Tackling Decision Processes with Non-Cumulative Objectives using Reinforcement Learning

https://arxiv.org/abs/2405.13609

Maximilian Nägele[1,2], Jan Olle[1], Thomas Fösel[2], Remmy Zen[1], and Florian Marquardt[1,2]

[1]Max Planck Institute for the Science of Light, Erlangen, [2]FAU, University of Erlangen-Nuernberg

General scheme

Decision process with non-cumulative objective (NCMDP)

states, rewards, actions $\tilde{s}_t, \tilde{r}_t, a_t$

maximize $\mathbb{E}_\pi\left[f(\tilde{r}_0, \dots, \tilde{r}_{T-1})\right]$

$\tilde{s}_t$ →, $\tilde{r}_t$ →, ← a_t

Markov decision process (MDP)

adapted rewards $r_t = f(\tilde{r}_0, \dots, \tilde{r}_t) - f(\tilde{r}_0, \dots, \tilde{r}_{t-1})$

adapted states $s_t = (\tilde{s}_t, h_t)$

maximize $\mathbb{E}_\pi\left[\sum_{t=0}^{T-1} r_t\right]$

s_t →, r_t →, ← a_t

Markov decision process solver

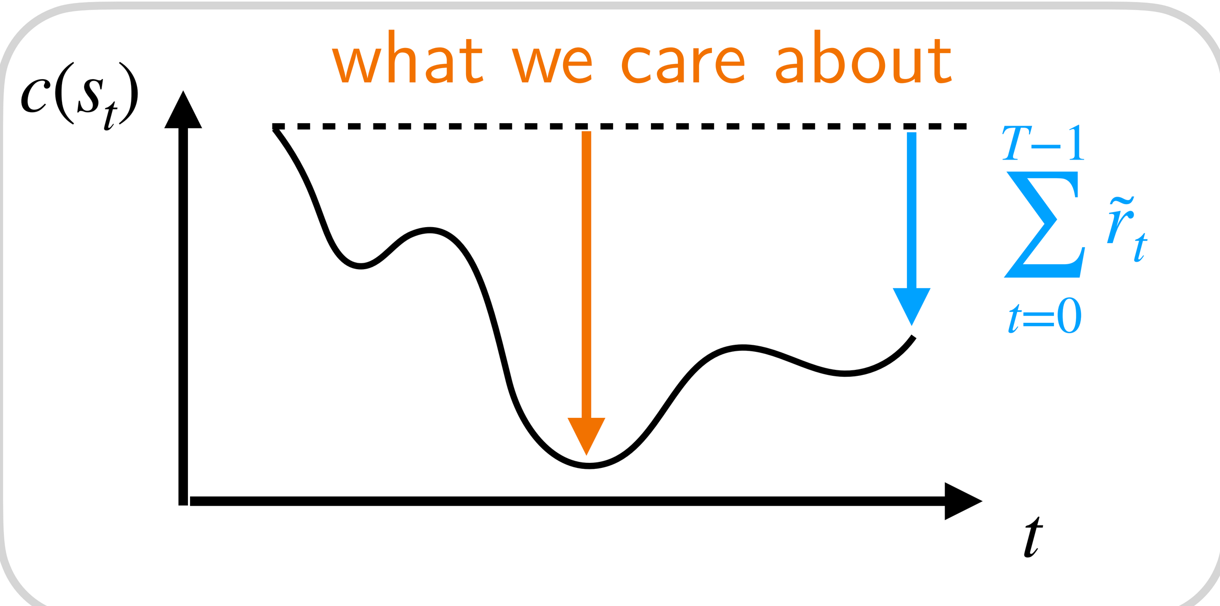
reinforcement learning, dynamic programming, ...

Table 1: Examples of non-cumulative objective functions f.

