LEARNING AND PLANNING IN AVERAGE-REWARD MDPS

Yi Wan*, Abhishek Naik*, Richard S. Sutton

{wan6, anaik1, rsutton}@ualberta.ca

ICML 2021

OUTLINE

- Contributions
- Background
  - Problem setting
  - Related work
- Algorithms and Experiments
  - Control
  - Prediction
  - Centering
- Conclusions

CONTRIBUTIONS

A family of average-reward learning and planning algorithms, including:

CONTRIBUTIONS

A family of average-reward learning and planning algorithms, including:

1. The first general proven-convergent off-policy model-free *control* algorithm without reference states

CONTRIBUTIONS

A family of average-reward learning and planning algorithms, including:

1. The first general proven-convergent off-policy model-free *control* algorithm without reference states
2. The first proven-convergent off-policy model-free *prediction* algorithm

CONTRIBUTIONS

A family of average-reward learning and planning algorithms, including:

1. The first general proven-convergent off-policy model-free *control* algorithm without reference states

2. The first proven-convergent off-policy model-free *prediction* algorithm

3. A general technique to estimate the actual value function rather than the value function plus an offset

PROBLEMS AND OBJECTIVES

PROBLEMS AND OBJECTIVES

Episodic problems

PROBLEMS AND OBJECTIVES

Episodic problems

- Total-reward objective
- Discounted objective

PROBLEMS AND OBJECTIVES

Episodic problems

- Total-reward objective
- Discounted objective

Continuing problems

PROBLEMS AND OBJECTIVES

Episodic problems

- Total-reward objective
- Discounted objective

Continuing problems

- Discounted objective
- Average-reward objective

PROBLEMS AND OBJECTIVES

Episodic problems

- Total-reward objective
- Discounted objective

Continuing problems

- Discounted objective
- Average-reward objective

BACKGROUND

PROBLEM SETTING

BACKGROUND

- Finite MDPs

BACKGROUND

- Finite MDPs
- Tabular representation

BACKGROUND

- Finite MDPs
- Tabular representation

Reward rate

$$r(\pi, s) \doteq \lim_{n \to \infty} \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}[R_t \mid S_0 = s, A_{0:t-1} \sim \pi]$$

BACKGROUND

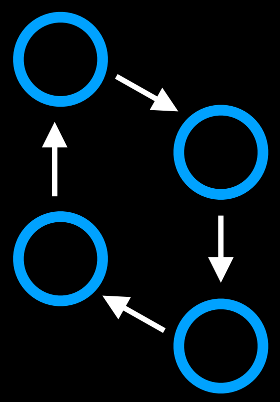
- Finite MDPs
- Tabular representation

Reward rate $$r(\pi, s) \doteq \lim_{n \to \infty} \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}[R_t \mid S_0 = s, A_{0:t-1} \sim \pi] \doteq r(\pi)$$

(Recurrent MDP)

BACKGROUND

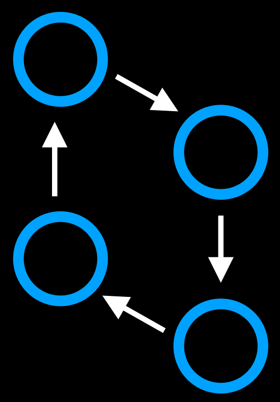
- Finite MDPs
- Tabular representation



Reward rate $$r(\pi, s) \doteq \lim_{n \to \infty} \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}[R_t \mid S_0 = s, A_{0:t-1} \sim \pi] \doteq r(\pi)$$

(Recurrent MDP)

BACKGROUND



- Finite MDPs
- Tabular representation

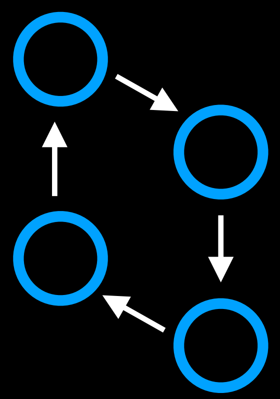
Reward rate $$r(\pi, s) \doteq \lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}[R_t \,|\, S_0 = s, A_{0:t-1} \sim \pi] \qquad \doteq r(\pi)$$

(Recurrent MDP)

$$r_*(s) \doteq \sup_{\pi} r(\pi, s)$$

BACKGROUND

- Finite MDPs
- Tabular representation



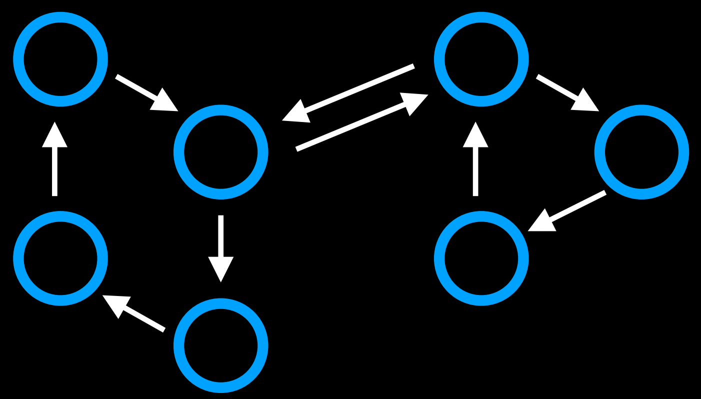
Reward rate $$r(\pi, s) \doteq \lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}[R_t \mid S_0 = s, A_{0:t-1} \sim \pi] \quad \doteq r(\pi)$$

(Recurrent MDP)

$$r_*(s) \doteq \sup_\pi r(\pi, s) \quad \doteq r_*$$ (Communicating MDP)

BACKGROUND

- Finite MDPs
- Tabular representation



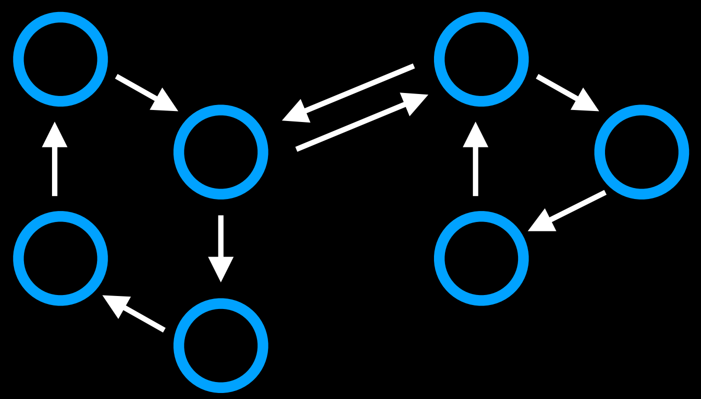
Reward rate $r(\pi, s) \doteq \lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}[R_t \mid S_0 = s, A_{0:t-1} \sim \pi] \doteq r(\pi)$

(Recurrent MDP)

$$r_*(s) \doteq \sup_{\pi} r(\pi, s) \doteq r_*$$

(Communicating MDP)

BACKGROUND



- Finite MDPs
- Tabular representation

Reward rate $$r(\pi, s) \doteq \lim_{n \to \infty} \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}[R_t \mid S_0 = s, A_{0:t-1} \sim \pi] \quad \doteq r(\pi)$$

(Recurrent MDP)

$$r_*(s) \doteq \sup_\pi r(\pi, s) \quad \doteq r_*$$

(Communicating MDP)

Differential value function $$v_\pi(s) \doteq \lim_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} \sum_{t=1}^{k} \mathbb{E}[R_t - r(\pi) \mid S_0 = s, A_{0:t-1} \sim \pi] \quad \forall s$$

BACKGROUND

BELLMAN EQUATIONS

BELLMAN EQUATIONS

Evaluation $$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a) \left[r - \bar{r} + v(s') \right] \quad \forall s$$

BELLMAN EQUATIONS

Evaluation

$$v(s) = \sum_{a} \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + v(s')\right] \quad \forall s$$

Optimality

$$q(s,a) = \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + \max_{a'} q(s', a')\right] \quad \forall s, a$$

BELLMAN EQUATIONS

Evaluation $$v(s) = \sum_{a} \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + v(s')\right] \quad \forall s$$

Optimality $$q(s, a) = \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + \max_{a'} q(s', a')\right] \quad \forall s, a$$

If the MDP is recurrent, the solution of $\bar{r}$ is unique, and the solution of v or q is unique up to an additive constant.

CLASSIFICATION OF ALGORITHMS

CLASSIFICATION OF ALGORITHMS

Learning algorithms

CLASSIFICATION OF ALGORITHMS

On-/off-policy   Learning algorithms

CLASSIFICATION OF ALGORITHMS

On-/off-policy

Learning algorithms

Planning algorithms

CLASSIFICATION OF ALGORITHMS

On-/off-policy

Learning algorithms

Planning algorithms

} Combined learning and planning algorithms

BACKGROUND

Average-reward *learning + combined* algorithms	Prediction	Control
On-policy		
Off-policy		

Legend: Tabular, *Function Approximation*, Missing theoretical results, **Ours**

BACKGROUND

Average-reward *learning + combined* algorithms	Prediction	Control
On-policy	*Average Cost TD (1999)* *LSTD (2002)*	
Off-policy		

Legend: Tabular, *Function Approximation*, Missing theoretical results, **Ours**

BACKGROUND

Average-reward *learning + combined* algorithms	**Prediction**	**Control**
On-policy	*Average Cost TD (1999)* *LSTD (2002)*	*Actor-critic (2000, 2009)* UCRL2 (2010) *Politex (2019)*
Off-policy		

Legend: Tabular, *Function Approximation*, Missing theoretical results, **Ours**

BACKGROUND

Average-reward *learning + combined* algorithms	**Prediction**	**Control**
On-policy	*Average Cost TD (1999)* *LSTD (2002)*	*Actor-critic (2000, 2009)* UCRL2 (2010) *Politex (2019)*
Off-policy	*Wen et al. (2020)* *GradientDICE (2020)*	

Legend: Tabular, *Function Approximation*, Missing theoretical results, **Ours**

BACKGROUND

Average-reward *learning + combined* algorithms	**Prediction**	**Control**
On-policy	*Average Cost TD (1999)* *LSTD (2002)*	*Actor-critic (2000, 2009)* UCRL2 (2010) *Politex (2019)*
Off-policy	*Wen et al. (2020)* *GradientDICE (2020)*	R-learning (1993) Singh (1994) RVI Q-learning (2001) Gosavi (2004)

Legend: Tabular, *Function Approximation*, Missing theoretical results, **Ours**

BACKGROUND

Average-reward *learning + combined* algorithms	Prediction	Control
On-policy	*Average Cost TD (1999)* *LSTD (2002)*	*Actor-critic (2000, 2009)* UCRL2 (2010) *Politex (2019)*
Off-policy	*Wen et al. (2020)* *GradientDICE (2020)* **Differential TD-learning**	R-learning (1993) Singh (1994) RVI Q-learning (2001) Gosavi (2004) **Differential Q-learning**

Legend: Tabular, *Function Approximation*, Missing theoretical results, **Ours**

BACKGROUND

Average-reward *planning* algorithms

BACKGROUND

Average-reward *planning* algorithms

- Value iteration (Bellman 1957)
- Policy iteration (Howard 1960)
- Relative value iteration (White 1963)

BACKGROUND

Average-reward *planning* algorithms

- Value iteration (Bellman 1957)
- Policy iteration (Howard 1960)
- Relative value iteration (White 1963)

Non-incremental

BACKGROUND

Average-reward *planning* algorithms

- Value iteration (Bellman 1957)
- Policy iteration (Howard 1960)
- Relative value iteration (White 1963)

Non-incremental

- Jalali and Ferguson (1990)
- RVI Q-planning (Abounadi et al. 2001)
- Linear Programming Methods (e.g., Wang 2017)

BACKGROUND

Average-reward *planning* algorithms

- Value iteration (Bellman 1957)
- Policy iteration (Howard 1960)
- Relative value iteration (White 1963)

