can be viewed as that of a direct and approximate sampler of the optimal update policy. Finally, we test the proposed LAC algorithm on Safety Gym and MuJoCo-based environments, results show that LAC outperforms other state-of-the-art methods in terms of improving reward performance and satisfying constraints.

2. Preliminaries

This section presents notations and definitions related to safe reinforcement learning and methods for solving CMDPs.

2.1. Reinforcement Learning

Reinforcement learning (RL) is formulated as a Markov Decision Process (MDP) (Sutton & Barto, 2018), which is denoted by a tuple $(\mathcal{S}, \mathcal{A}, R, P, \mu)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition kernel, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\mu : \mathcal{S} \to [0, 1]$ is the initial state distribution. Let $\Pi =: \{\pi(a|s) : s \in \mathcal{S}, a \in \mathcal{A}\}$ denote the policy space, where each $\pi(\cdot|s) \in \Pi$ is a probability simplex on the action space $\mathcal{A}$. The value function is defined as $V_R^\pi(s) =: \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^\infty \gamma^t R(s_t, a_t)|s_0 = s]$ and action-value function as $Q_R^\pi(s, a) =: \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^\infty \gamma^t R(s_t, a_t)|s_0 = s, a_0 = a]$. The advantage function is defined as $A_R^\pi(s, a) =: Q_R^\pi(s, a) - V_R^\pi(s)$. Here $\tau = (s_0, a_0, \dots,)$ is a trajectory generated by policy π, where $s_0 \sim \mu, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t, a_t)$ and $\gamma \in (0, 1)$ is the discount factor. Finally, let $d^\pi(s) =: (1 - \gamma)\sum_{t=0}^\infty \gamma^t P(s_t = s|\pi)$ be the stationary discounted future state distribution of the Markov chain (starting at s_0) induced by policy π. The goal of reinforcement learning is to find a stationary policy $\pi \in \Pi$ that maximizes the expected discount return $J_R(\pi) =: \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^\infty \gamma^t R(s_t, a_t)]$.

2.2. Safe Reinforcement Learning

This paper formulates safe reinforcement learning as a constrained MDP (CMDP) $\mathcal{M} \cup \mathcal{C}$ problem (Altman, 1999), a standard MDP $\mathcal{M}$ augmented with an additional constraint set $\mathcal{C}$. The set $\mathcal{C} = \{(C_i, b_i)\}_{i=1}^m$, where C_i are the cost functions: $C_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and $b_i, i = 1, \cdots, m$ are the cost thresholds. We define the value function, action-value function, and advantage function w.r.t. cost as $V_{C_i}^\pi$, $Q_{C_i}^\pi$, and $A_{C_i}^\pi$ analogous to the standard V_R^π, Q_R^π, and A_R^π. We define the set of feasible stationary policies w.r.t. CMDP as

$$\Pi_C =: \cap_{i=1}^m \{\pi \in \Pi : J_{C_i} \le b_i\},$$

where $J_C(\pi) =: \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^\infty \gamma^t C(s_t, a_t)]$. Note that, we primarily focus on scenarios with a single constraint. The objective of safe reinforcement learning is to find an optimal policy $\pi^\star \in \Pi_C$ that satisfies the following problem:

$$\pi^\star = \arg\max_{\pi\in\Pi_C} J_R(\pi). \tag{1}$$

2.3. Constrained Policy Optimization

In the classical setting of addressing optimal policy problems in continuous MDPs, local policy search (Peters & Schaal, 2008) suggests iteratively updating the policy π_ϕ by maximizing the objective function $J_R(\pi_\phi)$ within a local region of the previous policy π_{ϕ_k} w.r.t. some distance measure $D(\pi_\phi, \pi_{\phi_k})$, which is carried out over a parameterized policy space denoted as $\Pi_\phi = \{\pi_\phi : \phi \in \Phi\} \subset \Pi$.

When extending the local policy search to CMDPs, we optimize over $\Pi_\phi \cap \Pi_C$ instead of Π_ϕ. However, directly solving CMDPs is difficult due to the additional constraint evaluation needed to judge whether an updated policy is feasible. To tackle the issues above, CPO algorithm (Achiam et al., 2017) proposes to use trust region method (Schulman et al., 2015) for CMDPs as follows: starting with policy π_{ϕ_k}, it updates new policy π_ϕ by solving the optimization problem in parameterized policy space Π_ϕ:

$$\pi^\dagger =: \max_{\pi_\phi\in\Pi_\theta} \mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}}\\ a\sim\pi_\phi}}\left[A_R^{\pi_{\phi_k}}(s, a)\right] \tag{2}$$

$$\text{s.t.} \quad J_C(\pi_{\phi_k}) + \frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}}\\ a\sim\pi_\phi}}\left[A_C^{\pi_{\phi_k}}(s, a)\right] \le b \tag{3}$$

$$\bar{D}_{\mathrm{KL}}(\pi_\phi||\pi_{\phi_k}) \le \delta, \tag{4}$$

where $\bar{D}_{\mathrm{KL}}(\pi_\phi||\pi_{\phi_k}) =: \mathbb{E}_{s\sim d^{\pi_{\phi_k}}}[D_{\mathrm{KL}}(\pi_\phi||\pi_{\phi_k})[s]]$, $D_{\mathrm{KL}}(\pi_\phi||\pi_{\phi_k})[s] =: \int_{a\in\mathcal{A}} \pi_\phi(a|s) \log \frac{\pi_\phi(a|s)}{\pi_{\phi_k}(a|s)} \mathrm{d}a$.

However, CPO introduces several errors, namely (i) It lacks a theory analysis to show the difference between the non-convex problem (2)-(4) and its convex approximation. Typically, policy optimization is a non-convex problem (Yang et al., 2021), its convex approximation may introduce some error for its original issue; (ii) CPO updates parameters according to conjugate gradient (Süli & Mayers, 2003), and its solution involves the inverse Fisher information matrix, which requires expensive computation for each update. To solve this issue, Zhang et al. (2020) propose FOCOPS to eliminate these two sources of error using a simple first-order method. Instead of solving (2-4) explicitly, Zhang et al. (2020) consider supervised learning to fit the optimal policy of (2-4), we present it as follows.

Theorem 2.1 (Optimal Policy of CPO, Zhang et al. (2020, Theorem 1)). *Let $\pi_{\phi_k}(a|s)$ be a feasible solution to the CPO problem (2-4), then $\pi^\dagger$ takes the following form*

$$\pi^\dagger(a|s) = \frac{\pi_{\phi_k}(a|s)}{Z_{\lambda^\star,\nu^\star}(s)} \exp\left(\frac{1}{\lambda}\left(A_R^{\pi_{\phi_k}}(s, a) - \nu A_C^{\pi_{\phi_k}}(s, a)\right)\right), \tag{5}$$

where $Z_{\lambda^\star,\nu^\star}(s)$ is the normalization term, $\lambda^\star, \nu^\star$ are the

solutions to the optimization problem:

$$(\lambda^\star, \nu^\star) =: \min_{\lambda,\nu\geq 0} \left\{ \lambda\delta + \nu\tilde{b} + \lambda \mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}} [\log Z_{\lambda,\nu}(s)] \right\}, \tag{6}$$

where $\tilde{b} = (1-\gamma)(b - \tilde{J}_C(\pi_{\phi_k}))$.

FOCOPS considers to project the policy $\pi^\dagger$ back into the parameterized policy space Π_ϕ via finding the $\pi_{\phi_{k+1}} \in \Pi_\phi$ that is closest to $\pi^\dagger$ as follows,

$$\pi_{\phi_{k+1}} =: \arg\min_{\pi\in\Pi_\phi} \mathbb{E}_{s\sim d^{\pi_{\phi_k}}} \left[D_{\mathrm{KL}}(\pi^\dagger \| \pi)[s] \right].$$

3. Langevin Policy

In this section, we propose Langevin policy, which models action sequences with Langevin chains as summarized in Algorithm 1.

The policy (5) is a typical energy-based model (EBM) (LeCun et al., 2006), we rewrite it as follows,

$$\pi^\dagger(a|s) = \frac{1}{Z_k(\phi)} \exp\{-E_k(s,a;\phi)\}, \tag{7}$$

where

$$E_k(s,a;\phi) = \frac{1}{\lambda}(\nu A_C^{\pi_k}(s,a) - A_R^{\pi_k}(s,a)) - \log \pi_k(a|s), \tag{8}$$

and $Z_k(\phi) = \iint_{s\in\mathcal{S}, a\in\mathcal{A}} \exp\{-E_k(s,a;\phi)\}\mathrm{d}s\mathrm{d}a$ is the normalization term. However, directly sampling actions according to policy (7) is difficult due to the unknown $Z_k(\phi)$. Langevin algorithm is a classic Markov chain Monte Carlo (MCMC) method to solve the EBM (7), which generates an action sequence $\{\tilde{a}_i\}_{0\leq i\leq T}$ as follows,

$$\tilde{a}_i = \tilde{a}_{i-1} + \frac{\eta}{2}\nabla_a \log \pi^\dagger(a|s)\Big|_{a=\tilde{a}_{i-1}} + \sqrt{\eta} z_i, \tag{9}$$

where $z_i \sim \mathcal{N}(0, I)$, $\eta > 0$ is the step size, T is total step, and the initial action $\tilde{a}_0 \sim p_0$, p_0 is some initial distribution. We provide the details of Langevin policy in Algorithm 1.