$f(\tilde{r}_0, \dots, \tilde{r}_{T-1})$	State adaption h_t	Adapted reward r_t
$\max(\tilde{r}_0, \dots, \tilde{r}_{T-1})$	$h_1 = \tilde{r}_0,\ h_{t+1} = \max(h_t, \tilde{r}_t)$	$r_t = \max(0, \tilde{r}_t - h_t)$
$\min(\tilde{r}_0, \dots, \tilde{r}_{T-1})$	$h_1 = \tilde{r}_0,\ h_{t+1} = \min(h_t, \tilde{r}_t)$	$r_t = \min(0, \tilde{r}_t - h_t)$
Sharpe ratio $\frac{\mathrm{MEAN}(\tilde{r}_0,\dots,\tilde{r}_{T-1})}{\mathrm{STD}(\tilde{r}_0,\dots,\tilde{r}_{T-1})}$	$h_0 = \begin{bmatrix}0\\0\\0\end{bmatrix},\ h_{t+1} = \begin{bmatrix}\frac{h_t^{(2)}}{h_t^{(2)}+1}h_t^{(0)} + \frac{1}{h_t^{(2)}+1}\tilde{r}_t \\ \frac{h_t^{(2)}}{h_t^{(2)}+1}h_t^{(1)} + \frac{1}{h_t^{(2)}+1}\tilde{r}_t^2 \\ h_t^{(2)}+1\end{bmatrix}$	$r_t = \frac{h_{t+1}^{(0)}}{\sqrt{h_{t+1}^{(1)} - h_{t+1}^{(0)2}}} - \frac{h_t^{(0)}}{\sqrt{h_t^{(1)} - h_t^{(0)2}}}$
$\max_{k\in[-1,T-1]} \sum_{t=0}^{k} \tilde{r}_t$	$h_0 = 0,\ h_{t+1} = \max(0, h_t - \tilde{r}_t)$	$r_t = \max(0, \tilde{r}_t - h_t)$
$\tilde{r}_0\tilde{r}_1 \dots \tilde{r}_{T-1}$	$h_0 = 1,\ h_{t+1} = \tilde{r}_t h_t$	$r_t = h_{t+1} - h_t$

An improved reward function for discrete optimization problems

Try to find the state with minimum cost $c(\tilde{s}_t)$ reachable from the start state.



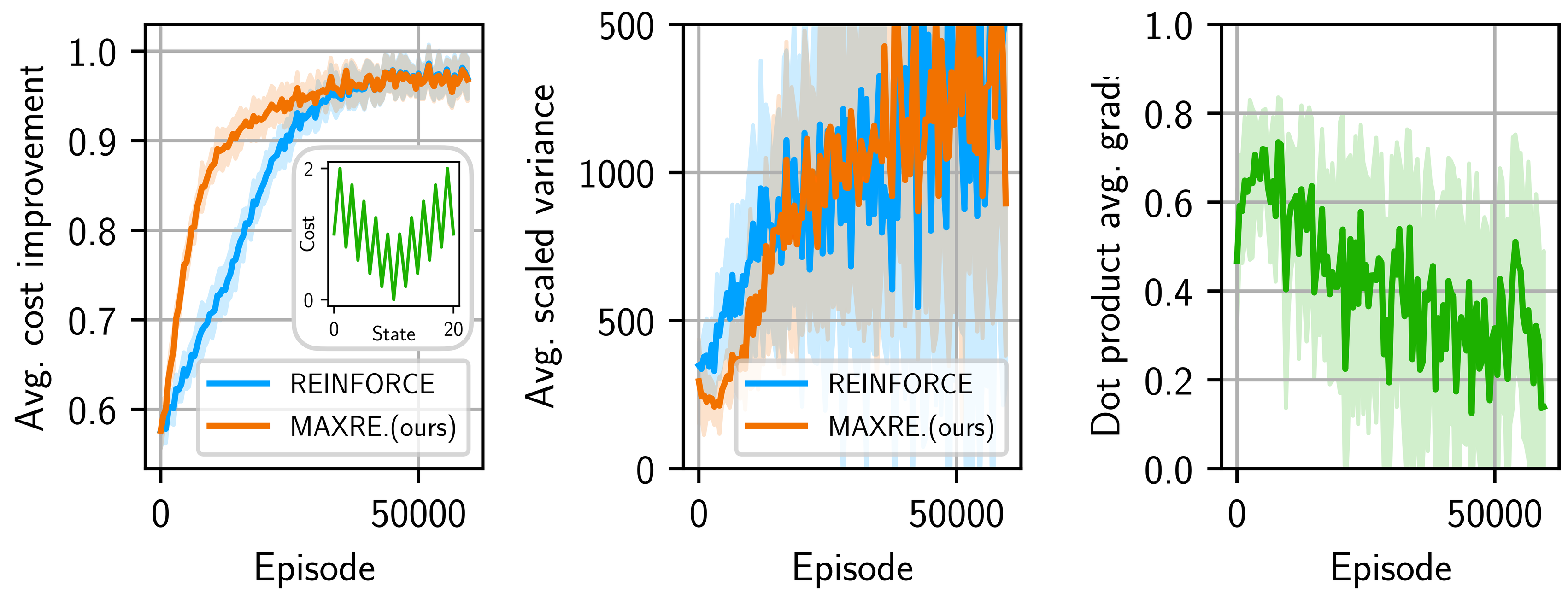
Usual reward: $\tilde{r}_t = c(\tilde{s}_{t+1}) - c(\tilde{s}_t)$