Non-incremental

- Jalali and Ferguson (1990)
- RVI Q-planning (Abounadi et al. 2001)
- Linear Programming Methods (e.g., Wang 2017)

Incremental

BACKGROUND

Average-reward *planning* algorithms

- Value iteration (Bellman 1957)
- Policy iteration (Howard 1960)
- Relative value iteration (White 1963)

Non-incremental

- Jalali and Ferguson (1990)
- RVI Q-planning (Abounadi et al. 2001)
- Linear Programming Methods (e.g., Wang 2017)
- **Differential TD-planning, Differential Q-planning**

Incremental

ALGORITHM MOTIVATION

ALGORITHM MOTIVATION

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

ALGORITHM MOTIVATION

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta(R_{t+1} - \bar{R}_t)$$

ALGORITHM MOTIVATION

new_estimate = old_estimate + stepsize*(target - old_estimate)

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta(R_{t+1} - \bar{R}_t)$$

ALGORITHM MOTIVATION

new_estimate = old_estimate + stepsize*(target - old_estimate)

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta(\underbrace{R_{t+1} - \bar{R}_t}_{\text{Conventional error}})$$

ALGORITHM MOTIVATION

new_estimate = old_estimate + stepsize*(target - old_estimate)

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta(\underbrace{R_{t+1} - \bar{R}_t}_{\text{Conventional error}})$$

Restricted to the on-policy setting

ALGORITHM MOTIVATION

new_estimate = old_estimate + stepsize*(target - old_estimate)

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta(\underbrace{R_{t+1} - \bar{R}_t}_{\text{Conventional error}})$$

Conventional error

Restricted to the on-policy setting

$$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + v(s')\right] \quad \forall s$$

ALGORITHM MOTIVATION

new_estimate = old_estimate + stepsize*(target - old_estimate)

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta\underbrace{(R_{t+1} - \bar{R}_t)}_{\text{Conventional error}}$$

Restricted to the on-policy setting

$$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + v(s')\right] \quad \forall s$$

$$\bar{r} = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - v(s) + v(s')\right] \quad \forall s$$

ALGORITHM MOTIVATION

new_estimate = old_estimate + stepsize*(target - old_estimate)

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta(\underbrace{R_{t+1} - \bar{R}_t}_{\text{Conventional error}})$$

Restricted to the on-policy setting

$$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + v(s')\right] \quad \forall s$$

$$\bar{r} = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - v(s) + v(s')\right] \quad \forall s$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta\big(R_{t+1} - V(S_t) + V(S_{t+1}) - \bar{R}_t\big)$$

ALGORITHM MOTIVATION

new_estimate = old_estimate + stepsize*(target - old_estimate)

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta(\underbrace{R_{t+1} - \bar{R}_t}_{\text{Conventional error}})$$

Restricted to the on-policy setting

$$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + v(s')\right] \quad \forall s$$

$$\bar{r} = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - v(s) + v(s')\right] \quad \forall s$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta\left(R_{t+1} - V(S_t) + V(S_{t+1}) - \bar{R}_t\right)$$

ALGORITHM MOTIVATION

new_estimate = old_estimate + stepsize*(target - old_estimate)

$$\ldots, S_t, A_t, R_{t+1}, S_{t+1}, \ldots$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta\underbrace{(R_{t+1} - \bar{R}_t)}_{\text{Conventional error}}$$

Restricted to the on-policy setting

$$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - \bar{r} + v(s')\right] \quad \forall s$$

$$\bar{r} = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[r - v(s) + v(s')\right] \quad \forall s$$

$$\bar{R}_{t+1} = \bar{R}_t + \beta\underbrace{\left(R_{t+1} - V(S_t) + V(S_{t+1}) - \bar{R}_t\right)}_{\text{TD error}}$$

DIFFERENTIAL Q-LEARNING

DIFFERENTIAL Q-LEARNING

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \max_a Q_t(S_{t+1}, a) - Q_t(S_t, A_t)$$

DIFFERENTIAL Q-LEARNING

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \max_a Q_t(S_{t+1}, a) - Q_t(S_t, A_t)$$

$$Q_{t+1}(S_t, A_t) \doteq Q_t(S_t, A_t) + \alpha_t \delta_t$$

$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta \alpha_t \delta_t$$

DIFFERENTIAL Q-LEARNING

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \max_a Q_t(S_{t+1}, a) - Q_t(S_t, A_t)$$

$$Q_{t+1}(S_t, A_t) \doteq Q_t(S_t, A_t) + \alpha_t \delta_t$$

$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta \alpha_t \delta_t$$

Theorem 1 (informal)

If the Bellman optimality equation has a unique solution for r^* and a unique solution for q^* up to an additive constant, under Borkar's (1998) asynchronous stochastic-approximation assumptions, Differential Q-learning algorithm converges a.s.:

- $\bar{R}_t$ to r_*,
- Q_t to a solution of the Bellman optimality equation, and
- $r(\pi_t, s)$ to r_* for all s where π_t is a greedy policy w.r.t. Q_t.

PSEUDOCODE

Algorithm 1: Differential Q-learning (one-step off-policy control)

Input: The policy b to be used (e.g., ϵ-greedy)

Algorithm parameters: step size α, η

Initialize $Q(s, a)\ \forall s, a$; $\bar{R}$ arbitrarily (e.g., to zero)

Obtain initial S

while *still time to train* **do**

- $A \leftarrow$ action given by b for S
- Take action A, observe R, S'
- $\delta = R - \bar{R} + \max_a Q(S', a) - Q(S, A)$
- $Q(S, A) = Q(S, A) + \alpha\delta$
- $\bar{R} = \bar{R} + \eta\alpha\delta$
- $S = S'$

end

return Q

PSEUDOCODE

Algorithm 1: Differential Q-learning (one-step off-policy control)

Input: The policy b to be used (e.g., ϵ-greedy)
Algorithm parameters: step size α, η

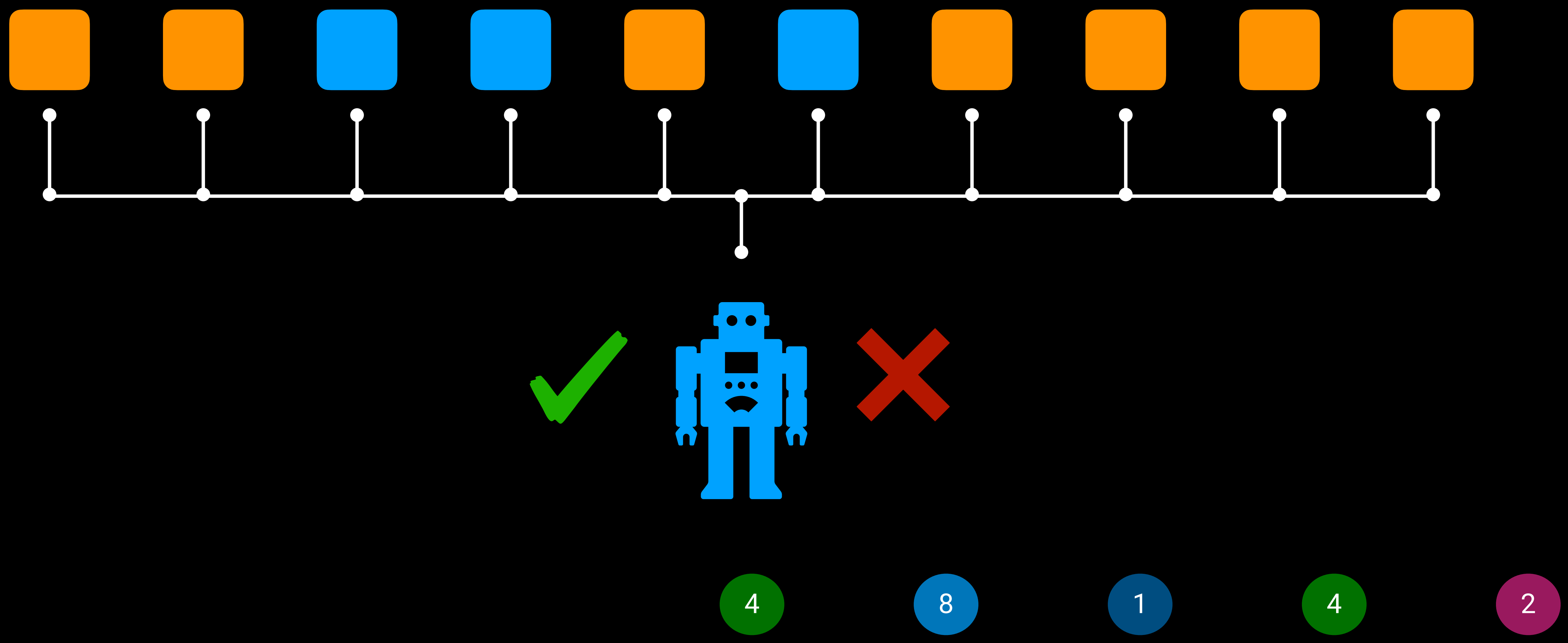
```
Initialize Q(s, a) ∀s, a; R̄ arbitrarily (e.g., to zero)
Obtain initial S
while still time to train do
    A ← action given by b for S
    Take action A, observe R, S'
    δ = R − R̄ + max_a Q(S', a) − Q(S, A)
    Q(S, A) = Q(S, A) + αδ
    R̄ = R̄ + ηαδ
    S = S'
end
return Q
```

RVI Q-learning

$$Q_{t+1}(S_t, A_t) = Q_t(S_t, A_t) + \alpha_t\Big(R_{t+1} - f(Q_t) + \max_a Q_t(S_{t+1}, a) - Q_t(S_t, A_t)\Big)$$

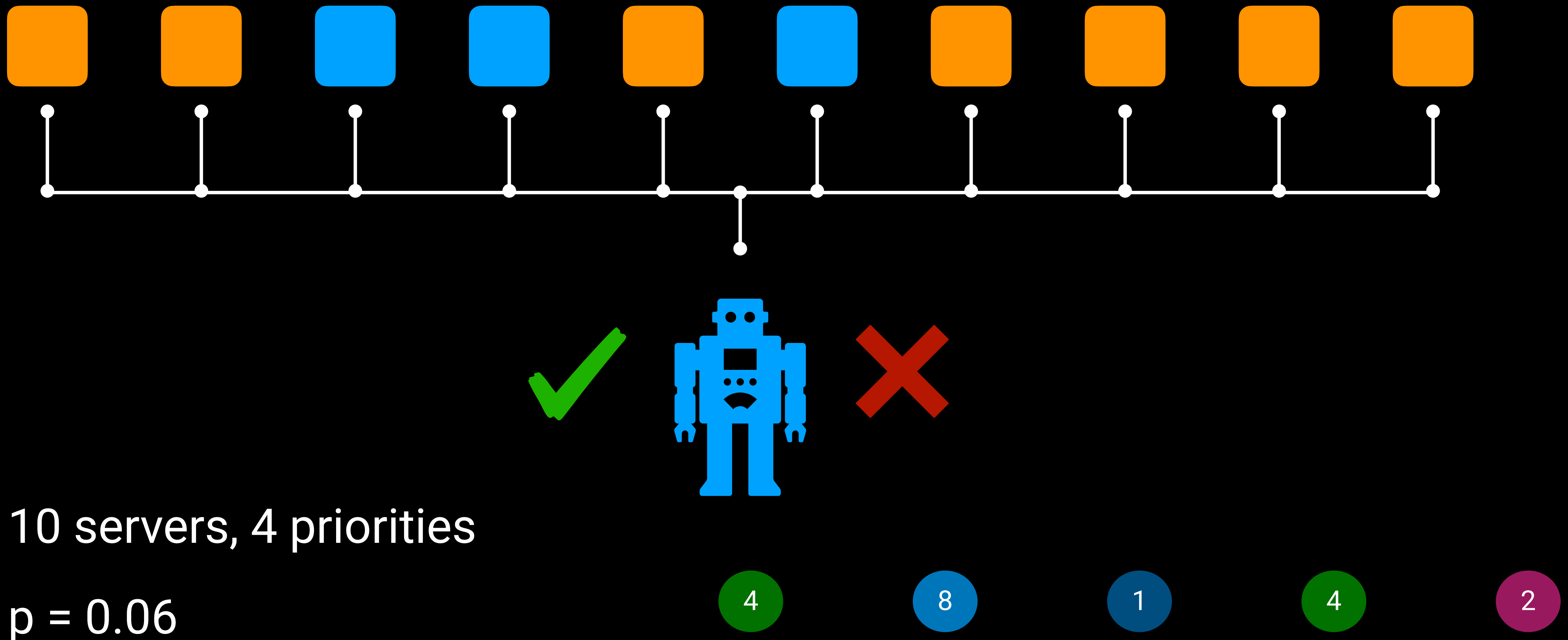
DOMAIN

- Access Control Queueing Task (Sutton & Barto 2018, Ch.10)