Theorem 3.1 (Convergence of Langevin Policy)**.** *Let the action sequence* $\{\tilde{a}_i\}_{0\leq i\leq T}$ *be generated by (9), and* $\pi_i(\cdot|s)$ *denotes the distribution of* a_i *under the state* s*. Let the energy* $E_k(s,a;\phi)$ *be with* l *Lipschitz continuous gradients and* m *strong convexity, i.e. for each* $s\in\mathcal{S}$*, for each* k*:* $m\mathbf{I} \preceq \nabla_a^2 E_k(s,a;\phi) \preceq l\mathbf{I}$*. If* $\pi_0(\cdot|s) = \mathcal{N}(0, \frac{1}{m})$*, and*

$$\eta =: \frac{m\epsilon}{8dl^2}, \quad T =: 16\frac{l^2}{m^2}\frac{d\log\frac{dl}{m\epsilon}}{\epsilon},$$

where d *is the dimension of the action (i.e., each action* $a\in\mathcal{A}$ *satisfies* $a\in\mathbb{R}^d$*), then*

$$\mathrm{KL}\left(\pi_T(\cdot|s)\|\pi^\dagger(\cdot|s)\right) \leq \epsilon.$$

Algorithm 1 Langevin Policy

Input: $s, \lambda, \nu, Q_R^\pi, Q_C^\pi, \pi, \tilde{a}_0$
Initialization:, Langevin steps T, Langevin stepsize η,
for $i = 1, \cdots, T$ **do**
  $z_i \sim \mathcal{N}(0, I)$;

$$E(s,a) = \frac{1}{\lambda}(\nu Q_C^\pi(s,a) - Q_R^\pi(s,a)) - \log\pi(a|s);$$
$$\tilde{a}_i = \tilde{a}_{i-1} - \frac{\eta}{2}\nabla_a E(s,a)|_{a=\tilde{a}_{i-1}} + \sqrt{\eta} z_i;$$

end for
Output $\tilde{a}_T$

We show its proof in Appendix C. Theorem 3.1 illustrates that the distribution of $\tilde{a}_T$ is consistent with $\pi^\dagger(\cdot|s)$ (7) when $\eta \to 0$ and $T \to \infty$, i.e., the action $\tilde{a}_T$ becomes a sample of the policy $\pi^\dagger(\cdot|s)$ (7). Additionally, Theorem 3.1 illustrates that Langevin policy produces a policy π with $\mathrm{KL}\left(\pi(\cdot|s)\|\pi^\dagger(\cdot|s)\right) \leq \epsilon$ in $\tilde{\mathcal{O}}\left(\frac{d}{\epsilon}\right)$ steps, which depends on the condition that energy function $E_k(\cdot, s; \phi)$ is with Lipschitz continuous gradients and strong convexity on the action space $\mathcal{A}$. Finally, we know Langevin policy is the stationary distribution follows the next stochastic dynamic of $\{A(t)\}_{t\geq 0}$ on the continuous time $t\in[0, +\infty)$,

$$\mathrm{d}A(t) = -\nabla_a E_k(a, s; \phi)\mathrm{d}t|_{a=A(t)} + \sqrt{2}\mathrm{d}B(t), \tag{10}$$

with $A_0 \sim \rho_0$, ρ_0 is some initial distribution, and $B(t)$ is Brownian motion at time t.

4. LAC: Langevin Actor-Critic

Although the Langevin policy provides a way to represent the policy (7), in practice, it meets the following challenges when applied to reinforcement learning.

(C1): The First Challenge. If we directly apply Langevin policy (see Algorithm 1) to interact with the environment, each action from $\pi_\dagger(\cdot|s)$ requests for T Langevin steps, which is time-consuming. It is inefficient for the practical application of Langevin policy to real-world scenarios.

(C2): The Second Challenge. By leveraging the reformulation of Eq. (7) and the efficacious Langevin policy described in Algorithm 1, it appears that there are no hindrances in obtaining exact actions $\tilde{a}$ from $\pi^\dagger(\cdot|s)$ within the parametric policy space Π_ϕ. It is crucial to notice that the assurance of obtaining exact samples from $\pi^\dagger(\cdot|s)$ is contingent upon T steps approaching infinity. However, running such a chain is excessively time-consuming to be considered feasible.

(C3): The Third Challenge. Langevin policy needs to calculate the gradient of $E(s,a)$ with respect to a, namely *score*. By leveraging the equality $\nabla_a A(s,a) = \nabla_a Q(s,a)$, we substitute the advantage function $A(s,a)$ in the energy

function $E(s,a)$ with $Q(s,a)$. However, a well-known challenge arises in the form of estimating bias in Q-value function (as illustrated in Section 4.1), which significantly affects the performance of Langevin policy.

Those three challenges are the main focus of this section. Section 4.1 presents the details of our approach to address these challenges. Section 4.2 presents the implementation of our proposed method: Langevin Actor-Critic (LAC).

4.1. Approach to Address Challenges

Generator to Approximate Langevin Policy. To solve the problem that appears in challenge **(C1)**, instead of acquiring each action a_T from $\pi^\dagger(\cdot|s)$ through complete Langevin chain, we consider a parameterized policy π_ϕ as a generator to approximate the output of Langevin policy, where π_ϕ plays the role of an estimator of Langevin policy. This enables the agent to use π_ϕ to interact with the environment to collect data, which reduces the training and testing time.

Informative-initializated Short-Run MCMC. To solve the problem from the second challenge (**C2**), for each action sequence $\{\tilde{a}_i\}_{0\leq i\leq T}$ generated by Langevin policy, we set the following initialization $\tilde{a}_0 \sim \pi_\phi$, and a fixed small T. We present those details as follows.

There exist two ways of initialization in MCMC methods (Nijkamp et al., 2020): noninformative and informative initialization. The noninformative initialization is independent of any model, where the initial action $\tilde{a}_0$ is generated from an unrelated distribution (e.g., uniform or Gaussian distribution). Since noninformative initialization loses the prior and historical information, it is inefficient to be applied to Langevin policy.

The central idea of informative initialization involves inheriting the hidden information in the model. For a given Markov chain initialized from its approximate steady distribution, such an initialization makes MCMC methods converge and learn faster than noninformative initialization. Thus, applying informative initialization to Langevin policy is an efficient way to solve **(C2)**. In this paper, we consider Langevin policy with informative initialization of $\tilde{a}_0$ as follows,

$$a \leftarrow \text{Langevin Policy}(s, \lambda, \nu, Q_R, Q_C, \tilde{a}_0 = \pi_\phi(\cdot|s)). \tag{11}$$

We should emphasize that the distinction between Eq. (9) and Eq. (11)) is that initialization in vanilla Langevin policy (9) draws $\tilde{a}_0$ from noninformative distribution such as uniform or Gaussian, while initialization in our revised Langevin policy (11) draws $\tilde{a}_0$ from a previously learned model π_ϕ (which incorporates hidden information acquired from past MCMC chains, hence speed up the convergence rate of Langevin policy (9)).

Clipped Double Q-Learning. The Langevin policy (see Algorithm 1) captures the underlying distribution by assigning an unnormalized probability scalar to each possible data point, which implies the policy $\pi^\dagger(\cdot|s)$ aims to assign lower energies

$$E(s,a) = \frac{1}{\lambda}(\nu Q_C^\pi(s,a) - Q_R^\pi(s,a)) - \log \pi(a|s)$$

to regions in the state-action space that correspond to higher returns and lower costs. We use parametric Q_θ to estimate the state-action, and we use Langevin policy as the target:

$$\mathcal{L}_Q(\theta) = \frac{1}{2}\left(r + \gamma Q_\theta(s', \tilde{a}_T) - Q_\theta(s,a)\right)^2, \tag{12}$$

where the action $\tilde{a}_T$ is the output of Langevin policy:

$$\tilde{a}_T = \text{Langevin Policy}(s = s', \lambda, \nu, Q_R, Q_C, \pi_\phi(\cdot|s')).$$

Thus, as classical policy evaluation (e.g., DQN (Van Hasselt et al., 2016)), the Langevin policy (see Algorithm 1) involves estimating Q-value, which also suffers from the well-known overestimate state-action values under certain conditions (Van Hasselt et al., 2016). Thus we consider the technique from TD3 (Fujimoto et al., 2018) to update targets for reward and cost:

$$\begin{aligned} y^R &\leftarrow r + \gamma \min_{i=1,2} Q_{\theta_i'^R}(s', \tilde{a}_T), \\ y^C &\leftarrow c + \gamma \max_{i=1,2} Q_{\theta_i'^C}(s', \tilde{a}_T). \end{aligned} \tag{13}$$

We present the insights as follows. In the presence of an error ϵ in the target, the "maximum" (or "minimum") over the value along with its error will tend to exceed the actual "maximum" (or "minimum"), for reward $\mathbb{E}_\epsilon[\max_{\tilde{a}_T}(Q_R(s', \tilde{a}_T)+\epsilon)] \geq \max_{\tilde{a}_T} Q_R(s', \tilde{a}_T)$ (for cost is $\mathbb{E}_\epsilon[\min_{\tilde{a}_T}(Q_C(s', \tilde{a}_T) + \epsilon)] \leq \min_{\tilde{a}_T} Q_C(s', \tilde{a}_T)$). Recall that Langevin policy incorporates the estimation of Q-value function for both reward and cost, which introduces bias of overestimation and underestimation. This bias is detrimental given that the effectiveness of Langevin policy is heavily reliant on the unbiased estimation of energy function, specifically the Q-value function for reward and cost. We follow the clipped variant of Double Q-learning (Fujimoto et al., 2018) to address the third challenge (**C3**), see Eq.(13).

4.2. Practical Implementation of Langevin Actor-Critic

We show the detailed implementation of our proposed Langevin Actor-Critic (LAC) in Algorithm 2.

Step1: Collect Transitions. As presented in Section 4.1, π_ϕ approximates Langevin policy. LAC collects the trajectories τ according to π_ϕ for estimation, see Line 3-4 in Algorithm 2.

