Sample and Computationally Efficient Continuous-Time Reinforcement Learning with General Function Approximation

Runze Zhao[*1], **Yue Yu**[*1], **Adams Yiyue Zhu**[2], **Chen Yang**[1], **Dongruo Zhou**[1]

[1]Indiana University, [2]University of Maryland, College Park

Motivation

Continuous-Time Reinforcement Learning (CTRL) is powerful for real-world problems like robotics and finance. While empirical methods using complex models like neural networks are successful, their theoretical understanding is limited, often confined to simpler models like Linear Quadratic Regulators. This creates a gap between what works in practice and what we can formally guarantee.

We address two fundamental questions:

- **Sample Complexity:** How many measurements are needed to learn a near-optimal policy in CTRL with general function approximation?
- **Computational Efficiency:** Can we design new measurement strategies to reduce the number of expensive policy updates and rollouts without sacrificing performance?

Our contributions

- We provide the first sample complexity for CTRL with general function approximation, using the Eluder dimension.
- We propose two variants of our base algorithm, PURE, that provably reduce the number of policy updates and rollouts.
- Our methods match baseline performance on control tasks and diffusion model fine-tuning, but with significantly lower computational cost.

Problem setup

We model the environment using a continuous-time stochastic differential equation (SDE):

$$dx(t) = f^*(x(t), u(t))dt + g^*(x(t), u(t))dw(t)$$

- $x(t)$: State at time t
- $u(t) = \pi(x(t))$: Control from policy π
- f^*: Unknown system dynamics (drift)
- g^*: Known system dynamics (diffusion)
- $w(t)$: Standard Wiener process

Objective: Find a policy $\pi \in \Pi$ that maximizes the total expected reward:

$$\max_{\pi\in\Pi} R(\pi) := \mathbb{E}\left[\int_0^T b^*(x(t), \pi(x(t)))dt\right]$$

where b^* is the unknown reward function.

Measurement Model: In practice, we can't observe the instantaneous drift f^* directly. We approximate it by observing the state at two close points jointly in time, $x(t)$ and $x(t+\Delta)$, and calculating apply the Euler – Maruyama method [Platen and Bruti-Liberati, 2010] to approximate $y(t) \approx \frac{x(t+\Delta)-x(t)}{\Delta}$.

Complexity Measure: We characterize the complexity of the function classes for the dynamics ($\mathcal{F}$) and reward ($\mathcal{R}$) using the **Distributional Eluder Dimension** ($d_{\mathcal{F}}, d_{\mathcal{R}}$).

Algorithm Overview

PURE (**P**olicy **U**pdate and **R**olling-out **E**fficient CTRL) is a model-based algorithm that learns optimistic estimates of the dynamics (f) and reward (b) within confidence sets and plans accordingly. We propose three variants:

- $\text{PURE}_{\text{base}}$: The foundational algorithm. It establishes sample efficiency by updating the policy at every step, providing a strong theoretical baseline.
- $\text{PURE}_{\text{LowSwitch}}$: Reduces computational cost by updating the policy **only when the current model no longer fits the data well**. This avoids unnecessary, expensive updates.
- $\text{PURE}_{\text{LowRollout}}$: Reduces the number of environment rollouts by taking **multiple measurements within a single trajectory**, balancing data efficiency with the cost of starting a new rollout.

Algorithm $\text{PURE}_{\text{base}}$

1: **Initialize:** Confidence sets for system dynamics ($\mathcal{F}_1$) and reward ($\mathcal{R}_1$).
2: **for** each episode $n = 1, \ldots, N$ **do**
3: **1. Plan with Optimism:** Find the best policy (π_n) and models (f_n, b_n) by maximizing the reward within the current confidence sets.

$$(\pi_n, q_n, f_n, b_n) \leftarrow \operatorname*{argmax}_{\pi\in\Pi, q\in\mathcal{Q}, f\in\mathcal{F}_n, b\in\mathcal{R}_n} R(\pi, q, f, b)$$

4: **2. Data Collection:** Execute the policy π_n to collect a new data point.
5: **3. Update Confidence Sets:** Use the new data to refine the confidence sets $\mathcal{F}_{n+1}$ and $\mathcal{R}_{n+1}$.
6: **Output:** Policy $(\widehat{\pi}, \widehat{q})$ from one of the episodes.

Key assumptions

Our theoretical results rely on standard assumptions in the field. Primarily, we assume the dynamics (f^*), reward (b^*), and policy (π) functions are **Lipschitz continuous**. This means that small changes in the state or control input lead to small changes in the system's evolution and reward, which is a common assumption for ensuring well-behaved systems.