DOMAIN

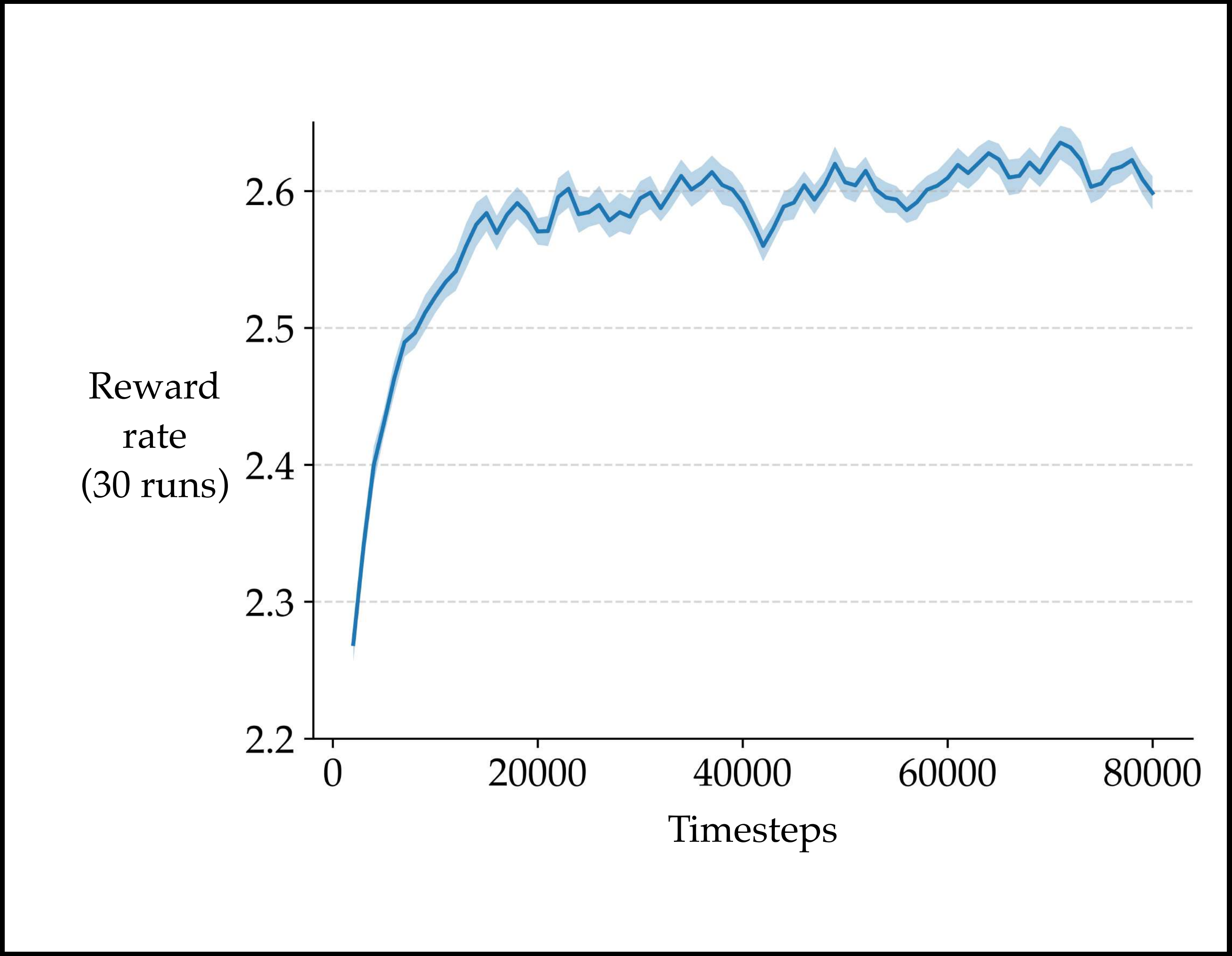
- Access Control Queueing Task (Sutton & Barto 2018, Ch.10)



- 10 servers, 4 priorities
- $p = 0.06$

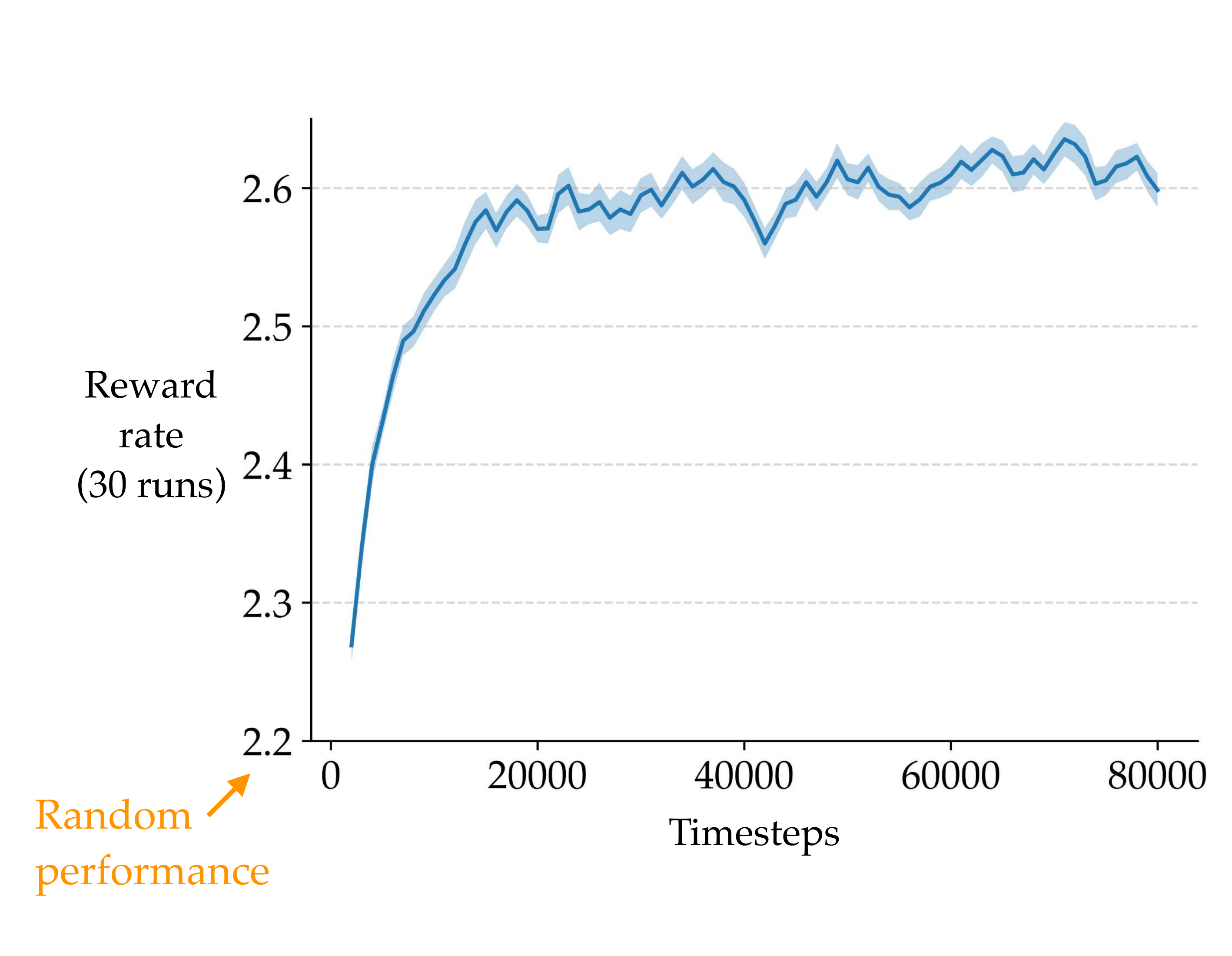
PARAMETERS AND SAMPLE LEARNING CURVE

- $\alpha \in \{0.0015625,\ 0.00625,\ 0.025,\ 0.1,\ 0.4\}$
- $\eta \in \{0.125,\ 0.25,\ 0.5,\ 1,\ 2\}$
- 80,000 steps
- 30 runs
- $\epsilon = 0.1$



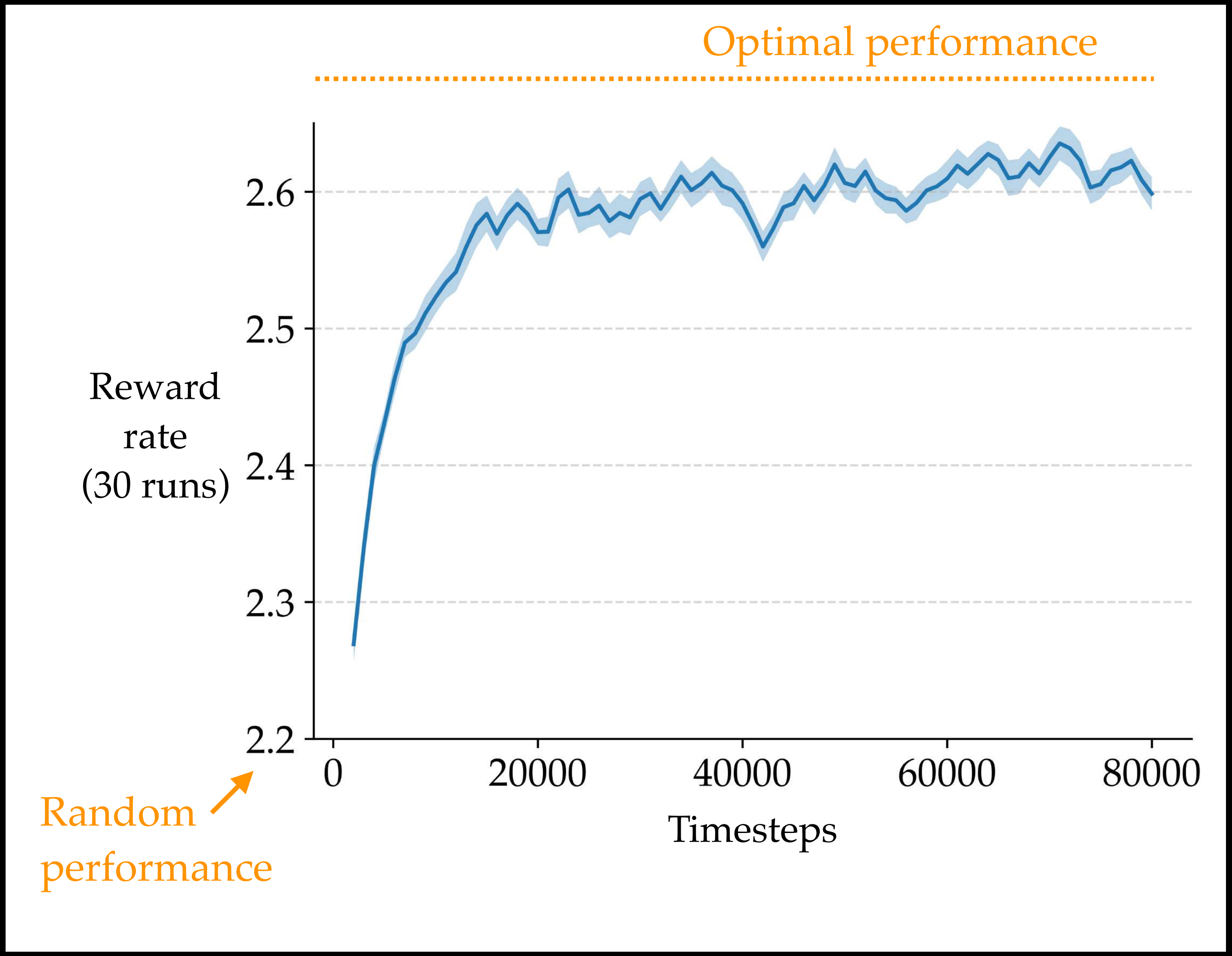
PARAMETERS AND SAMPLE LEARNING CURVE

- $\alpha \in \{0.0015625, 0.00625, 0.025, 0.1, 0.4\}$
- $\eta \in \{0.125, 0.25, 0.5, 1, 2\}$
- 80,000 steps
- 30 runs
- $\epsilon = 0.1$