Step2: Run Langevin Policy. For each iteration, LAC runs Langevin policy according to Algorithm 1 to get action $\tilde{a}_T$. Concretely, for each action $\tilde{a}_T$, LAC starts with

Algorithm 2 LAC (Langevin Actor-Critic)

Initialization: λ, ν, reward critic networks $Q_{\theta^R_{\{1,2\}}}$, cost critic networks $Q_{\theta^C_{\{1,2\}}}$, actor networks π_ϕ, target networks $\theta'^R_{\{1,2\}} \leftarrow \theta^R_{\{1,2\}}, \theta'^C_{\{1,2\}} \leftarrow \theta^C_{\{1,2\}}, \phi' \leftarrow \phi$, replay buffer $\mathcal{B}$;
for $k = 0, 1, \cdots, K-1$ **do**
  Generate trajectories $\tau \sim \pi_\phi$;
  Store transitions tuple (s, a, r, c, s') in $\mathcal{B}$;
  Update ν according to (14);
  for mini-batch of transitions (s, a, r, c, s') in $\mathcal{B}$ **do**
    $\tilde{a} \leftarrow$ Langevin Policy$(s', \lambda, \nu, Q_{\theta^R_i}, Q_{\theta^C_i}, \pi_{\phi'}(\cdot|s'))$;
    Calculate target value y^R, y^C according to (13);
    Update critic $\theta^R_{\{1,2\}}$ and $\theta^C_{\{1,2\}}$ according to (12);
    Update actor ϕ according to (15);
    Update target networks:
    $\theta'^R_i \leftarrow \tau\theta_i + (1-\tau)\theta'^R_i$;
    $\theta'^C_i \leftarrow \tau\theta_i + (1-\tau)\theta'^C_i$;
    $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$;
  end for
end for

$\tilde{a}_0 \sim \pi_\phi$ and takes a fixed Langevin steps T, see line 4 in Algorithm 1. To overcome overestimation (or underestimation) of Q-value function, LAC considers $Q^\pi_R = \min_{i\in\{1,2\}} Q_{\theta^R_i}, Q^\pi_C = \max_{i\in\{1,2\}} Q_{\theta^C_i}$.

Additionally, since Langevin policy depends on λ, ν, LAC considers the following update rule. We adopt a fixed value for λ throughout the training process, which has been shown able to obtain comparable results (Zhang et al., 2020). Recall that the optimal $(\lambda^\star, \nu^\star)$ minimizes the dual function (6), which is denoted as

$$L(\pi^\dagger(\cdot|s), \lambda, \nu) =: \lambda\delta + \nu\tilde{b} + \lambda \mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}} [\log Z_{\lambda,\nu}(s)].$$

Since $(\lambda^\star, \nu^\star) = \min_{\nu\geq 0,\lambda}\{L(\pi^\dagger(\cdot|s), \lambda, \nu)\}$, LAC considers gradient decent method to obtain ν.

Proposition 4.1. *The derivative of $L(\pi^\dagger(\cdot|s), \lambda, \nu)$ w.r.t. ν:*

$$\frac{\partial L(\pi^\dagger, \lambda, \nu)}{\partial \nu} = \tilde{b} - \mathbb{E}_{s\sim d^{\pi_{\phi_k}}, a\sim\pi^\dagger(\cdot|s)}[A^{\pi_{\phi_k}}(s, a)].$$

We show its proof in Appendix B. With proper parameter tuning, it is reasonable to consider that π_{ϕ_k} is close to $\pi_\dagger(\cdot|s)$, which implies for any $s \in \mathcal{S}$,

$$\mathbb{E}_{a\sim\pi^\dagger(\cdot|s)}[A^{\pi_{\phi_k}}(s, a)] \approx \mathbb{E}_{a\sim\pi_{\phi_k}}[A^{\pi_{\phi_k}}(s, a)] = 0.$$

Thus, in practice, we update ν as follows,

$$\nu \leftarrow \{\nu - \eta(b - J_C(\pi_{\phi_k})\}_+, \tag{14}$$

where $\{\cdot\}_+$ denotes the positive part operator, i.e., if $x \leq 0$, $\{x\}_+ = 0$, else $\{x\}_+ = x$, $\eta > 0$ is step-size.

Step3: Update Actor and Critic. For actor π_ϕ, recall that π_ϕ plays the role of a generator that approximates the Langevin policy. Furthermore, inspired by that negative log probability of the action $\tilde{a}_T$, i.e., $-\log\pi_\phi(\tilde{a}_T|s)$ is proportional to the energy function $E(s, a)$ in Algorithm 1, LAC considers a heuristic method that resembles the maximum likelihood learning to update π_ϕ by gradient ascent on

$$\mathcal{L}_A(\phi) = -\log\pi_\phi(\tilde{a}_t|s) \tag{15}$$

to make $\pi_\phi(\cdot|s)$ close to $\pi^\dagger(\cdot|s)$. For critic Q_{θ^R} and Q_{θ^C}, LAC updates the pair of critics towards the minimum and maximum target value of actions selected by Langevin policy as described by Eq. (13).

Remark 4.2 (Regularization for Policy Update). *The proposed heuristic loss (15) for actor π_ϕ in LAC raises concerns regarding potential fluctuations in policy updates and their potential impact on performance. To make the LAC update stable, we make use of KL-regularization to penalize the discrepancy between new policy π_ϕ and target policy $\pi_{\phi'}$, which results in a new loss function as follows,*

$$\mathcal{L}^{\mathrm{KL}}_A(\phi) = -\log\pi_\phi(\tilde{a}_t|s) + \beta D_{\mathrm{KL}}(\pi_\phi(\tilde{a}_t|s)\|\pi_{\phi'}(\tilde{a}_t|s)). \tag{16}$$

Section 6.3 shows that LAC achieves stable and comparative performance with KL-regularization.

5. Related Work

Safe RL is a fast-growing field, the recent work (Gu et al., 2022) provides a comprehensive review of safe RL from the perspectives of methods, theory, and applications. This section mainly reviews the works with respect to CPO-based safe RL and RL with generative or sampling methods.

CPO-based Safe RL. CPO (Achiam et al., 2017) (2)-(4) is the first approach that utilizes policy gradient to solve safe RL problems. It approximates the constrained optimization problem with surrogate functions, making it a convex optimization problem. Existing recent works (e.g., (Vuong et al., 2019; Yang et al., 2020; Han et al., 2020; Bisi et al., 2020; Bharadhwaj et al., 2021)) try to find some convex approximations to replace the term $A_\pi(s, a)$ and $D_{\mathrm{KL}}(\pi, \pi_\phi)$ in (2)-(4). FOCOPS (Zhang et al., 2020) first gives a near-closed form solution for the best policy in the nonparametric policy space and then projects the policy back into the parametric policy space by minimizing the KL-divergence. Then, CUP (Yang et al., 2022) provides a non-convex implementation via only first-order optimizers, which does not require any convex approximation on the convexity of the objectives. P3O (Zhang et al., 2022) and APPO (Dai et al., 2023) are also non-convex implementations to solve safe RL problems.

RL with Generative or Sampling Methods. Several recent works (Chen et al., 2021; Janner et al., 2022; Yang et al.,

2023) have investigated the application of generative models in RL. Derived from Decision Transformer (DT) (Chen et al., 2021), Liu et al. (2023) leverages the modeling capability of Transformer to solve offline safe RL problems. Zheng et al. (2024) study safe offline reinforcement learning with feasibility-guided diffusion model (Ho et al., 2020). Instead of solving safe RL in the context of planning. Some other works adopt sampling methods to serve as dynamics model, thereby enhancing the exploration ability in online safe RL. Maeda et al. (2019) sample trajectories from a generative model to learn threat function to determine whether a given action is safe. HasanzadeZonuzy et al. (2021) sample a safe action according to safety critic via rejection sampling. Wang et al. (2023) introduce a generative-model-based soft barrier function that leverages the generative model to learn the dynamics and stochasticity of the environment. Rather than learning the world model, Kamalaruban et al. (2020) introduces a sampling perspective to solve the robust reinforcement learning problem by leveraging Stochastic Gradient Langevin Dynamics. However, none of the above methods consider a sampling based policy to solve safe RL, which we demonstrate in our framework as a powerful method for effectively exploring the safe optimal policy and resulting in comparative performance.

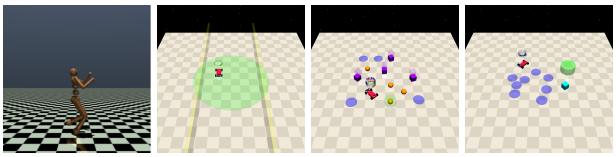
6. Experiments

In this section, we perform extensive experiments to demonstrate that: (1) LAC outperforms current state-of-the-art benchmarks on various high-dimensional tasks. (2) LAC balances the exploitation-exploration trade-off resulting in faster convergence towards the safe optimal policy compared to the selected baseline methods.

Baselines. We compare LAC against baseline methods, including CPO (Achiam et al., 2017), FOCOPS (Zhang et al., 2020), CUP (Yang et al., 2022), PCPO (Yang et al., 2020), P3O (Zhang et al., 2022) and RESPO (Ganai et al., 2023). In addition, we incorporate PPO-L and TRPO-L into our evaluation, which represents the Lagrangian-based approach with PPO (Schulman et al., 2017) and TRPO (Schulman et al., 2015) as demonstrated by Ray et al. (2019).

Tasks. We compare the proposed algorithm with existing approaches on four kinds of tasks as described in the caption of Figure 1. *Velocity* and *Circle* tasks are implemented using OpenAI Gym API (Brockman et al., 2016) for MuJoCo physical simulator (Todorov et al., 2012). The other two tasks *Button* and *Goal* are implemented in Safety Gym benchmark suite (Ray et al., 2019).

We conduct two sets of experiments to verify the performance of our algorithm. For the MuJoCo tasks, we train three agents (i.e., Ant-v3, Swimmer-v3, Humanoid-v3) to run across *Velocity* and *Circle* tasks. For the Safety Gym tasks, we train two agents (i.e., Point and Car) across *Goal* and *Button* tasks with Level 1 difficulty setting. Implementation details for all experiments can be found in Appendix D.



(a) Velocity (b) Circle (c) Button (d) Goal

Figure 1. The velocity, circle, button and goal tasks. (a) Velocity task: The agent is rewarded for moving as quickly as possible while adhering to velocity limits. (b) Circle task: The agent is rewarded for circling the center of the circle and penalized for going outside the boundaries. (c) Button tasks: The agent is rewarded for pressing a goal button while avoiding hazards, gremlins, and pressing the wrong buttons. (d) Goal tasks: The agent is rewarded for entering a goal area while avoiding hazards.