Normal RL minimizes cost at **end** of the trajectory.

We care about the minimum cost at **any point** of the trajectory, i.e.

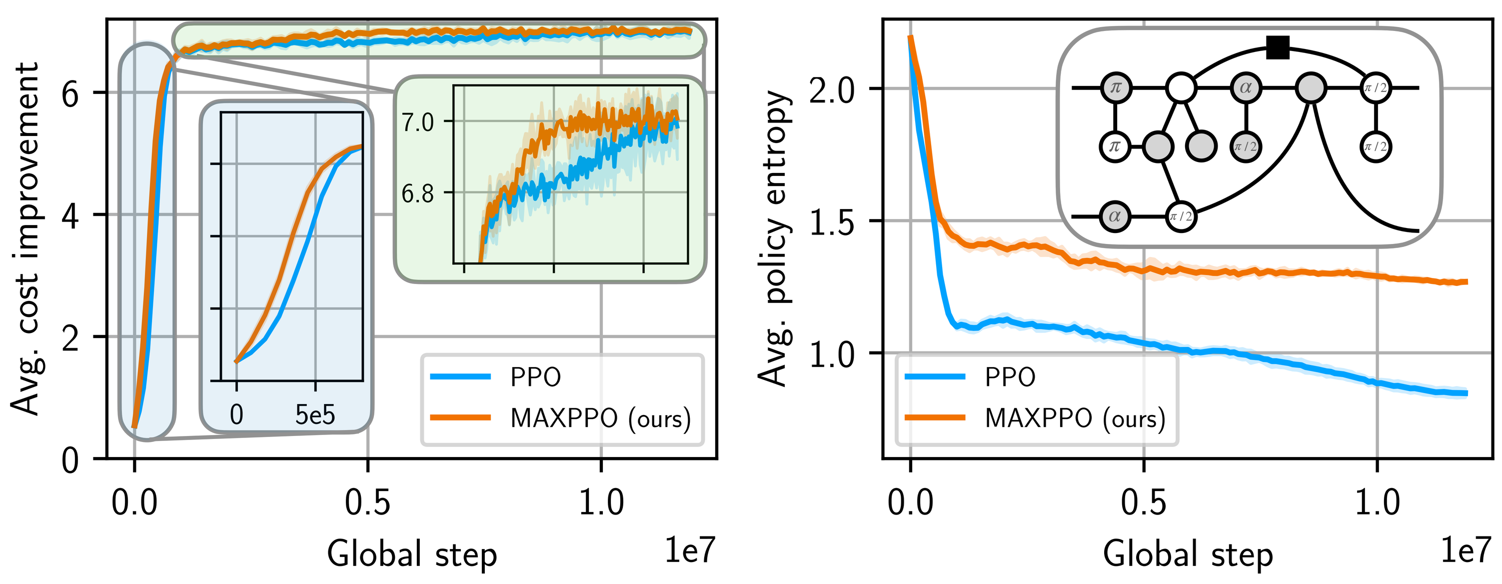
$$\mathbb{E}_\pi\left[c(\tilde{s}_0) - \min_{k\in[0,T-1]} c(\tilde{s}_k)\right] = \mathbb{E}_\pi\left[\max_{k\in[-1,T-1]} \sum_{t=0}^{k} \tilde{r}_t\right]$$