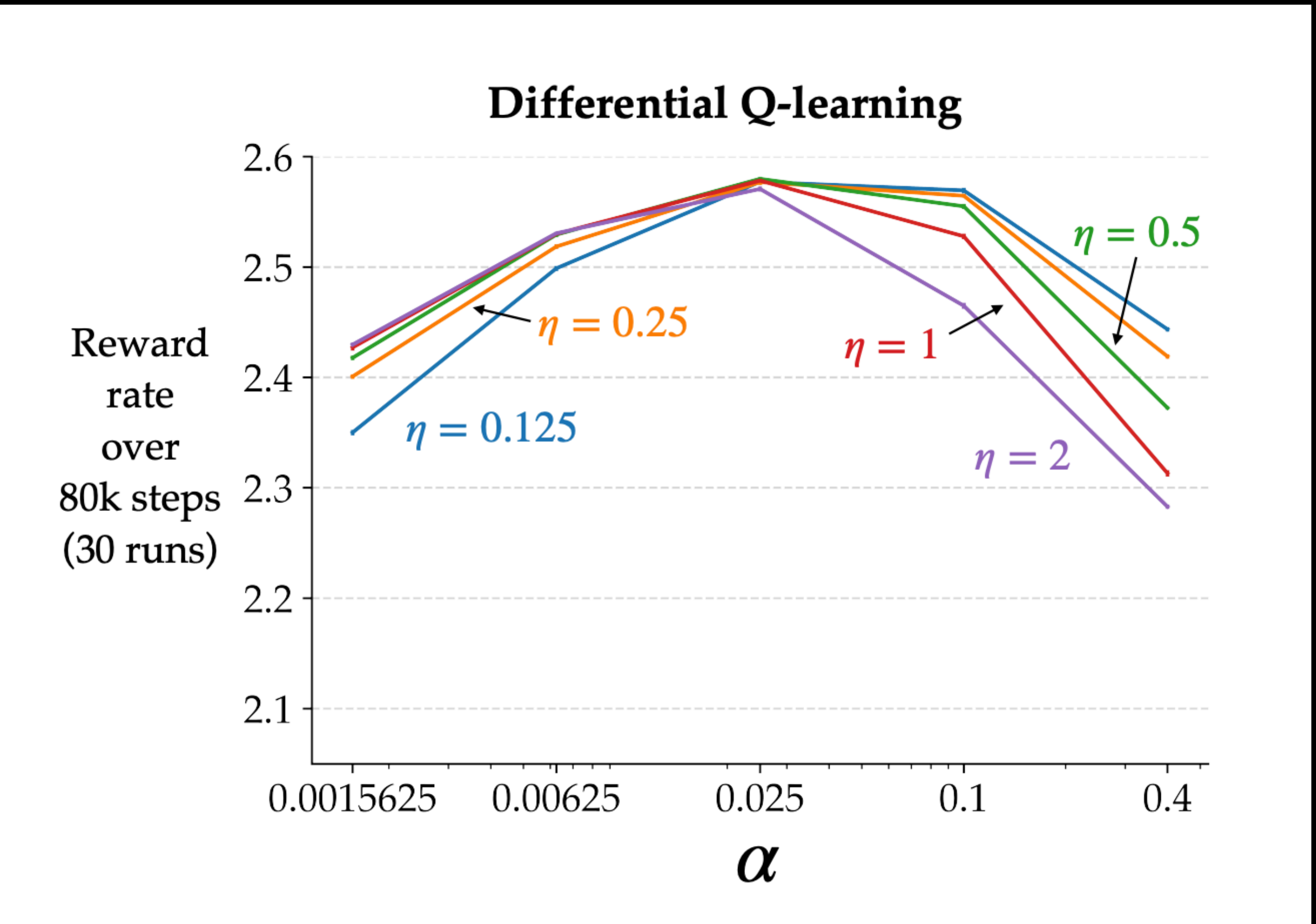


PARAMETERS AND SAMPLE LEARNING CURVE

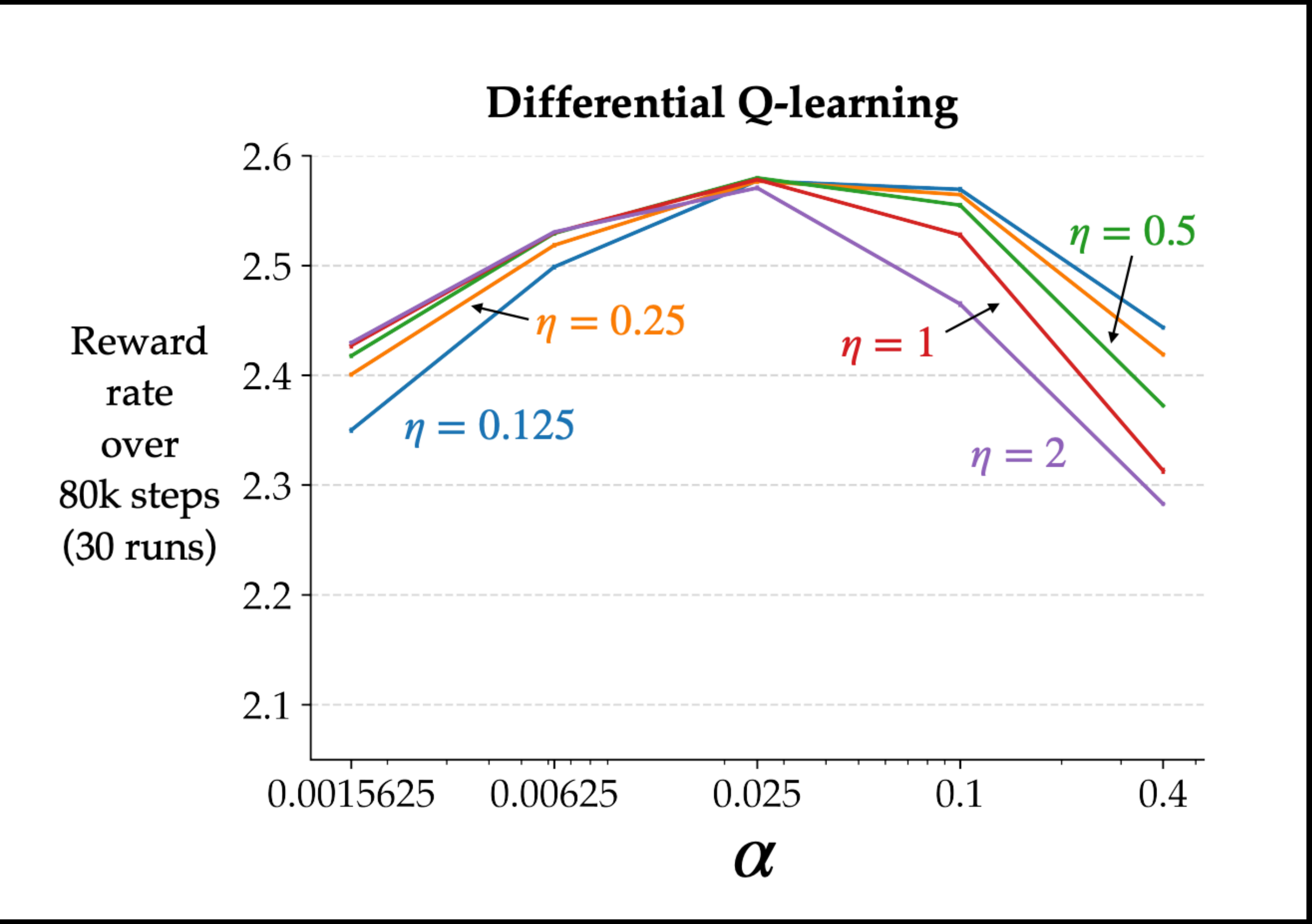
- $\alpha \in \{0.0015625, 0.00625, 0.025, 0.1, 0.4\}$
- $\eta \in \{0.125, 0.25, 0.5, 1, 2\}$
- 80,000 steps
- 30 runs
- $\epsilon = 0.1$



PARAMETER STUDY AND INFERENCES



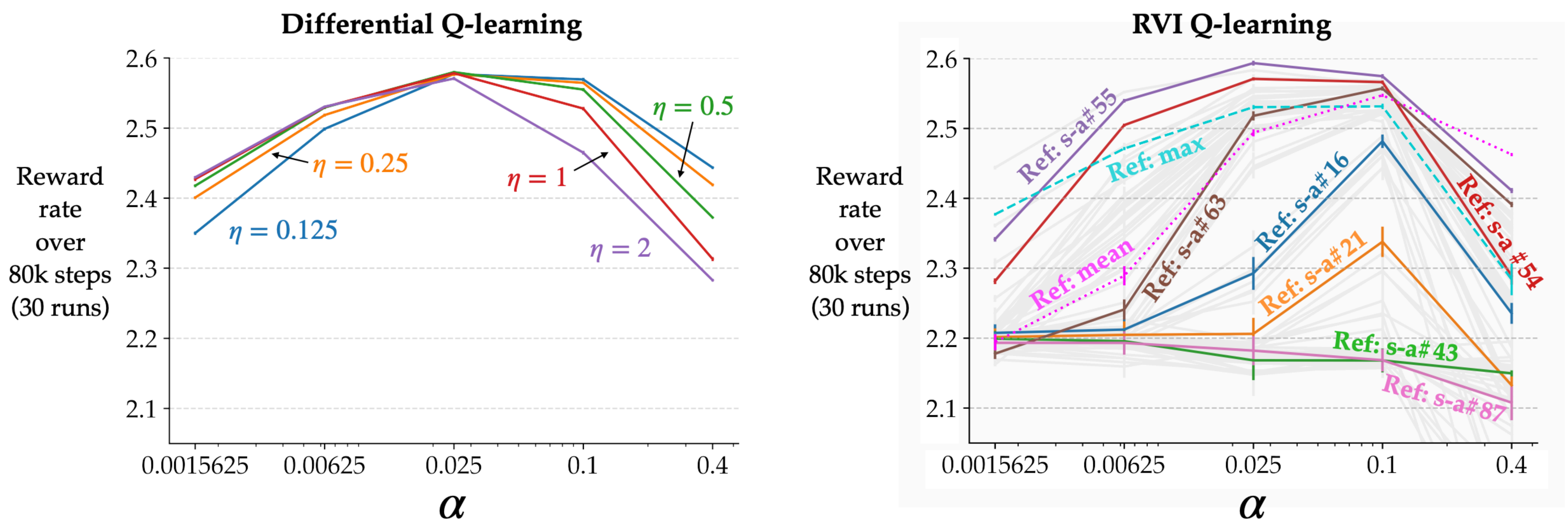
PARAMETER STUDY AND INFERENCES



- Differential Q-learning’s performance varies only slightly over a wide range of parameter values.