6.1. MuJoCo Tasks with Constraint

Table 1 lists the comparison results on MuJoCo tasks. It is observed that LAC consistently outperforms other baseline methods in terms of satisfying the imposed cost limits and achieving a higher reward across all evaluated tasks.

Figure 2 shows the learning curves, it can be seen that LAC outperforms other baselines by a large margin (particularly evident in Humanoid-v3 and HumanoidCircle-v0) in terms of reward on all tasks while satisfying the cost limits. In the Humanoid-v3 task, LAC initially exhibits an aggressive exploring policy (in terms of cost) at the beginning. However, as the training proceeds and reaches the 0.5 million step mark, LAC gradually converges towards a safe policy and substantially improves reward performance. Although achieving the highest reward performance in Swimmer-v3 task, PPO-L and TRPO-L incur significant constraint violations and experience degradation in reward performance when successfully approaching safe policy (e.g., Ant-v3, HumanoidCircle-v0 and Humanoid-v3). Similarly, Table 1 reveals that LAC consistently outperforms other approaches in terms of satisfying the imposed cost limits and achieving a higher reward across all evaluated tasks.

6.2. Safety Gym Tasks with Constraint

Safety Gym tasks introduce randomized safety-relevant environmental elements (e.g., Hazards, Gremlins, and Buttons) at the beginning of each episode, posing a higher level of difficulty compared to MuJoCo tasks.

We provide learning curves in Figure 3 and numerical summaries in Table 2. Overall, LAC exhibits a favorable trade-off between improving reward performance and adhering

Table 1. Average results for CPO, CUP, FOCOPS, PPO-L, TRPO-L and LAC after 2.5 million steps. Cost limits are in brackets under the environment names. **Bold**: Safe agents with the cost under cost limit. **Blue**: Safe agents with the highest reward.

Environment		CPO	CUP	FOCOPS	PPO-L	TRPO-L	LAC (ours)
Ant-v3	Reward	994.50 ± 27.44	1159.81 ± 84.08	1803.86 ± 42.90	-24.32 ± 8.26	-15.38 ± 11.13	$\mathbf{1940.71 \pm 16.31}$
(103.12)	Cost	113.87 ± 7.24	$\mathbf{100.68 \pm 1.80}$	$\mathbf{102.91 \pm 3.12}$	$\mathbf{5.48 \pm 1.82}$	$\mathbf{6.70 \pm 2.63}$	$\mathbf{103.04 \pm 0.12}$
Swimmer-v3	Reward	-2.45 ± 13.88	30.42 ± 4.48	34.46 ± 3.35	49.07 ± 3.33	47.07 ± 5.74	$\mathbf{34.56 \pm 4.94}$
(24.52)	Cost	27.75 ± 6.21	$\mathbf{23.65 \pm 0.61}$	25.51 ± 3.04	69.65 ± 2.61	70.22 ± 4.56	$\mathbf{24.45 \pm 1.30}$
Humanoid-v3	Reward	375.93 ± 21.19	1795.83 ± 1099.19	3196.58 ± 909.07	233.74 ± 52.49	144.92 ± 16.72	$\mathbf{5023.27 \pm 107.99}$
(20.14)	Cost	24.58 ± 2.12	$\mathbf{19.21 \pm 0.62}$	$\mathbf{19.83 \pm 0.91}$	$\mathbf{5.98 \pm 2.10}$	$\mathbf{3.42 \pm 0.70}$	$\mathbf{20.12 \pm 0.58}$
HumanoidCircle-v0	Reward	174.96 ± 19.76	390.39 ± 51.27	862.92 ± 67.24	0.01 ± 0.04	1.90 ± 4.06	$\mathbf{925.03 \pm 111.95}$
(50.00)	Cost	$\mathbf{40.00 \pm 5.19}$	$\mathbf{38.11 \pm 2.05}$	$\mathbf{48.61 \pm 2.47}$	$\mathbf{9.72 \pm 0.63}$	$\mathbf{10.65 \pm 0.62}$	$\mathbf{49.57 \pm 1.16}$

Table 2. Average results for P3O, RESPO, CPO, PPO-L, TRPO-L, FOCOPS, PCPO and LAC after 2.5 million steps. Cost limits are in brackets under the environment names. **Bold**: Safe agents with the cost under cost limit. **Blue**: Safe agents with the highest reward.

Environment		P3O	RESPO	CPO	PPO-L	TRPO-L	FOCOPS	PCPO	LAC (ours)
PointGoal	Reward	0.94 ± 1.04	13.71 ± 3.44	23.16 ± 2.09	11.36 ± 4.23	16.34 ± 4.96	6.01 ± 3.59	9.00 ± 2.09	$\mathbf{14.20 \pm 3.35}$
(25.00)	Cost	30.53 ± 29.50	35.51 ± 35.90	46.00 ± 15.85	$\mathbf{16.48 \pm 10.95}$	28.40 ± 16.14	38.08 ± 12.63	46.21 ± 12.60	$\mathbf{22.21 \pm 3.03}$
PointButton	Reward	-0.43 ± 0.34	3.52 ± 3.23	18.42 ± 3.04	0.90 ± 1.98	7.61 ± 1.98	2.79 ± 2.12	1.14 ± 0.91	$\mathbf{2.45 \pm 1.36}$
(25.00)	Cost	35.21 ± 6.05	45.08 ± 51.49	114.50 ± 32.53	$\mathbf{17.76 \pm 17.49}$	54.10 ± 26.55	31.11 ± 12.74	38.29 ± 5.43	$\mathbf{21.11 \pm 6.74}$
CarGoal	Reward	-0.11 ± 1.17	15.23 ± 3.55	25.24 ± 4.80	12.38 ± 4.21	18.68 ± 2.31	7.91 ± 7.70	10.42 ± 4.11	$\mathbf{24.01 \pm 2.21}$
(25.00)	Cost	112.75 ± 92.75	$\mathbf{21.44 \pm 30.35}$	45.42 ± 19.30	$\mathbf{19.50 \pm 16.92}$	27.44 ± 10.99	$\mathbf{20.09 \pm 9.20}$	41.71 ± 18.74	$\mathbf{22.15 \pm 4.71}$
CarButton	Reward	-4.52 ± 4.76	0.75 ± 1.24	10.16 ± 2.54	0.71 ± 0.42	0.80 ± 1.02	0.65 ± 0.60	0.01 ± 0.30	$\mathbf{1.50 \pm 1.44}$
(25.00)	Cost	90.10 ± 28.59	32.01 ± 47.79	155.92 ± 70.06	73.40 ± 73.48	49.04 ± 53.23	39.67 ± 7.28	60.70 ± 13.67	$\mathbf{16.54 \pm 13.83}$

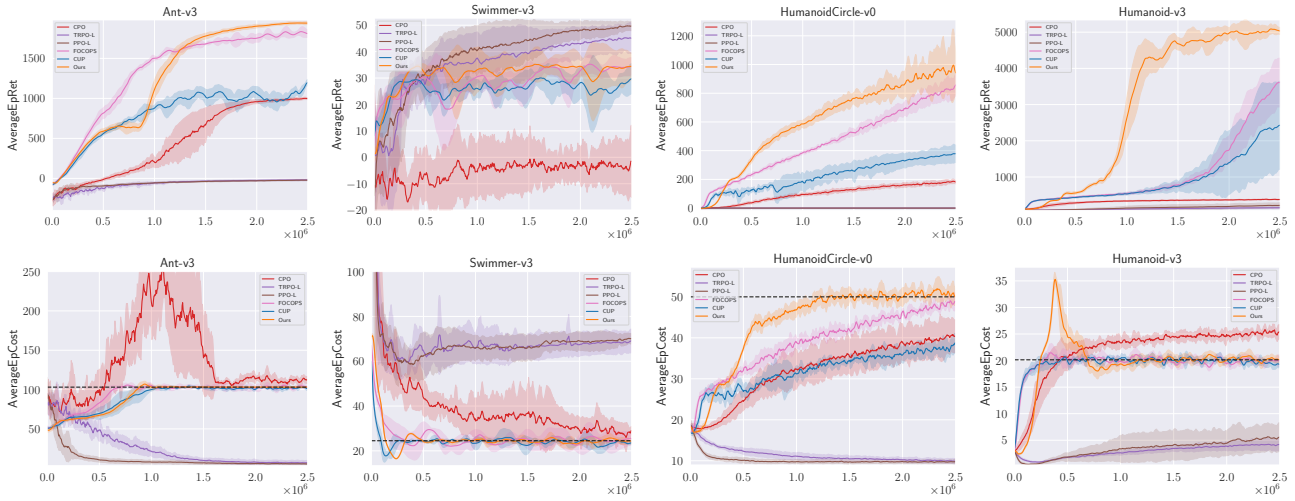


Figure 2. Comparison of the average rewards and costs in MuJoCo tasks. Each column corresponds to a task. The x-axis represents the total environment step. The first row shows the performance in terms of reward (higher is better), and the second row shows the evaluated cost (lower is better) under different cost thresholds, which are represented by the dashed line in the second row. Each.The solid line represents the mean and the shaded area represents the confidence interval for one standard deviation. The plots are smoothed with a moving average window of 5.

to the imposed constraint limits (set as 25) across all tasks. When comparing the cumulative reward under satisfying constraints, LAC outperforms all other methods. Given the increased difficulty of these tasks, most of the chosen baseline methods face a significant challenge in exploring a safe policy that satisfies the predetermined cost limits within 2.5 million steps. It is evident that CPO, TRPO-L, and PCPO struggle to satisfy cost limits, only PPO-L (excluding CarButton tasks), FOCOPS (in CarGoal task) and RESPO (in CarGoal task) achieve relatively better reward performance while maintaining satisfied constraint violations. The inferior performance of these baselines compared to MuJoCo tasks highlights the complexity of Safety Gym tasks and demonstrates the efficacy of LAC.