Toy example



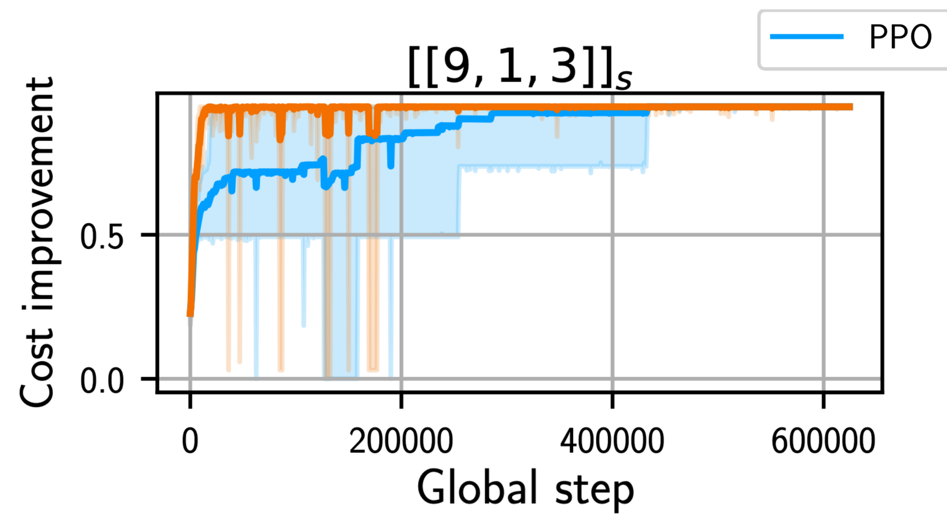
1. No need to learn optimal stopping point.
2. Decreases variance of gradient estimate.
3. Better exploration as cost-increasing actions are not discouraged.