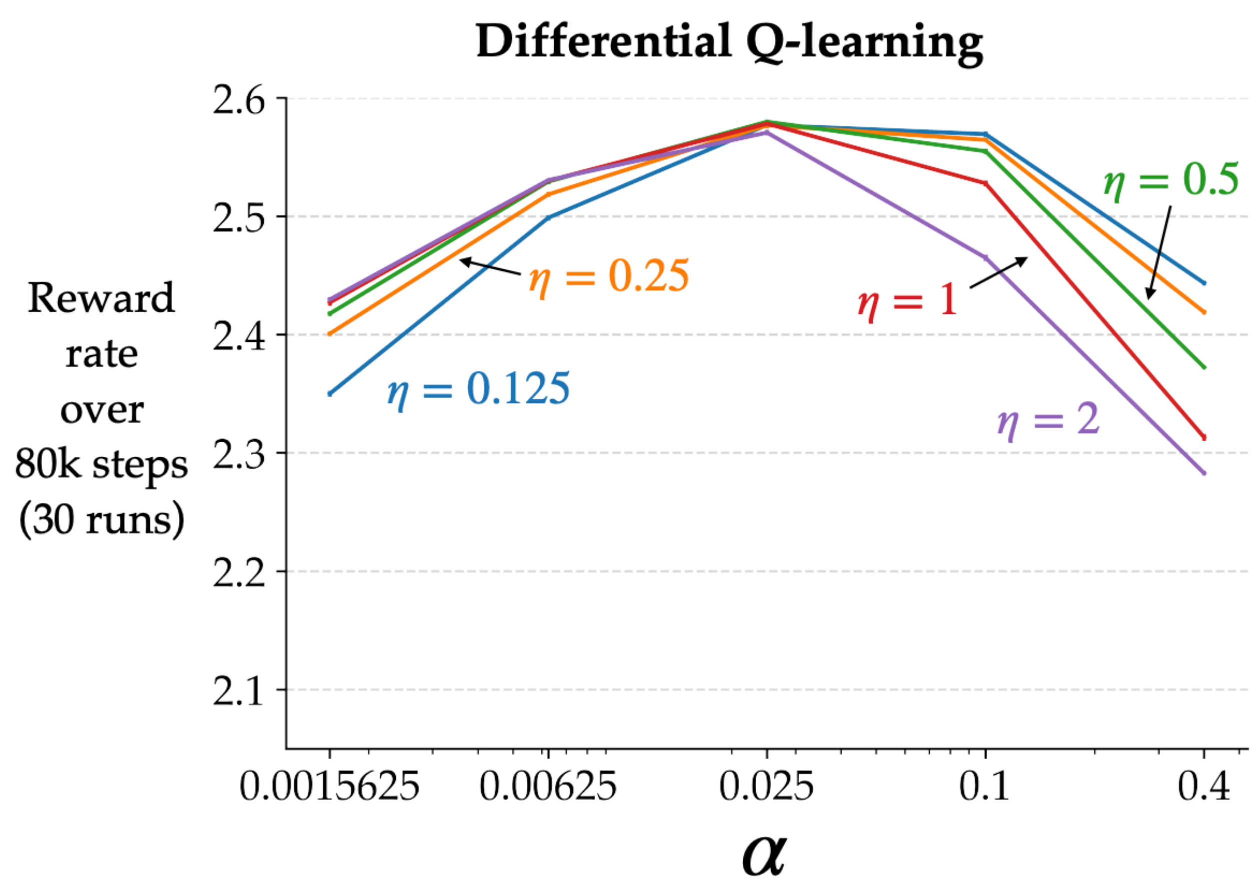
CONTROL EXPERIMENTS

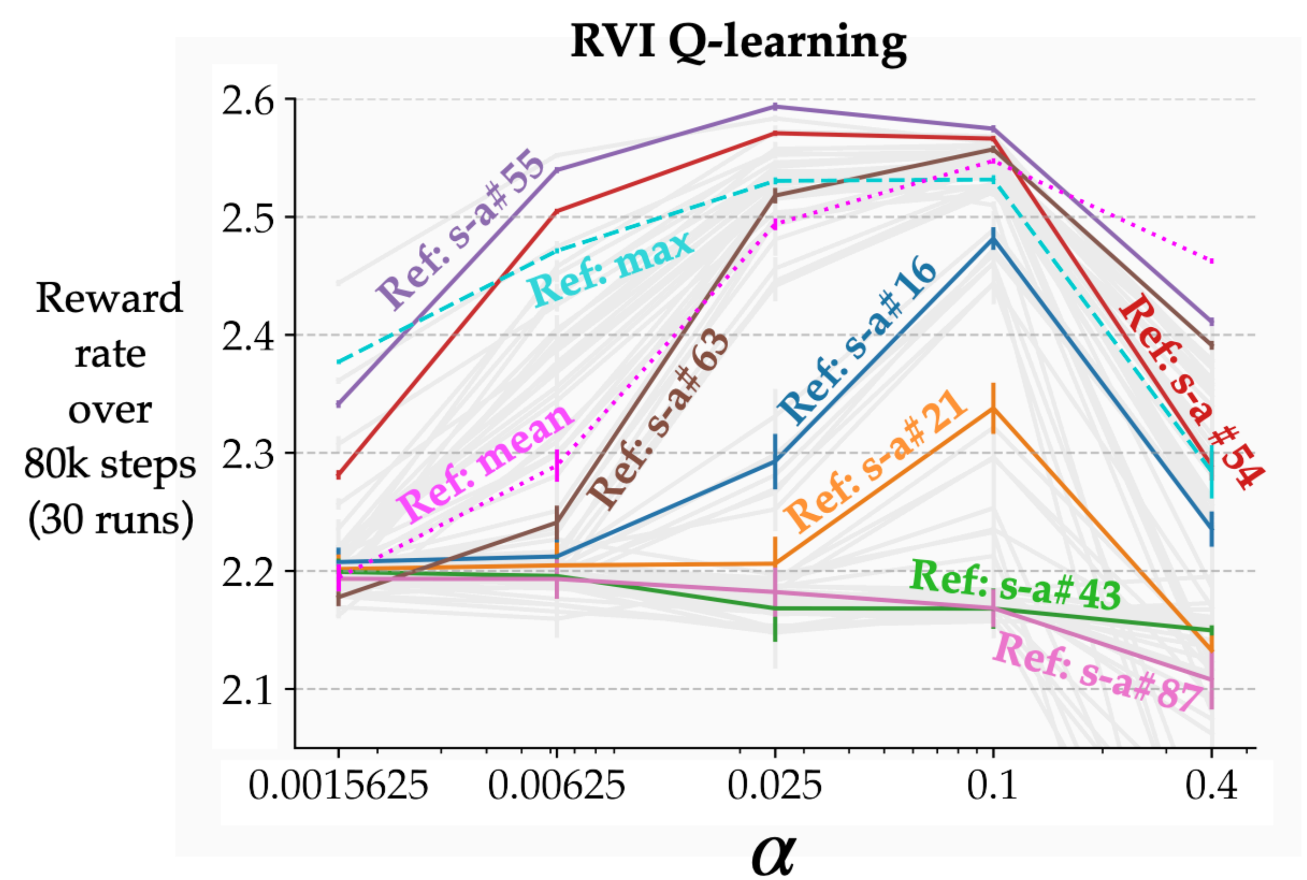
PARAMETER STUDY AND INFERENCES



- Differential Q-learning's performance varies only slightly over a wide range of parameter values.

PARAMETER STUDY AND INFERENCES





- Differential Q-learning's performance varies only slightly over a wide range of parameter values.
- RVI Q-learning's performance depends significantly on the choice of the reference state.

CONTRIBUTIONS

A family of average-reward learning and planning algorithms, including:

1. The first general proven-convergent off-policy model-free *control* algorithm without reference states

2. The first proven-convergent off-policy model-free *prediction* algorithm

3. A general technique to estimate the actual value function rather than the value function plus an offset

ALGORITHM

PREDICTION

PREDICTION

$$\delta_t \doteq R_{t+1} - \bar{R}_t + V_t(S_{t+1}) - V_t(S_t)$$

ALGORITHM

PREDICTION

$$\delta_t \doteq R_{t+1} - \bar{R}_t + V_t(S_{t+1}) - V_t(S_t)$$

$$V_{t+1}(S_t) \doteq V_t(S_t) + \alpha_t \rho_t \delta_t$$

$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta \alpha_t \rho_t \delta_t$$

PREDICTION

Differential TD-learning

$$\delta_t \doteq R_{t+1} - \bar{R}_t + V_t(S_{t+1}) - V_t(S_t)$$

$$V_{t+1}(S_t) \doteq V_t(S_t) + \alpha_t \rho_t \delta_t$$

$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta \alpha_t \rho_t \delta_t$$

ALGORITHM

PREDICTION

Differential TD-learning

$$\delta_t \doteq R_{t+1} - \bar{R}_t + V_t(S_{t+1}) - V_t(S_t)$$

$$V_{t+1}(S_t) \doteq V_t(S_t) + \alpha_t \rho_t \delta_t$$

$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta \alpha_t \rho_t \delta_t$$

Theorem 2 (informal)

If 1) the MDP is recurrent,
2) the stepsizes are decreased appropriately,
3) all the states are updated infinite number of times,
4) the maximum ratio of the update frequencies is finite,
5) b covers all the actions that π may choose in all states,

then the Differential TD-learning algorithm converges a.s.:
$\bar{R}_t$ to $r(\pi)$, V_t to a solution of the Bellman evaluation equation.

PREDICTION

Algorithm 3: Differential TD-learning (one-step off-policy prediction)

Input: The policy π to be evaluated, and b to be used
Algorithm parameters: step sizes α, η

```
Initialize V(s) ∀s, R̄ arbitrarily (e.g., to zero)
while still time to train do
    A ← action given by b for S
    Take action A, observe R, S'
    δ = R − R̄ + V(S') − V(S)
    ρ = π(A|S) / b(A|S)
    V(S) = V(S) + αρδ
    R̄ = R̄ + ηαρδ
    S = S'
end
return V
```

PREDICTION

Algorithm 3: Differential TD-learning (one-step off-policy prediction)

Input: The policy π to be evaluated, and b to be used

Algorithm parameters: step sizes α, η

Initialize $V(s)\ \forall s,\ \bar{R}$ arbitrarily (e.g., to zero)

while *still time to train* **do**

- $A \leftarrow$ action given by b for S
- Take action A, observe R, S'
- $\delta = R - \bar{R} + V(S') - V(S)$
- $\rho = \frac{\pi(A|S)}{b(A|S)}$
- $V(S) = V(S) + \alpha\rho\delta$
- $\bar{R} = \bar{R} + \eta\alpha\rho\delta$
- $S = S'$

end

return V

Average Cost TD-learning

$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta\alpha_t(R_{t+1} - \bar{R}_t)$$

PREDICTION

Algorithm 3: Differential TD-learning (one-step off-policy prediction)

Input: The policy π to be evaluated, and b to be used

Algorithm parameters: step sizes α, η

```
Initialize V(s) ∀s, R̄ arbitrarily (e.g., to zero)
while still time to train do
    A ← action given by b for S
    Take action A, observe R, S'
    δ = R − R̄ + V(S') − V(S)
    ρ = π(A|S) / b(A|S)
    V(S) = V(S) + αρδ
    R̄ = R̄ + ηαρδ
    S = S'
end
return V
```