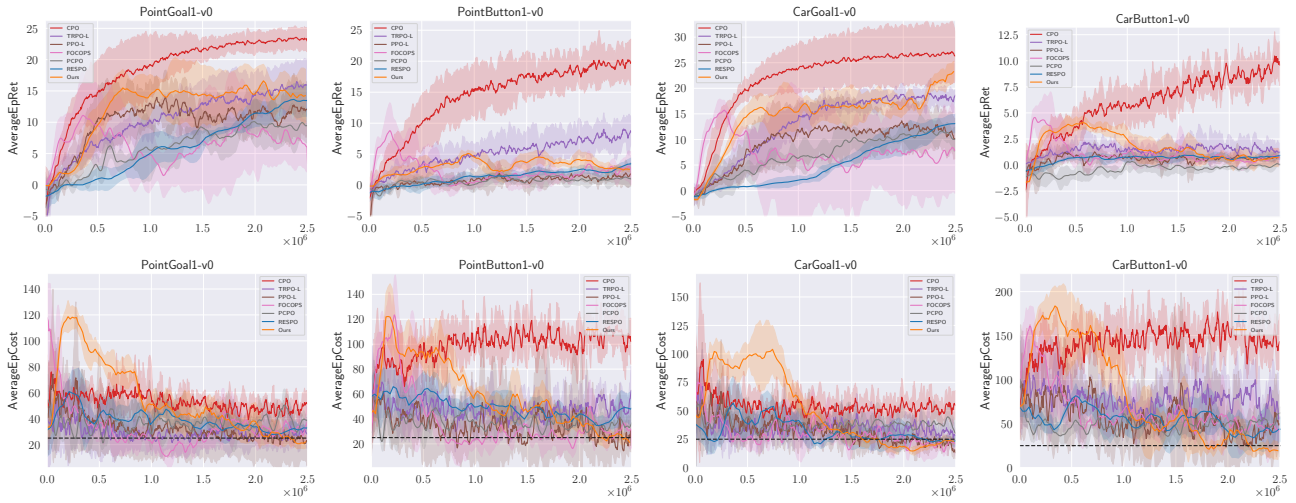


Figure 3. Comparison of the average rewards and costs on Safety Gym Tasks. The x-axis represents the total environment step.

Table 3. Average rewards and costs after 2.5 million environment steps. Comparison ablation over noninformative initialization (NI), varying Langevin steps T (VT) and policy KL-regularization (PR). We run T among {30,50,100,200,500} for NI, T among {10,20,30,40,50,100} for VT and β among {0,1,5,10,20} for PR. We choose the best performance for NI, VT and PR.

Method		**PointGoal**	**CarGoal**
LAC	Reward	14.20 ± 3.35	24.01 ± 2.21
	Cost	22.21 ± 3.03	22.15 ± 4.71
NI	Reward	5.58 ± 1.65	13.25 ± 7.92
	Cost	29.62 ± 18.33	16.02 ± 5.88
VT	Reward	15.33 ± 2.03	23.00 ± 1.33
	Cost	21.53 ± 3.03	22.98 ± 8.20
PR	Reward	13.63 ± 1.36	19.75 ± 5.07
	Cost	17.93 ± 2.60	18.33 ± 2.26
PR (1e7)	Reward	14.71 ± 0.70	21.85 ± 4.47
	Cost	20.85 ± 7.61	17.27 ± 5.58

6.3. Ablation Study

Additionally, we perform ablation studies (without any other hyperparameters tuning) to validate the design choices of informative initialization and analyze the effects of policy regularization and varying Langevin steps. To fully explore the potential of LAC, we conduct experiments on *Goal* tasks, where LAC has already exhibited remarkable performance.

In contrast to LAC with informative initialization, noninformative initialization (NI) fails to effectively exploit the latent information acquired from the Langevin policy. As shown in the second row of Table 3, LAC with noninformative initialization (NI) undergoes a degradation both in reward performance and constraint satisfaction. It is worth to mention that the best performance for NI is achieved when T is set to 500. However, the large T results in a considerable computational cost. For varying T (VT), the best performance is observed with T=50 for the PointGoal task and T=100 for the CarGoal task. VT outperforms LAC in PointGoal, while it performs worse on CarGoal task. This indicates that a larger T is not always essential for LAC with informative initialization. For policy KL-regularization (PR), we provide the best performance both after 2.5 million and 10 million steps. We observe that PR does not yield a substantial performance improvement in LAC, which can be attributed to the inherent stability of LAC. However, PR does contribute to a lower cost while maintaining a relatively comparative performance. For a detailed and comprehensive comparison of the results, please refer to Appendix D.1.

7. Limitations and Future Works

Although we have demonstrated the effectiveness of LAC in both MuJoCo-based and Safety Gym tasks, there are limitations that future work can improve. First, our implementation for LAC incorporates a generator π_ϕ as an intermediary actor between Langevin policy and the environment, to reduce the time consumed by sampling process and accumulate the transitions induced by Langevin policy. Such a compromise may introduce errors that can potentially undermine the capabilities of LAC. Second, we choose Gaussian policy as the generator to ensure the tractability of policy distribution, which is crucial for calculating energy function in Langevin policy. However, this may constrain the capability of Langevin policy due to its limited expressiveness, potentially leading to inferior performance. Future work can focus on substituting the Gaussian generator with a more expressive model to fully explore LAC's capability.

8. Conclusion

In this work, we formulate the Langevin policy for safe RL, which can directly access the safe optimal update policy through an iterative sampling way to achieve higher reward and lower cost. We further propose LAC to effectively balances the exploitation-exploration trade-off introduced by Langevin policy. We hope that our work will stimulate further exploration and development of sampling based methods in the field of safe reinforcement learning.

Acknowledgements

This research was supported by Zhejiang Provincial Natural Science Foundation of China under Grant No. LY24F020021, and the Ningbo Science and Technology Special Projects under Grant No. 2023Z129 and 2022Z235.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Achiam, J., Held, D., Tamar, A., and Abbeel, P. Constrained policy optimization. In *Proceedings of International Conference on Machine Learning*, pp. 22–31, 2017.

Altman, E. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

Bharadhwaj, H., Kumar, A., Rhinehart, N., Levine, S., Shkurti, F., and Garg, A. Conservative safety critics for exploration. In *International Conference on Learning Representations*, 2021.

Bisi, L., Sabbioni, L., Vittori, E., Papini, M., and Restelli, M. Risk-averse trust region optimization for reward-volatility reduction. In Bessiere, C. (ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 4583–4589, 2020.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym, 2016.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Cheng, X. and Bartlett, P. Convergence of langevin mcmc in kl-divergence. In *Algorithmic Learning Theory*, pp. 186–211. PMLR, 2018.

Chow, Y., Ghavamzadeh, M., Janson, L., and Pavone, M. Risk-constrained reinforcement learning with percentile risk criteria. *Journal of Machine Learning Research*, 18 (1):6070–6120, 2017.

Chow, Y., Nachum, O., Duenez-Guzman, E., and Ghavamzadeh, M. A lyapunov-based approach to safe reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Dai, J., Ji, J., Yang, L., Zheng, Q., and Pan, G. Augmented proximal policy optimization for safe reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7288–7295, 2023.

Ding, D., Wei, C.-Y., Zhang, K., and Ribeiro, A. Last-iterate convergent policy gradient primal-dual methods for constrained MDPs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Fawzi, A., Balog, M., Huang, A., Hubert, T., Romera-Paredes, B., Barekatain, M., Novikov, A., R Ruiz, F. J., Schrittwieser, J., Swirszcz, G., et al. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53, 2022.

Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

Ganai, M., Gong, Z., Yu, C., Herbert, S., and Gao, S. Iterative reachability estimation for safe reinforcement learning. *arXiv preprint arXiv:2309.13528*, 2023.

Gu, S., Yang, L., Du, Y., Chen, G., Walter, F., Wang, J., Yang, Y., and Knoll, A. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*, 2022.

Han, M., Tian, Yuanand Zhang, L., Wang, J., and Pan, W. Reinforcement learning control of constrained dynamic systems with uniformly ultimate boundedness stability guarantee. *arXiv preprint arXiv:2011.06882*, 2020.

HasanzadeZonuzy, A., Bura, A., Kalathil, D., and Shakkottai, S. Learning with safety constraints: Sample complexity of reinforcement learning for constrained mdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7667–7674, 2021.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Janner, M., Du, Y., Tenenbaum, J. B., and Levine, S. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

Ji, J., Zhou, J., Zhang, B., Dai, J., Pan, X., Sun, R., Huang, W., Geng, Y., Liu, M., and Yang, Y. Omnisafe: An infrastructure for accelerating safe reinforcement learning research. *arXiv preprint arXiv:2305.09304*, 2023.

Kamalaruban, P., Huang, Y.-T., Hsieh, Y.-P., Rolland, P., Shi, C., and Cevher, V. Robust reinforcement learning via adversarial training with langevin dynamics. *Advances in Neural Information Processing Systems*, 33:8127–8138, 2020.

LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Liu, Z., Guo, Z., Yao, Y., Cen, Z., Yu, W., Zhang, T., and Zhao, D. Constrained decision transformer for offline safe reinforcement learning. *arXiv preprint arXiv:2302.07351*, 2023.

Luo, Y. and Ma, T. Learning barrier certificates: Towards safe reinforcement learning with zero training-time violations. *Advances in Neural Information Processing Systems*, 34:25621–25632, 2021.

Ma, Y.-A., Chen, Y., Jin, C., Flammarion, N., and Jordan, M. I. Sampling can be faster than optimization. *Proceedings of the National Academy of Sciences*, 116(42): 20881–20885, 2019.

Maeda, S.-i., Watahiki, H., Okada, S., and Koyama, M. Reconnaissance and planning algorithm for constrained mdp. *arXiv preprint arXiv:1909.09540*, 2019.

Mankowitz, D. J., Michi, A., Zhernov, A., Gelmi, M., Selvi, M., Paduraru, C., Leurent, E., Iqbal, S., Lespiau, J.-B., Ahern, A., et al. Faster sorting algorithms discovered using deep reinforcement learning. *Nature*, 618(7964): 257–263, 2023.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Nijkamp, E., Hill, M., Han, T., Zhu, S.-C., and Wu, Y. N. On the anatomy of mcmc-based maximum likelihood learning of energy-based models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5272–5280, 2020.