Optimizing ZX-diagrams



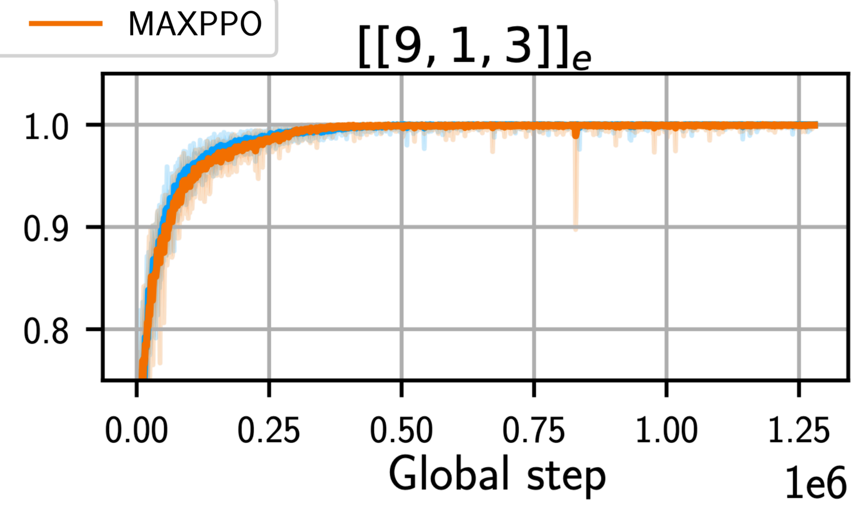
https://arxiv.org/abs/2311.18588

Preparing logical states of quantum error-correction codes



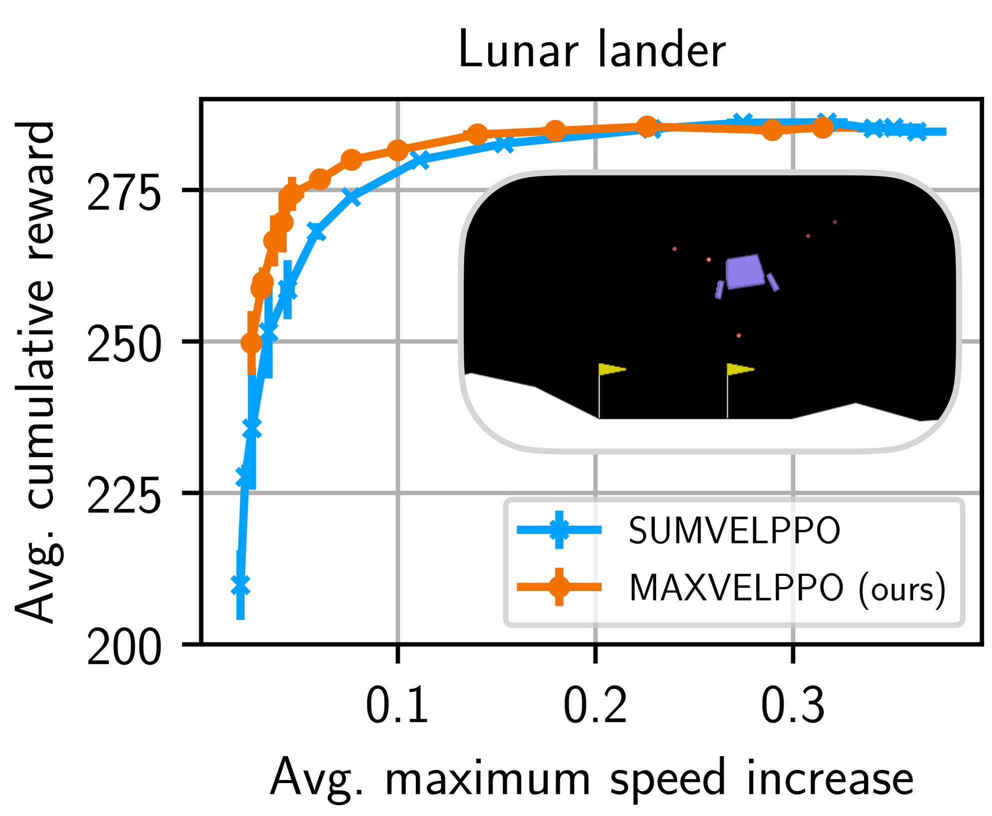
https://arxiv.org/abs/2402.17761

Discovering new quantum error-correction codes



https://arxiv.org/abs/2311.04750

Classical control



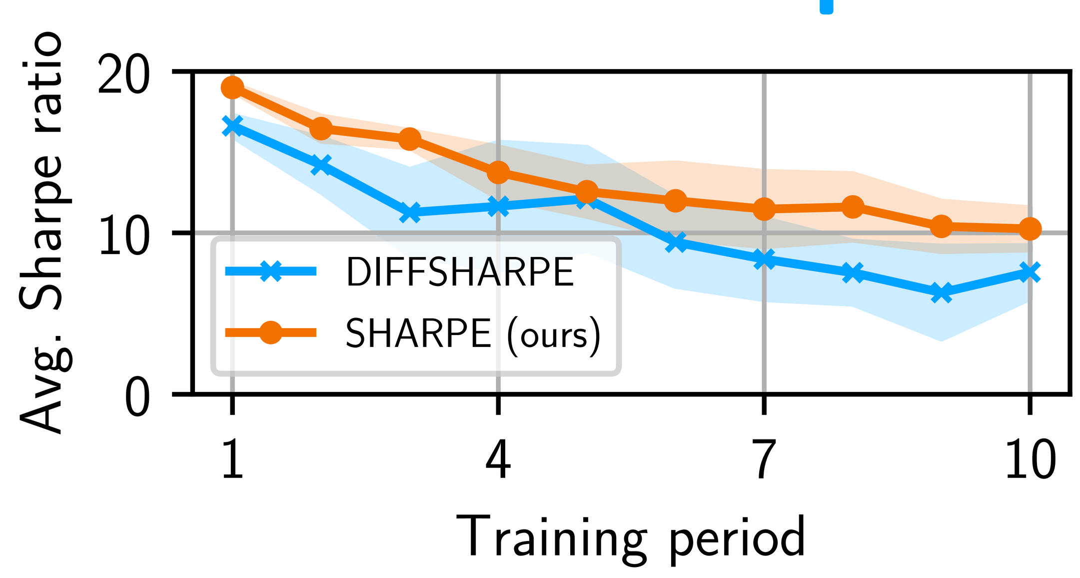
Goal: Land with low speed increase

$$\sum_{t=0}^{T-1} \tilde{r}_t - c\max(v_0, \dots, v_{T-1})$$

$$\sum_{t=0}^{T-1} \left(\tilde{r}_t - c v_t^2\right)$$

Similar application: Robotics

Portfolio optimization



Trade on S&P500 indices

Goal: Maximize SHARPE ratio $\mathbb{E}_\pi\left[\frac{\mathrm{MEAN}(p_t)}{\mathrm{STD}(p_t)}\right]$

(DIFFSHARPE is cumulative approximation)