Average Cost TD-learning

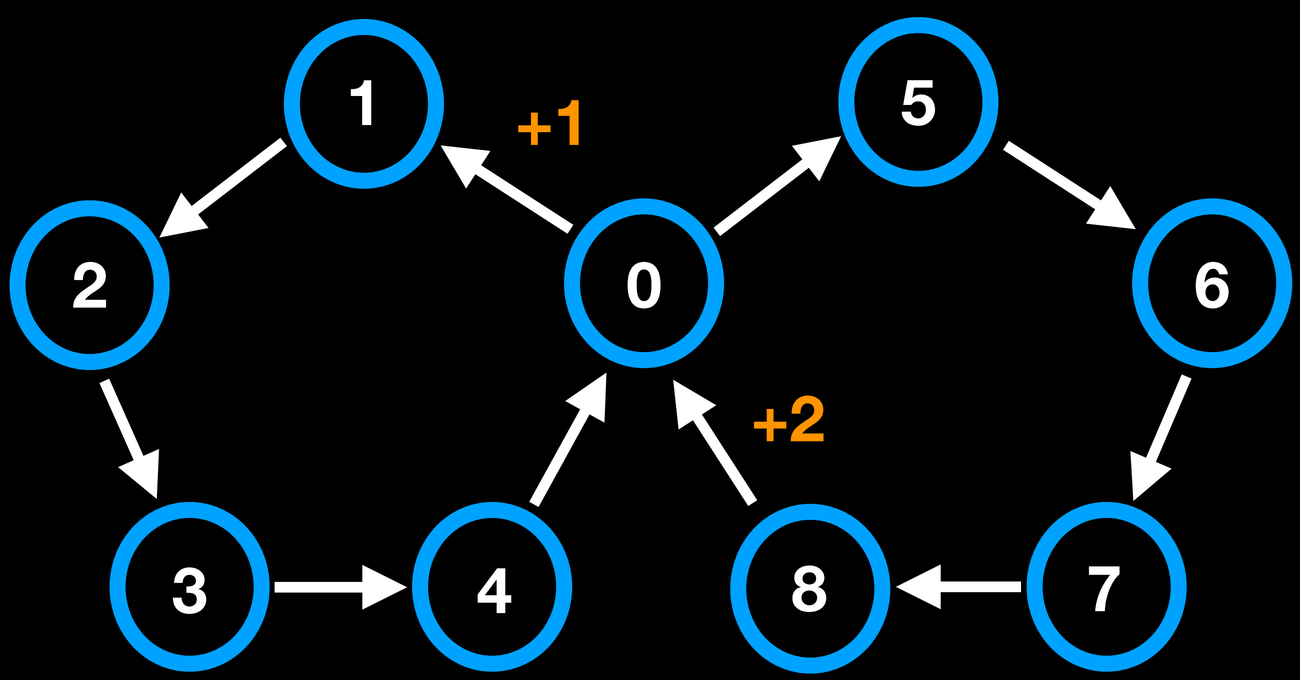
$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta\alpha_t(R_{t+1} - \bar{R}_t)$$

(restricted to on-policy)

DOMAIN

PREDICTION

- Two Loop Task



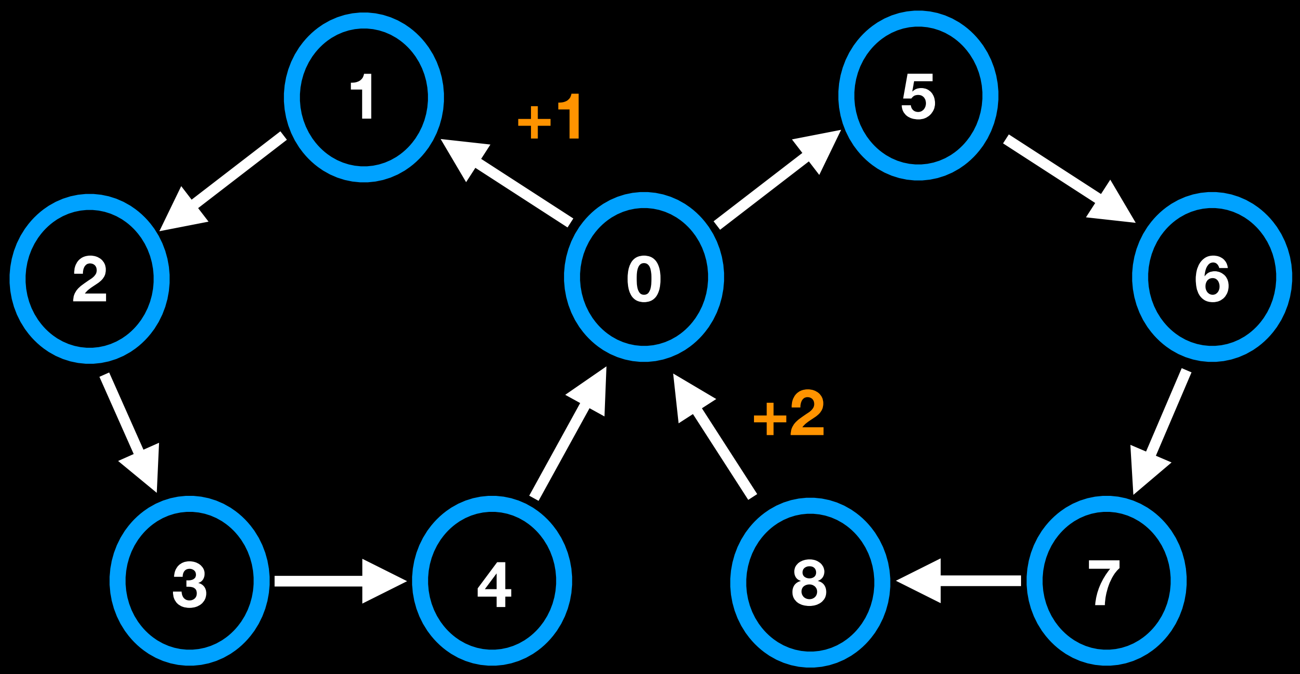
EXPERIMENT

- $\pi_0 = [0.5,\ 0.5],\ b_0 = [0.9,\ 0.1]$
- $\alpha \in \{0.025,\ 0.05,\ 0.1,\ 0.2,\ 0.4\}$
- $\eta \in \{0.125,\ 0.25,\ 0.5,\ 1,\ 2\}$
- $\epsilon = 0.1$
- 10,000 steps
- 30 runs
- Target policy: 0.5 left, 0.5 right
- Behavior policy: 0.9 left, 0.1 right

DOMAIN

PREDICTION

- Two Loop Task



EXPERIMENT

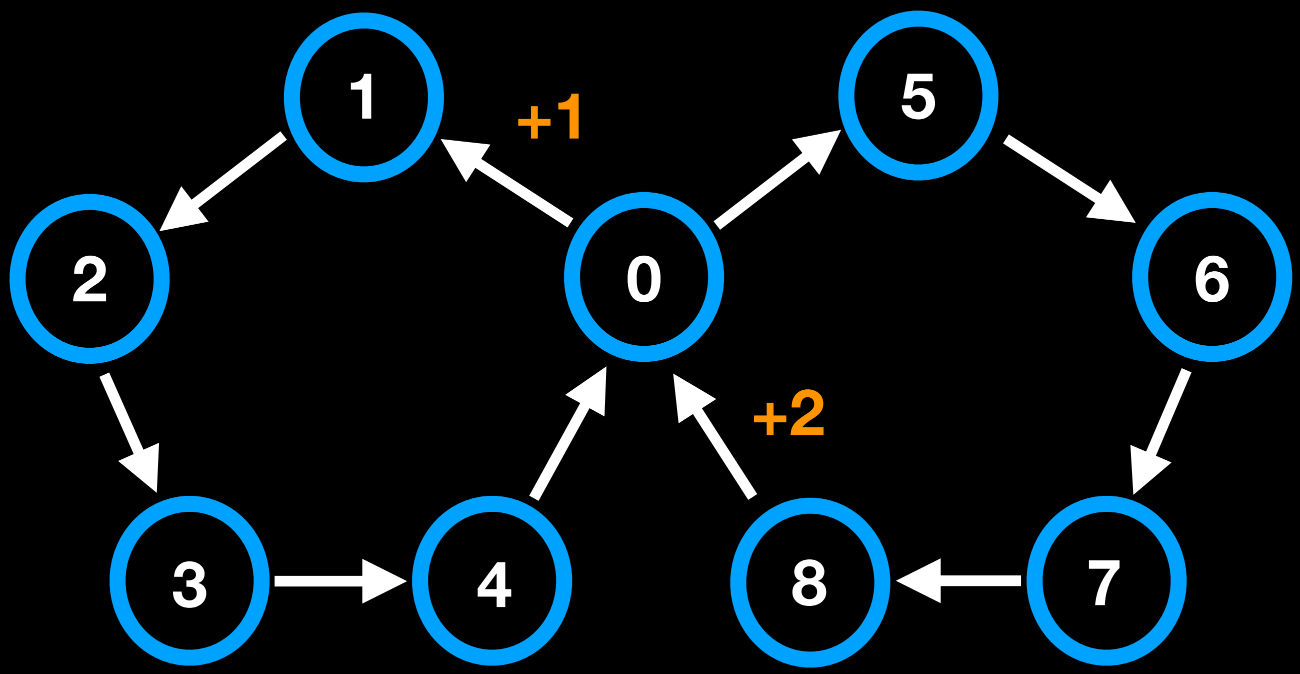
- $\pi_0 = [0.5,\ 0.5],\ b_0 = [0.9,\ 0.1]$
- $\alpha \in \{0.025,\ 0.05,\ 0.1,\ 0.2,\ 0.4\}$
- $\eta \in \{0.125,\ 0.25,\ 0.5,\ 1,\ 2\}$
- $\epsilon = 0.1$
- 10,000 steps
- 30 runs
- Target policy: 0.5 left, 0.5 right
- Behavior policy: 0.9 left, 0.1 right
- Evaluation metric:
  - RMSVE

$$\|v - \ v_\pi \qquad \|_{d_\pi}$$

DOMAIN

PREDICTION

- Two Loop Task



EXPERIMENT

- $\pi_0 = [0.5, 0.5]$, $b_0 = [0.9, 0.1]$
- $\alpha \in \{0.025, 0.05, 0.1, 0.2, 0.4\}$
- $\eta \in \{0.125, 0.25, 0.5, 1, 2\}$
- $\epsilon = 0.1$
- 10,000 steps
- 30 runs
- Target policy: 0.5 left, 0.5 right
- Behavior policy: 0.9 left, 0.1 right
- Evaluation metric:
  - RMSVE