Peters, J. and Schaal, S. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697, 2008.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

Ray, A., Achiam, J., and Amodei, D. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 7(1):2, 2019.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *Proceedings of International Conference on Machine Learning*, pp. 1889–1897, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Süli, E. and Mayers, D. F. *An introduction to numerical analysis*. Cambridge university press, 2003.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Tessler, C., Mankowitz, D. J., and Mannor, S. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Vuong, Q., Zhang, Y., and Ross, K. W. Supervised policy update for deep reinforcement learning. In *International Conference on Learning Representation*, 2019.

Wang, Y., Zhan, S. S., Jiao, R., Wang, Z., Jin, W., Yang, Z., Wang, Z., Huang, C., and Zhu, Q. Enforcing hard constraints with soft barriers: Safe reinforcement learning in unknown stochastic environments. In *International Conference on Machine Learning*, pp. 36593–36604. PMLR, 2023.

Xie, J., Lu, Y., Gao, R., and Wu, Y. N. Cooperative learning of energy-based model and latent variable model via mcmc teaching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Yang, L., Zheng, Q., and Pan, G. Sample complexity of policy gradient finding second-order stationary points. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 10630–10638, 2021.

Yang, L., Ji, J., Dai, J., Zhang, L., Zhou, B., Li, P., Yang, Y., and Pan, G. Constrained update projection approach to safe policy optimization. *Advances in Neural Information Processing Systems*, 35:9111–9124, 2022.

Yang, L., Huang, Z., Lei, F., Zhong, Y., Yang, Y., Fang, C., Wen, S., Zhou, B., and Lin, Z. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122*, 2023.

Yang, T.-Y., Rosca, J., Narasimhan, K., and Ramadge, P. J. Projection-based constrained policy optimization. *arXiv preprint arXiv:2010.03152*, 2020.

Zhang, L., Shen, L., Yang, L., Chen, S., Yuan, B., Wang, X., and Tao, D. Penalized proximal policy optimization for safe reinforcement learning. *arXiv preprint arXiv:2205.11814*, 2022.

Zhang, Y., Vuong, Q., and Ross, K. First order constrained optimization in policy space. *Advances in Neural Information Processing Systems*, 33:15338–15349, 2020.

Zheng, Y., Li, J., Yu, D., Yang, Y., and Liu, J. Safe offline reinforcement learning with feasibility-guided diffusion model. In *The Twelfth International Conference on Learning Representations*, 2024.

A. Comments on FOCOPS (Zhang et al., 2020)

In this section, we present necessary details about FOCOPS (Zhang et al., 2020) that provides the optimal policy of CPO, which is the foundation for us to propose LAC.

A.1. Optimal Policy of CPO

For a given policy π_{ϕ_k}, firstly, FOCOPS finds an *optimal update policy* $\pi^\dagger$ by solving the optimization problem (2-4) in the non-parameterized policy space, which substitutes $\pi_\phi \in \Pi_\theta$ to $\pi \in \Pi$.

$$\pi^\dagger = \underset{\pi_\phi \in \Pi_\phi}{\text{maximize}} \quad \mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi}}[A^{\pi_{\phi_k}}(s,a)], \tag{17}$$

$$\text{subject to} \quad J_C(\pi_{\phi_k}) + \frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi}}[A_C^{\pi_{\phi_k}}(s,a)] \le b, \tag{18}$$

$$\bar{D}_{\mathrm{KL}}(\pi||\pi_{\phi_k}) \le \delta. \tag{19}$$

If π_{ϕ_k} is feasible, then optimal policy for (17-19) takes the form

$$\pi^\dagger(a|s) = \frac{\pi_{\phi_k}(a|s)}{Z_{\lambda,\nu}(s)} \exp\left(\frac{1}{\lambda}\left(A^{\pi_{\phi_k}}(s,a) - \nu A_C^{\pi_{\phi_k}}(s,a)\right)\right), \tag{20}$$

where $Z_{\lambda,\nu}(s)$ is the partition function which ensures (20) is a valid probability distribution, λ and ν are solutions to the optimization problem:

$$\min_{\lambda,\nu\ge 0}\left\{\lambda\delta + \nu\tilde{b} + \lambda\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}[\log Z_{\lambda,\nu}(s)]\right\} \tag{21}$$

where $\tilde{b} = (1-\gamma)(b - \tilde{J}_C(\pi_{\phi_k}))$.

A.2. Projection

FOCOPS considers to project the policy $\pi^\dagger$ found in the previous step back into the parameterized policy space Π_θ by solving for the closest policy $\pi_\phi \in \Pi_\theta$ to $\pi^\dagger$ in order to obtain $\pi_{\phi_{k+1}}$. Minimizing the loss function:

$$\mathcal{L}(\theta) = \mathbb{E}_{s\sim d^{\pi_{\phi_k}}}\left[D_{\mathrm{KL}}(\pi_\phi||\pi^\dagger)[s]\right] \tag{22}$$

Here $\pi_\phi \in \Pi_\theta$ is some projected policy which we will use to approximate the optimal update policy. We can use first-order methods to minimize this loss function where we make use of the following result:

Corollary A.1. *The gradient of $\mathcal{L}(\theta)$ takes the form*

$$\nabla_\theta\mathcal{L}(\theta) = \mathbb{E}_{s\sim d^{\pi_{\phi_k}}}\left[\nabla_\theta D_{\mathrm{KL}}(\pi_\phi||\pi^\dagger)[s]\right] \tag{23}$$

where

$$\nabla_\theta D_{\mathrm{KL}}(\pi_\phi \parallel \pi^\dagger)[s] = \nabla_\theta D_{\mathrm{KL}}(\pi_\phi||\pi_{\phi_k})[s] - \frac{1}{\lambda}\mathbb{E}_{a\sim\pi_{\phi_k}}\left[\frac{\nabla_\theta\pi_\phi(a|s)}{\pi_{\phi_k}(a|s)}\left(A^{\pi_{\phi_k}}(s,a) - \nu A_C^{\pi_{\phi_k}}(s,a)\right)\right]. \tag{24}$$

B. Proof of Proposition 4.1

Proposition 4.1 *The derivative of $L(\pi^\dagger(\cdot|s),\lambda,\nu)$ w.r.t. ν:*

$$\frac{\partial L(\pi^\dagger,\lambda,\nu)}{\partial\nu} = \tilde{b} - \mathbb{E}_{s\sim d^{\pi_{\phi_k}},a\sim\pi^\dagger(\cdot|s)}[A^{\pi_{\phi_k}}(s,a)].$$

We also provide a proof of (14) as follows.

Proof. First note that $L(\pi^\dagger, \lambda, \nu) = \lambda\delta + \nu\tilde{b} + \lambda\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}[\log Z_{\lambda,\nu}(s)]$. We first calculate the derivative of $\pi^\dagger$ w.r.t. ν,

$$\begin{aligned}\frac{\partial \pi^\dagger(a|s)}{\partial \nu} &= \frac{\pi_{\phi_k}(a|s)}{Z^2_{\lambda,\nu}(s)}\left[Z_{\lambda,\nu}(s)\frac{\partial}{\partial\nu}\exp\left(\frac{1}{\lambda}\left(A^{\pi_{\phi_k}}(s,a) - \nu A_C^{\pi_{\phi_k}}(s,a)\right)\right)\right.\\ &\quad\left. - \exp\left(\frac{1}{\lambda}\left(A^{\pi_{\phi_k}}(s,a) - \nu A_C^{\pi_{\phi_k}}(s,a)\right)\right)\frac{\partial Z_{\lambda,\nu}(s)}{\partial\nu}\right]\\ &= -\frac{A_C^{\pi_{\phi_k}}(s,a)}{\lambda}\pi^\dagger(a|s) - \pi^\dagger(a|s)\frac{\partial \log Z_{\lambda,\nu}(s)}{\partial\nu}\end{aligned}$$

Therefore the derivative of the expectation in the last term of $L(\pi^\star, \lambda, \nu)$ can be written as

$$\begin{aligned}&\frac{\partial}{\partial\nu}\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}[\log Z_{\lambda,\nu}(s)]\\ =&\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}\left[\frac{\partial}{\partial\nu}\left(\frac{\pi^\dagger(a|s)}{\pi_{\phi_k}(a|s)}\log Z_{\lambda,\nu}(s)\right)\right]\\ =&\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}\left[\frac{1}{\pi_{\phi_k}(a|s)}\left(\frac{\partial\pi^\dagger(a|s)}{\partial\nu}\log Z_{\lambda,\nu}(s) + \pi^\dagger(a|s)\frac{\partial\log Z_{\lambda,\nu}(s)}{\partial\nu}\right)\right]\\ =&\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}\left[\frac{\pi^\dagger(a|s)}{\pi_{\phi_k}(a|s)}\left(-\frac{A_C^{\pi_{\phi_k}}(s,a)}{\lambda}\log Z_{\lambda,\nu}(s) - \frac{\partial\log Z_{\lambda,\nu}(s)}{\partial\nu}\log Z_{\lambda,\nu}(s) + \frac{\partial\log Z_{\lambda,\nu}(s)}{\partial\nu}\right)\right]\\ =&\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}\left[-\frac{A_C^{\pi_{\phi_k}}(s,a)}{\lambda}\log Z_{\lambda,\nu}(s) - \frac{\partial\log Z_{\lambda,\nu}(s)}{\partial\nu}\log Z_{\lambda,\nu}(s) + \frac{\partial\log Z_{\lambda,\nu}(s)}{\partial\nu}\right].\end{aligned} \tag{25}$$