Theoretical guarantee

Our algorithms are backed by rigorous guarantees on performance and efficiency. The following table contrasts their suboptimality gaps with their computational costs (number of updates and rollouts).

Algorithm	Gap	#Update / Rollout
$\text{PURE}_{\text{base}}$	$\widetilde{O}\left(\sqrt{\frac{d_R+d_F}{N}}\right)$	N, N
$\text{PURE}_{\text{LowSwitch}}$	$\widetilde{O}\left(\sqrt{\frac{d_R+d_F}{N}}\right)$	$O(\log N)$, N
$\text{PURE}_{\text{LowRollout}}$	$\widetilde{O}\left(\sqrt{\frac{C_{T,m}}{N}+\frac{m}{N}}\right)$	N/m, N/m

- N: Total number of measurements.
- m: Number of measurements per rollout.
- $C_{\mathcal{T},m}$: A term related to the measurement strategy within a rollout, appearing in the suboptimality gap for $\text{PURE}_{\text{LowRollout}}$.

Exp 1: Fine-Tuning Diffusion Models

- We apply our rollout-efficient strategy, $\text{PURE}_{\text{SEIKO}}$, to fine-tune a Stable Diffusion model on the SEIKO framework [Uehara et al., 2024], aiming to enhance image aesthetic scores.
- **Result:** $\text{PURE}_{\text{SEIKO}}$ achieves aesthetic scores comparable to the baseline while reducing total training time by ≈50%, validating the efficiency of performing multiple measurements per rollout.

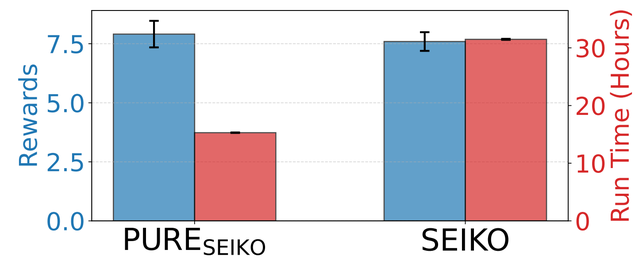

Figure: Summary of the experiment for fine-tuning Diffusion Models.

Exp 2: Continuous-Time Control

- Our low-switching strategy, $\text{PURE}_{\text{ENODE}}$, is integrated into the ENODE baseline [Yildiz et al., 2021] to reduce policy update frequency in Acrobot, Pendulum, and CartPole environments.
- **Result:** $\text{PURE}_{\text{ENODE}}$ matches the baseline's performance with only 25% to 50% of the policy updates, cutting training time by nearly 50% and showcasing the efficacy of our update strategy.

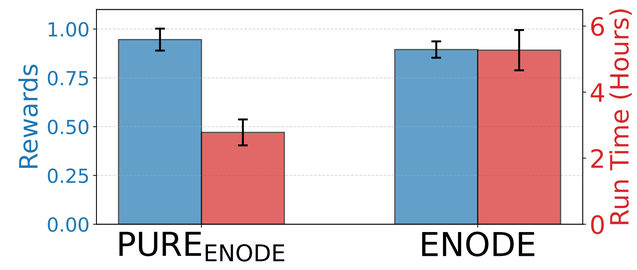

Figure: Comparison on Acrobot

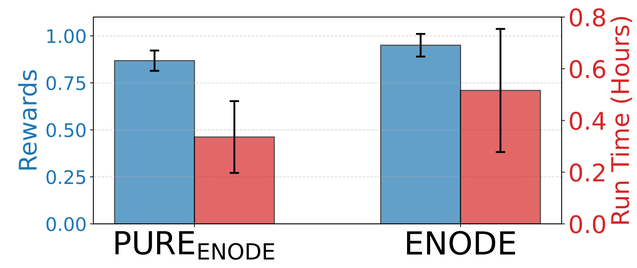

Figure: Comparison on Pendulum

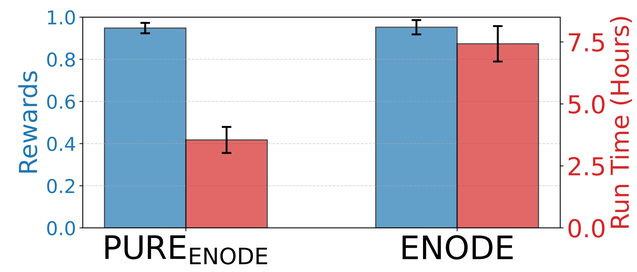

Figure: Comparison on Cart Pole

Conclusion

- We propose PURE, a provably sample and computationally efficient algorithm for continuous-time reinforcement learning (CTRL).
- Empirical results confirm our method achieves comparable performance with significantly fewer policy updates and rollouts across all tested tasks.