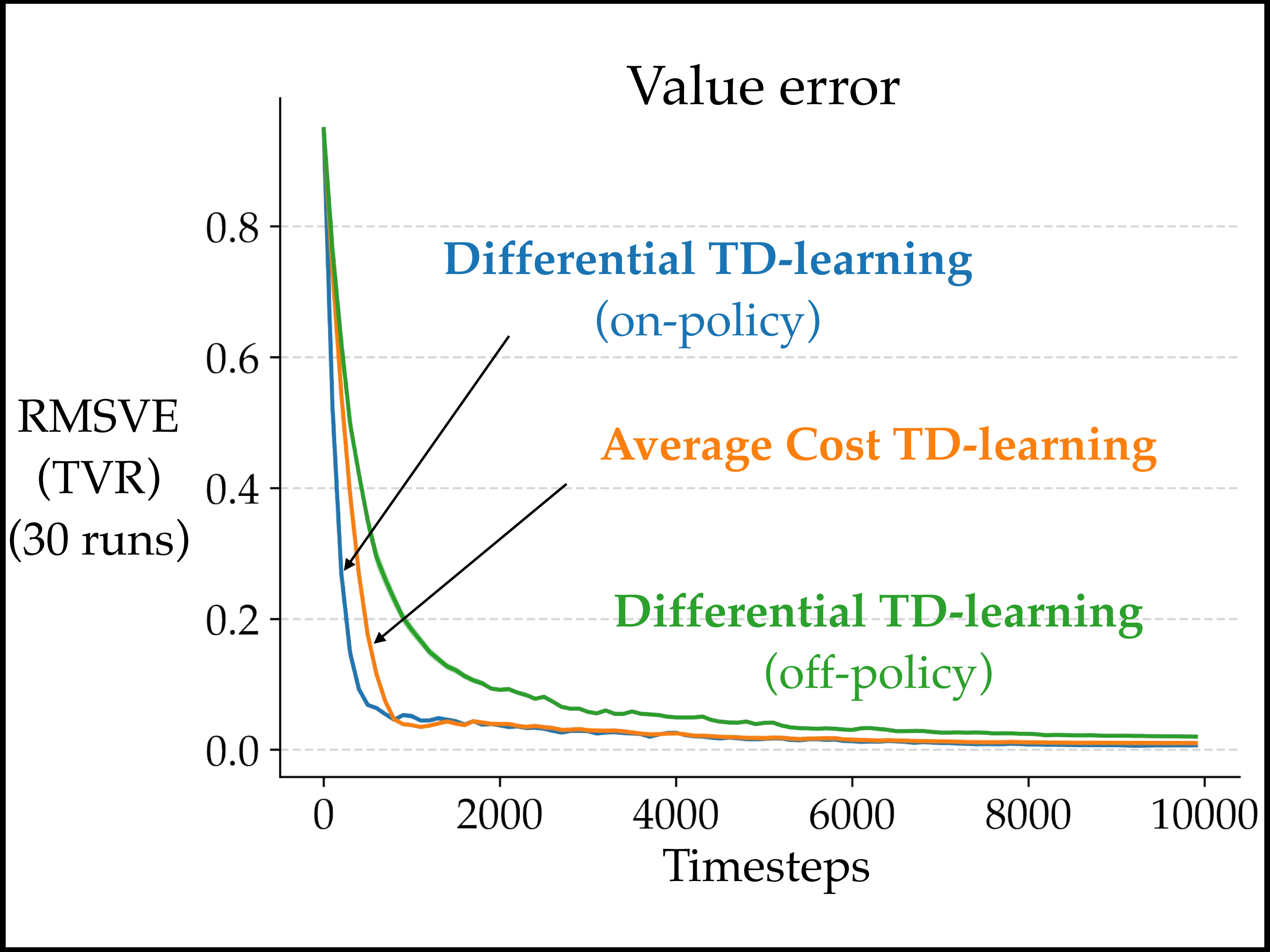
$$\inf_c \|v - (v_\pi + ce)\|_{d_\pi}$$

(Tsitsiklis and Van Roy, 1999)

RESULTS

PREDICTION

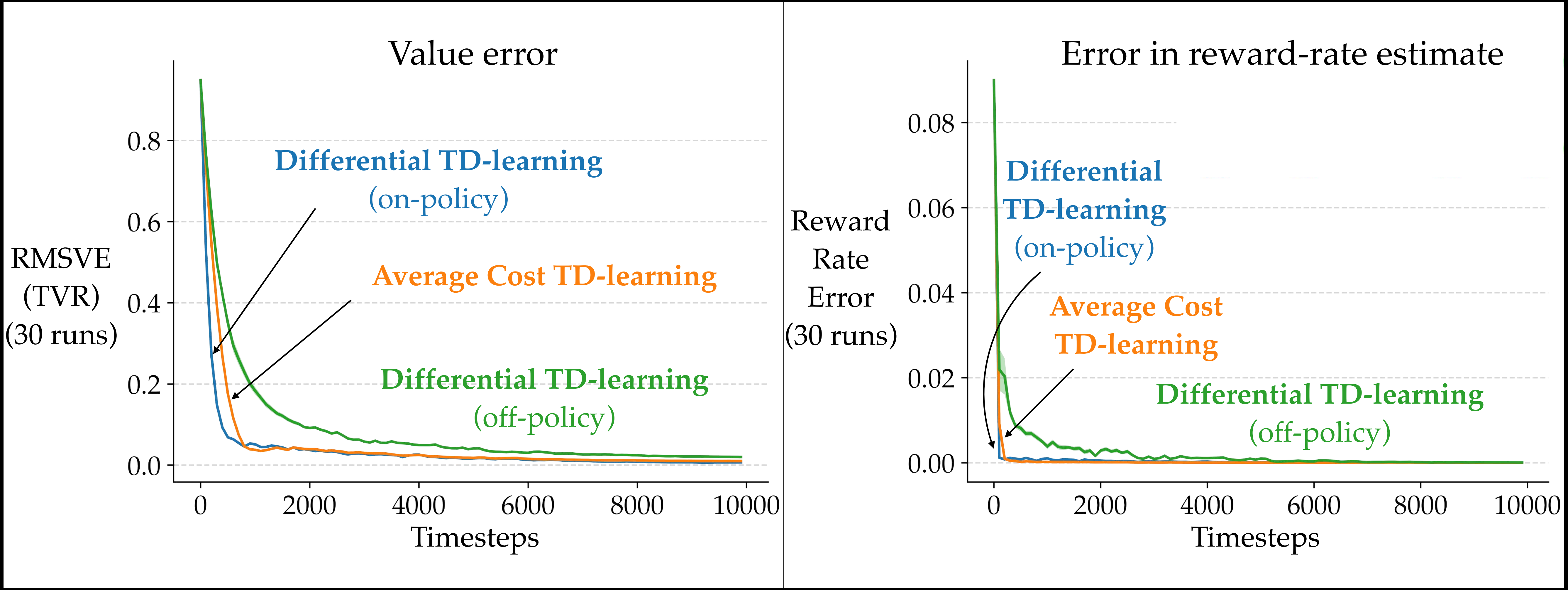
Learning curves



RESULTS

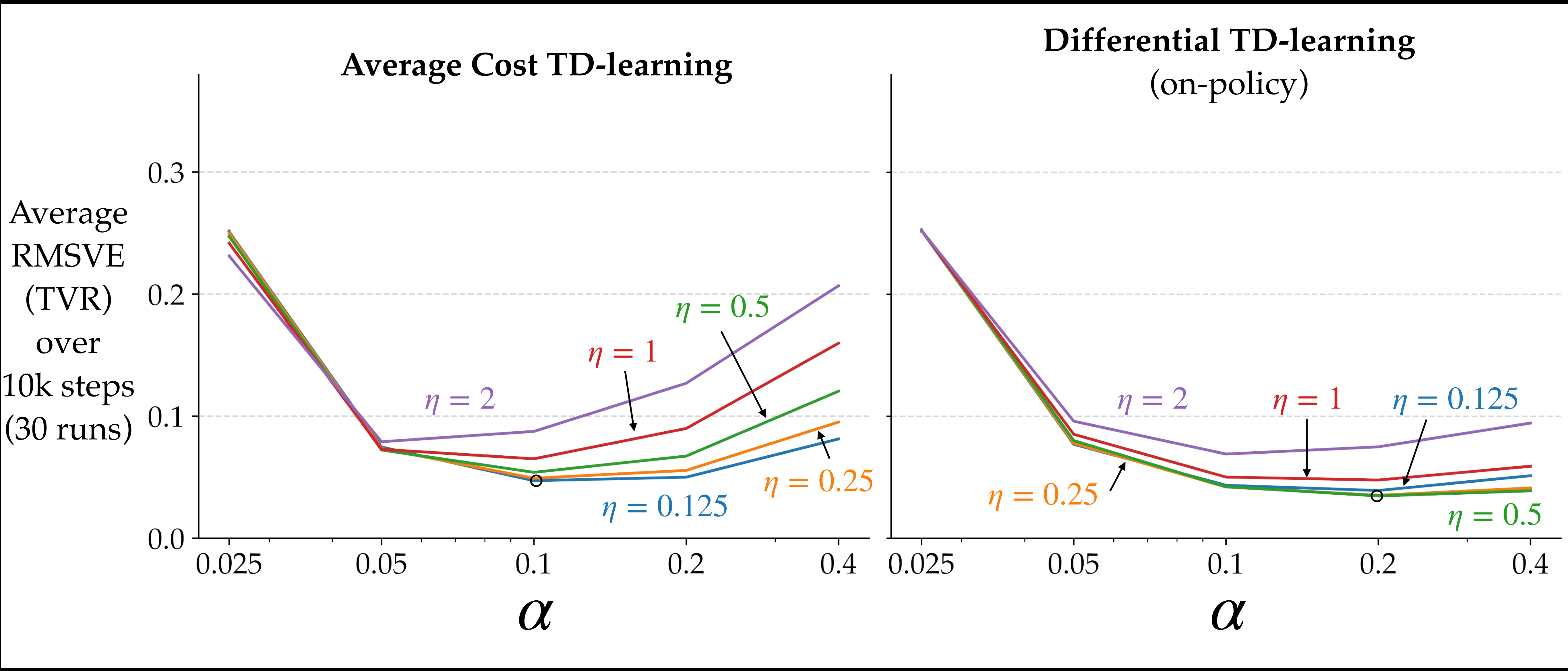
PREDICTION

Learning curves



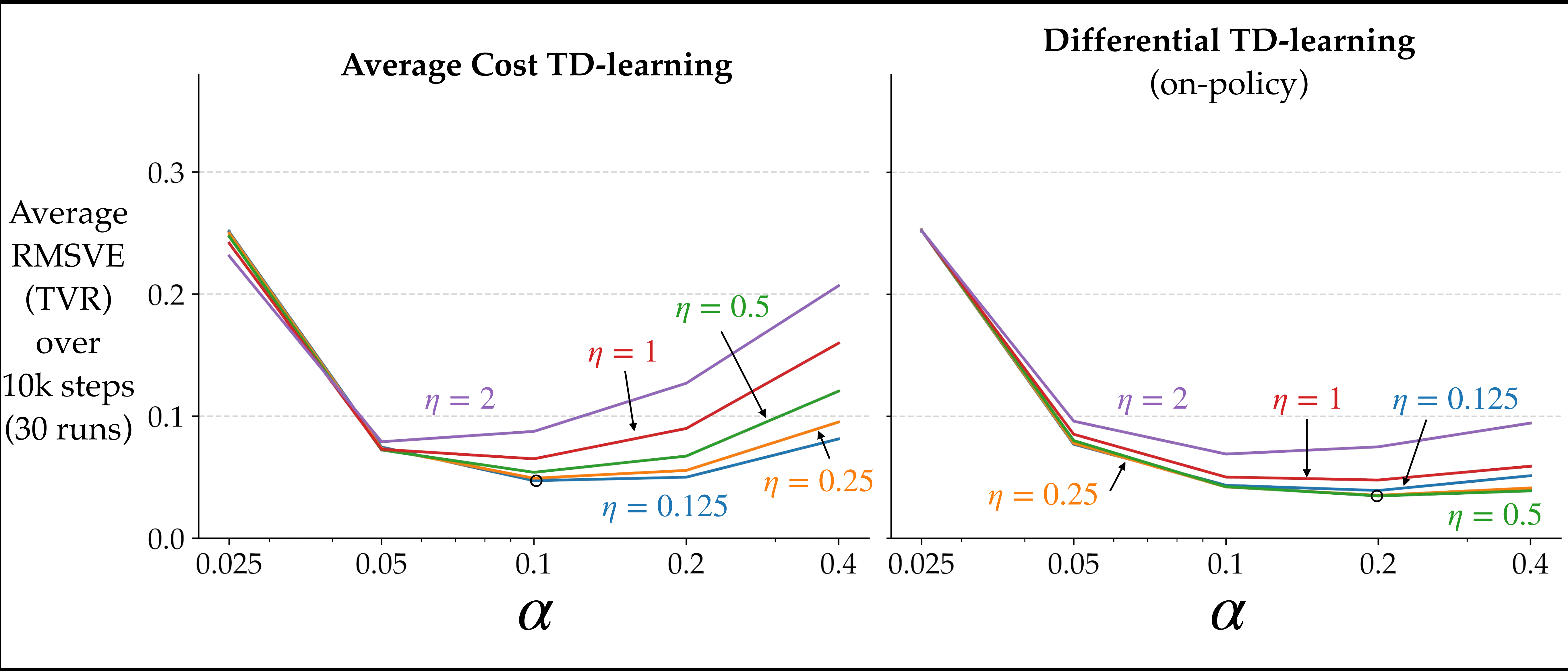
PREDICTION

Sensitivity analysis (value error)