Also,

$$\begin{aligned}\frac{\partial Z_{\lambda,\nu}(s)}{\partial\nu} &= \frac{\partial}{\partial\nu}\sum_a \pi_{\phi_k}(a|s)\exp\left(\frac{1}{\lambda}\left(A^{\pi_{\phi_k}}(s,a) - \nu A_C^{\pi_{\phi_k}}(s,a)\right)\right)\\ &= \sum_a -\pi_{\phi_k}(a|s)\frac{A_C^{\pi_{\phi_k}}(s,a)}{\lambda}\exp\left(\frac{1}{\lambda}\left(A^{\pi_{\phi_k}}(s,a) - \nu A_C^{\pi_{\phi_k}}(s,a)\right)\right)\\ &= \sum_a -\frac{A_C^{\pi_{\phi_k}}(s,a)}{\lambda}\frac{\pi_{\phi_k}(a|s)}{Z_{\lambda,\nu}(s)}\exp\left(\frac{1}{\lambda}\left(A^{\pi_{\phi_k}}(s,a) - \nu A_C^{\pi_{\phi_k}}(s,a)\right)\right)Z_{\lambda,\nu}(s)\\ &= -\frac{Z_{\lambda,\nu}(s)}{\lambda}\mathbb{E}_{a\sim\pi^\dagger(\cdot|s)}\left[A_C^{\pi_{\phi_k}}(s,a)\right].\end{aligned} \tag{26}$$

Therefore,

$$\frac{\partial\log Z_{\lambda,\nu}(s)}{\partial\nu} = \frac{\partial Z_{\lambda,\nu}(s)}{\partial\nu}\frac{1}{Z_{\lambda,\nu}(s)} = -\frac{1}{\lambda}\mathbb{E}_{a\sim\pi^\dagger(\cdot|s)}\left[A_C^{\pi_{\phi_k}}(s,a)\right]. \tag{27}$$

Plugging (27) into the last equality in (25) gives us

$$\begin{aligned}\frac{\partial}{\partial\nu}\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}[\log Z_{\lambda,\nu}(s)] &= \mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}\left[-\frac{A_C^{\pi_{\phi_k}}(s,a)}{\lambda}\log Z_{\lambda,\nu}(s) + \frac{A_C^{\pi_{\phi_k}}(s,a)}{\lambda}\log Z_{\lambda,\nu}(s) - \frac{1}{\lambda}A_C^{\pi_{\phi_k}}(s,a)\right]\\ &= -\frac{1}{\lambda}\mathbb{E}_{\substack{s\sim d^{\pi_{\phi_k}} \\ a\sim\pi^\dagger}}\left[A_C^{\pi_{\phi_k}}(s,a)\right].\end{aligned} \tag{28}$$

Combining (28) with the derivatives of the affine term gives us the final desired result. □

C. Convergence of Langevin Policy

In this section, we provide a proof to analyze the convergence of Langevin policy.

C.1. Auxiliary Result from (Cheng & Bartlett, 2018)

Langevin Diffusion. Let us consider sampling from a density

$$p_\star(x) = \exp\{-U(x) + C\}$$

where C is the normalizing term but unknown the normalizing constant, U is known. One way to sample from $p_\star$ is to consider the Langevin diffusion:

$$\begin{aligned} \bar{x}_0 &\sim \bar{p}_0, \\ \mathrm{d}\bar{x}_t &= -\nabla U(\bar{x}_t)\mathrm{d}t + \sqrt{2}\mathrm{d}B_t, \end{aligned} \tag{29}$$

where $\bar{p}_0$ is some initial distribution, and B_t is Brownian motion. The stationary distribution of the above SDE is $p_\star$.

Langevin MCMC Algorithm. For a given initial distribution p_0, and for a given stepsize h, the Langevin MCMC Algorithm is given by the following:

$$\begin{aligned} u_0 &\sim p_0, \\ u_{i+1} &= u_i - h\nabla U(u_i)\mathrm{d}t + \sqrt{2h}z_i, \end{aligned} \tag{30}$$

where $z_i \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,1)$.

For a given initial distribution p_0 and stepsize h, the Discretized Langevin Diffusion is given by the following SDE:

$$\begin{aligned} x_0 &\sim p_0, \\ \mathrm{d}x_t &= -\nabla U(x_{\tau(t)})\mathrm{d}t + \sqrt{2}\mathrm{d}B_t, \end{aligned} \tag{31}$$

where $\tau(t) = \lfloor \frac{t}{h} \rfloor$ (note that $\tau(t)$ is parametrized by h). It is easily verified h that for any i, x_{ih} from (30) is equivalent to u_i in (31).

Theorem C.1 ((Cheng & Bartlett, 2018, Theorem 3)). *Let $U(x)$ be with l Lipschitz continuous gradients and m strong convexity, i.e. for all x: $m\mathbf{I} \preceq \nabla^2 U(x) \preceq l\mathbf{I}$. Let x_t be defined according to (31) with $p_0 = \mathcal{N}(0, \frac{1}{m})$, and let p_t denote the distribution of x_t. Let*

$$h =: \frac{m\epsilon}{16dl^2}, \quad k =: 16\frac{l^2}{m^2}\frac{d\log\frac{dl}{m\epsilon}}{\epsilon}$$

where d is the dimension of sampling space. Then

$$\mathrm{KL}(p_{kh}|p_\star) \le \epsilon.$$

C.2. Application to Langevin Policy

Recall Langevin policy (9)

$$\tilde{a}_i = \tilde{a}_{i-1} + \frac{\eta}{2}\nabla_a \log\pi^\dagger(\tilde{a}_{i-1}|s) + \sqrt{\eta}z_i \tag{32}$$

$$= \tilde{a}_{i-1} - \frac{\eta}{2}E_k(s, \tilde{a}_{i-1}; \phi) + \sqrt{\eta}z_i, \tag{33}$$

where $z_i \sim \mathcal{N}(0, I)$, $\eta > 0$ is the step size, T is total step, and the initial action $\tilde{a}_0 \sim p_0$, p_0 is some initial distribution. Furthermore, we know the Langevin policy is the stationary distribution follows the next stochastic dynamic of $\{A(t)\}_{t\ge 0}$ on the continuous time $t \in [0, +\infty)$,

$$\mathrm{d}A(t) = -\nabla_a E_k(a, s; \phi)\mathrm{d}t|_{a=A(t)} + \sqrt{2}\mathrm{d}B(t), \tag{34}$$

with $A_0 \sim p_0$,and $B(t)$ is Brownian motion at time t.

Let $\pi_i(\cdot|s)$ denotes the distribution of $\tilde{a}_i$ that generates by (32), the according to Theorem C.1, if

$$\frac{\eta}{2} =: \frac{m\epsilon}{16dl^2}, \quad T =: 16\frac{l^2}{m^2}\frac{d\log\frac{dl}{m\epsilon}}{\epsilon}$$

for any state s, we obtain

$$\mathrm{KL}\left(\pi_T(\cdot|s)\|\pi^\dagger(\cdot|s)\right) \le \epsilon.$$

D. Implementation Details for Experiments

We implement CPO, TRPO-L, and PPO-L based on https://github.com/openai/safety-starter-agents, which was released by OpenAI. FOCOPS is implemented based on https://github.com/ymzhang01/focops. RESPO is based on https://github.com/milanganai/milanganai.github.io/tree/main/NeurIPS2023/code. The other baseline methods are implemented based on Omnisafe (Ji et al., 2023) with the default hyperparameters to ensure a fair comparison. For the simulations in all tasks, we employ a neural network consisting of two hidden layers with a size of (256, 256) to represent our Gaussian policies π_ϕ. We conduct 10 runs for MuJoCo tasks and no fewer than 3 runs for Safety Gym tasks to get the mean and standard deviation for both reward and cost value over the policy updates. We run our experiments on Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz with 8 cores.

D.1. Additional Experiment Results

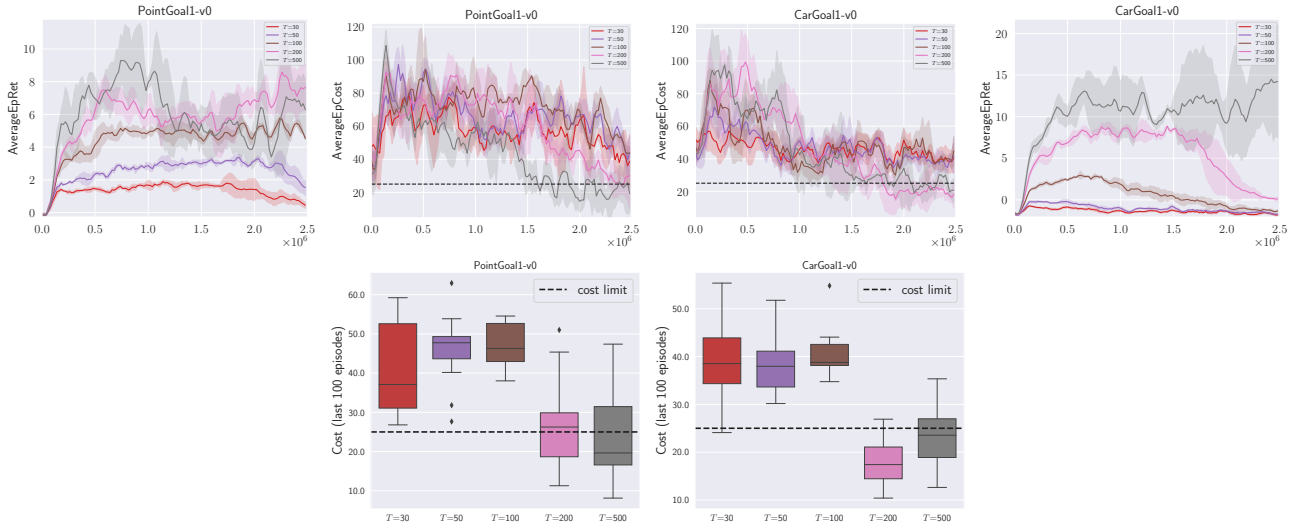


Figure 4. Ablation study of T with noninformative initialization ($\tilde{a}_0 \sim \mathcal{N}(0, 1)$). The box plots show the convergence cost.

Ablation on Initialization Methods. We conduct ablation studies to investigate the non-informative initialization with Langevin policy in LAC. As shown in Figure 4, it requires a large T for LAC with noninformative initialization (i.e., Gaussian distribution) to approach a safe policy, which aligns with our earlier discussion on initialization methods in Section 4.1. Although with a smaller T (i.e., 30,50,100, in this case), LAC with non-informative initialization still improves reward performance and reduces constraint violations. However, we notice that the cost limit is never satisfied with a smaller T. In both PointGoal and CarGoal tasks, the imposed cost limit is only met when the value of T is set to either 200 or 500. Under this condition, LAC achieves a relatively higher reward. However, a larger T is accompanied by more computational cost. In contrast, LAC with informative initialization demonstrates superior performance, which can be attributed to the generator's ability to preserve previously acquired knowledge and information.

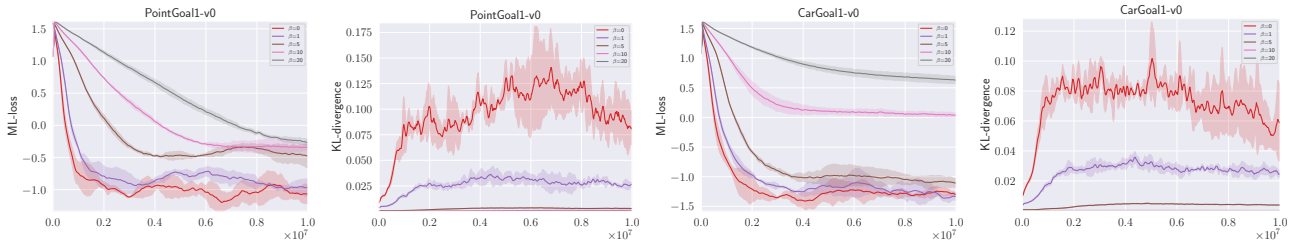


Figure 5. Learning curves for maximum likelihood actor loss (ML-loss) and policy KL-regularization loss (KL-divergence).

Ablation on Policy Regularization. As shown in Figure 5, we find that the second term (KL-divergence) in Eq. (16) is relatively small compared to the first term (vanilla maximum likelihood actor loss). To fully examine the effect of policy

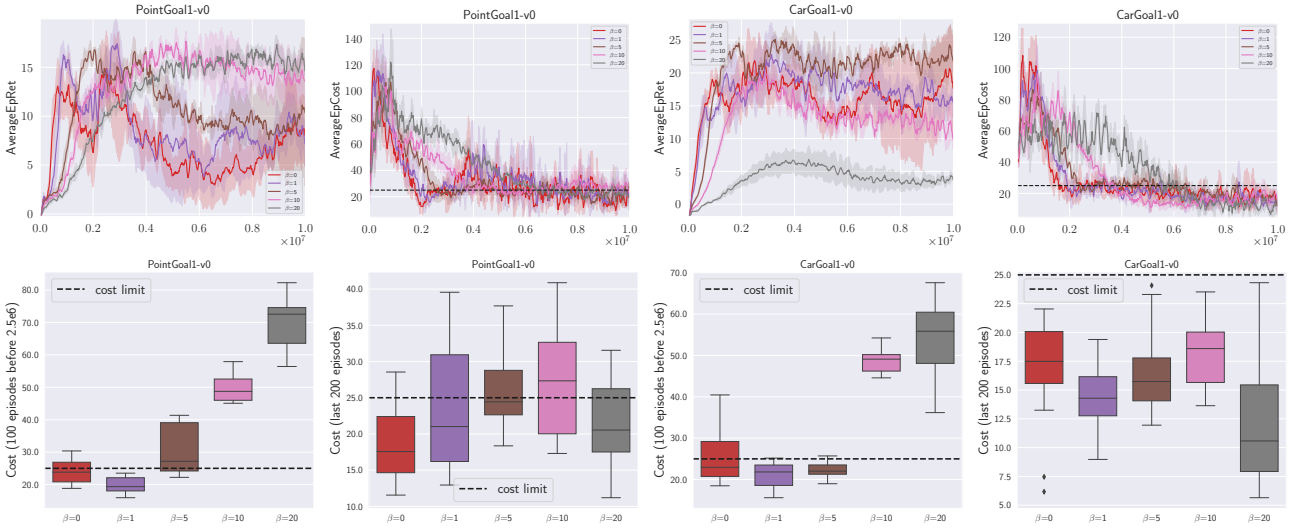


Figure 6. Ablation study of β with policy KL-regularization. The box plots show the convergence cost.

KL-regularization, we conduct an ablation study on hyperparameter β among $\{0,1,5,10,20\}$. In Figure 6, we compare the performance of LAC with ($\beta > 0$) and without ($\beta = 0$) policy KL-regularization. LAC demonstrates increased stability with $\beta = \{10, 20\}$. However, this also results in a slightly slower convergence towards a safe policy. It is also important to point out that the proposed policy KL-regularization measures the distance from the distribution π_ϕ to the target distribution $\pi_{\phi'}$ rather than the current policy distribution π_{ϕ_k}. The rationale behind not penalizing discrepancy between new policy π_ϕ and old policy π_{ϕ_k} during the policy update is rooted in the observation that during the sampling process of Langevin policy 1 the acquired $\tilde{a}_T$ from policy $\pi^\dagger(\cdot|s)$ is close to π_{ϕ_k}, which is attributed to the energy function $E(s,a)$ assigns lower energies to regions in the state-action space that correspond to higher $\log \pi_{\phi_k}(a|s)$ and plays a role of penalizing the distance between distribution $\pi^\dagger(\cdot|s)$ and distribution π_{ϕ_k}.

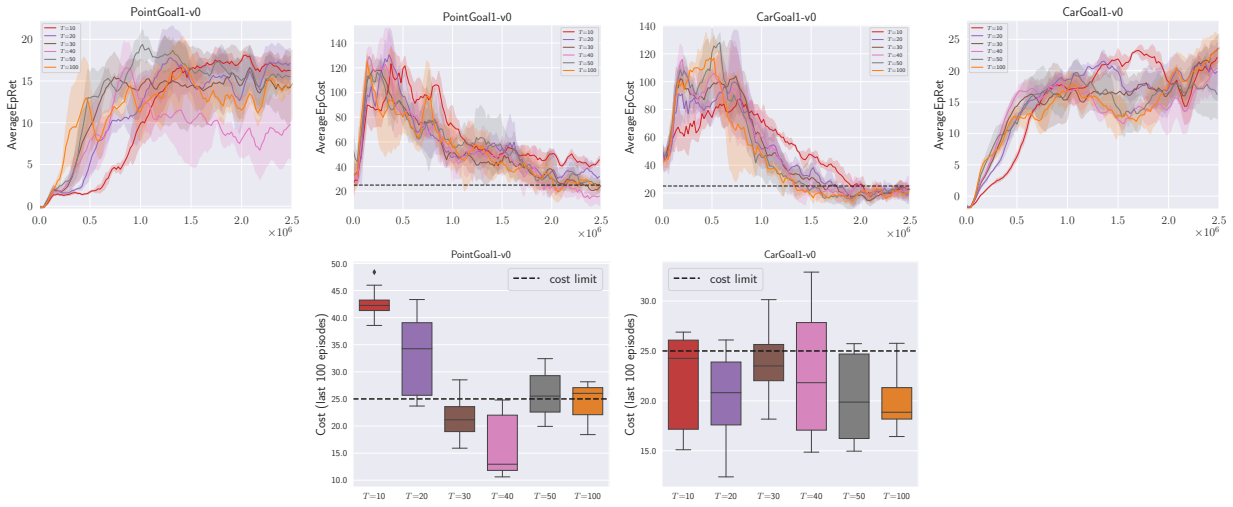


Figure 7. Ablation study of T on selected tasks. The box plots show the convergence cost.

Ablation on Langevin Steps. In Figure 7, we conduct an ablation study on varying Langevin steps T. In PointGoal task, smaller T results in higher cost violations. However, this does not indicates that short-run Langevin policy cannot achieve a comparative performance. As shown in CarGoal task, LAC demonstrates robustness to varying T. Compared to the noninformative initialization (as shown in 4), informative initialization has exhibited its superiority in terms of both performance and efficiency within the LAC.

D.2. Algorithmic Hyperparameters

Table 4. Hyperparameters for MuJoCo tasks.

Hyperparameter	Ant-v3	Swimmer-v3	Humanoid-v3	Humanoid-Circle
State space dimension	27	8	376	376
Action space dimension	8	2	17	17
Activation	$\tanh$	$\tanh$	$\tanh$	$\tanh$
Initial log std	-0.5	-0.5	-0.5	-0.5
Discount for reward γ	0.99	0.99	0.99	0.99
Discount for cost γ_C	0.99	0.99	0.99	0.99
Batch size	5000	5000	5000	5000
Replay buffer	50000	50000	50000	50000
Minibatch size	256	256	256	256
No. of optimization epochs	1	1	1	1
Maximum episode length	1000	1000	1000	1000
Learning rate for actor	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Learning rate for reward/cost critic	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Learning rate for ν	0.01	0.01	0.01	0.01
Temperature λ	0.001	0.001	0.001	0.001
Initial ν	0	0	0	0
Steps for Langevin Policy	30	30	30	30
Stepsize for Langevin Policy	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}

Table 5. Hyperparameters for Safety Gym tasks.

Hyperparameter	PointGoal1-v0	PointButton1-v0	CarGoal1-v0	CarButton1-v0
State space dimension	60	76	72	88
Action space dimension	2	2	2	2
Activation	$\tanh$	$\tanh$	$\tanh$	$\tanh$
Initial log std	-0.5	-0.5	-0.5	-0.5
Discount for reward γ	0.99	0.99	0.99	0.99
Discount for cost γ_C	1	1	1	1
Batch size	20000	20000	20000	20000
Replay buffer	50000	50000	50000	50000
Minibatch size	256	256	256	256
No. of optimization epochs	1	1	1	1
Maximum episode length	1000	1000	1000	1000
Learning rate for actor	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Learning rate for reward/cost critic	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Learning rate for ν	0.01	0.01	0.01	0.01
Temperature λ	0.001	0.001	0.001	0.001
Initial ν	0	0	0	0
Steps for Langevin Policy	30	50	30	50
Stepsize for Langevin Policy	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}