PREDICTION

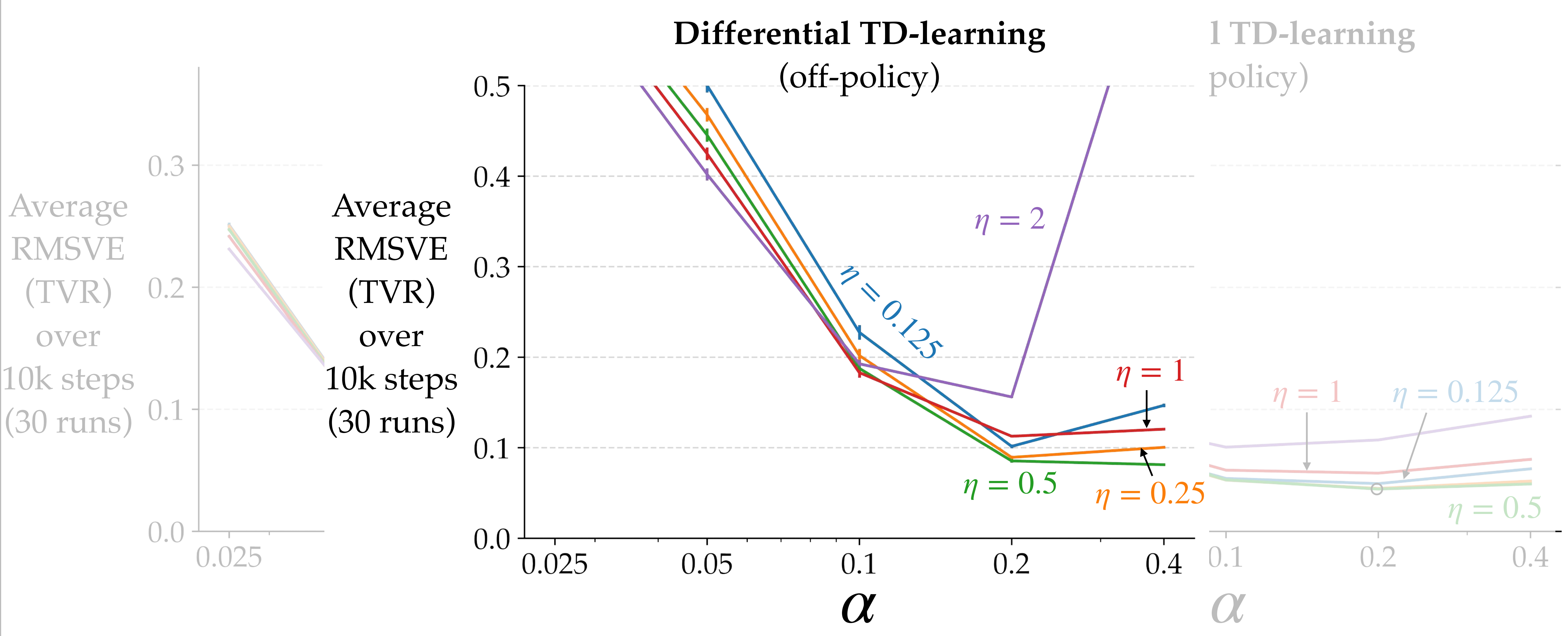
Sensitivity analysis (value error)



- Differential TD-learning converges faster for a wide range of parameters.

PREDICTION

Sensitivity analysis (value error)

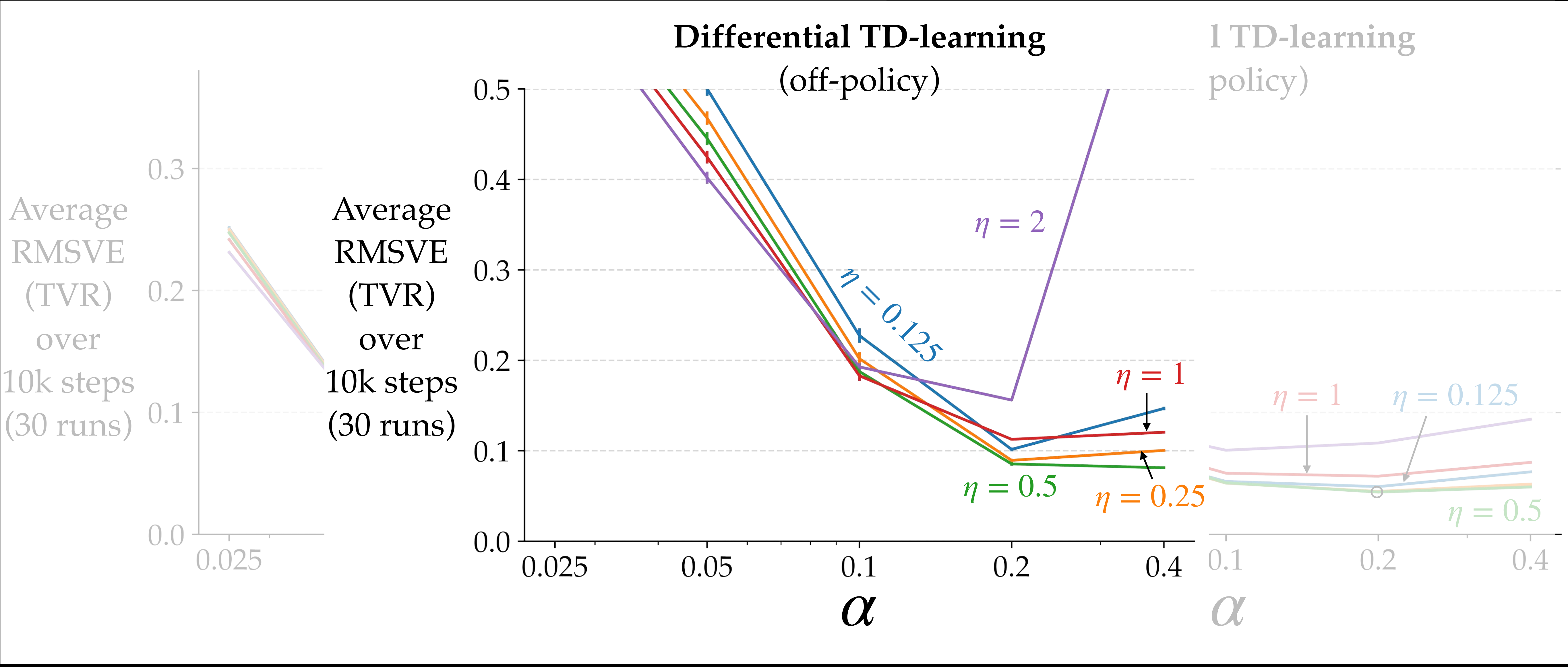


- Differential TD-learning converges faster for a wide range of parameters.

RESULTS

PREDICTION

Sensitivity analysis (value error)



- Differential TD-learning converges faster for a wide range of parameters.
- Differential TD-learning works in the off-policy setting as well.

CONTRIBUTIONS

A family of average-reward learning and planning algorithms, including:

1. The first general proven-convergent off-policy model-free *control* algorithm without reference states

2. The first proven-convergent off-policy model-free *prediction* algorithm

3. A general technique to estimate the actual value function rather than the value function plus an offset

CENTERING

CENTERING

Recall: $$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[R_{t+1} - \bar{r} + v(s')\right] \quad \forall s$$

CENTERING

Recall: $$v(s) = \sum_a \pi(a \mid s) \sum_{s', r} p(s', r \mid s, a) \left[R_{t+1} - \bar{r} + v(s') \right] \quad \forall s$$

Solutions: $$v = v_\pi + ce$$

CENTERING

Recall: $$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[R_{t+1} - \bar{r} + v(s')\right] \quad \forall s$$

Solutions: $$v = v_\pi + ce$$

Lemma

$$d_\pi^T v_\pi = 0,$$

i.e., the average of the differential value function is zero.

CENTERING

Recall: $v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\big[R_{t+1} - \bar{r} + v(s')\big] \quad \forall s$

Solutions: $v = v_\pi + ce$

Lemma

$$d_\pi^T v_\pi = 0,$$

i.e., the average of the differential value function is zero.

$\Longrightarrow$ there is only one *centered* differential value function

CENTERING

Recall: $$v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[R_{t+1} - \bar{r} + v(s')\right] \quad \forall s$$

Solutions: $$v = v_\pi + ce$$

Lemma

$$d_\pi^T v_\pi = 0,$$

i.e., the average of the differential value function is zero.

$\implies$ there is only one *centered* differential value function

$$v = v_\pi + ce$$

CENTERING

Recall: $v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[R_{t+1} - \bar{r} + v(s')\right] \quad \forall s$

Solutions: $v = v_\pi + ce$

Lemma

$$d_\pi^T v_\pi = 0,$$

i.e., the average of the differential value function is zero.

$\implies$ there is only one *centered* differential value function

$$v = v_\pi + ce$$
$$\implies c = d_\pi^T v$$

CENTERING

Recall: $v(s) = \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a)\left[R_{t+1} - \bar{r} + v(s')\right] \quad \forall s$

Solutions: $v = v_\pi + ce$

Lemma

$$d_\pi^T v_\pi = 0,$$

i.e., the average of the differential value function is zero.

$\implies$ there is only one *centered* differential value function

$$v = v_\pi + ce$$
$$\implies c = d_\pi^T v$$
$$r(\pi) = d_\pi^T r_\pi$$

CENTERING

$$\delta_t \doteq R_{t+1} - \bar{R}_t + V_t(S_{t+1}) - V_t(S_t)$$

$$V_{t+1}(S_t) \doteq V_t(S_t) + \alpha_t \rho_t \delta_t$$

$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta \alpha_t \rho_t \delta_t$$

System 1

CENTERING

$$\delta_t \doteq R_{t+1} - \bar{R}_t + V_t(S_{t+1}) - V_t(S_t)$$

$$V_{t+1}(S_t) \doteq V_t(S_t) + \alpha_t \rho_t \delta_t$$

$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta \alpha_t \rho_t \delta_t$$

System 1

$$\Delta_t \doteq V_t(S_t) - \bar{V}_t + F_t(S_{t+1}) - F_t(S_t)$$

$$F_{t+1}(S_t) \doteq F_t(S_t) + \beta_t \rho_t \Delta_t$$

$$\bar{V}_{t+1} \doteq \bar{V}_t + \kappa \beta_t \rho_t \Delta_t$$

System 2

CENTERING

$$\delta_t \doteq R_{t+1} - \bar{R}_t + V_t(S_{t+1}) - V_t(S_t)$$
$$V_{t+1}(S_t) \doteq V_t(S_t) + \alpha_t \rho_t \delta_t$$
$$\bar{R}_{t+1} \doteq \bar{R}_t + \eta \alpha_t \rho_t \delta_t$$

System 1

$$\Delta_t \doteq V_t(S_t) - \bar{V}_t + F_t(S_{t+1}) - F_t(S_t)$$
$$F_{t+1}(S_t) \doteq F_t(S_t) + \beta_t \rho_t \Delta_t$$
$$\bar{V}_{t+1} \doteq \bar{V}_t + \kappa \beta_t \rho_t \Delta_t$$

System 2

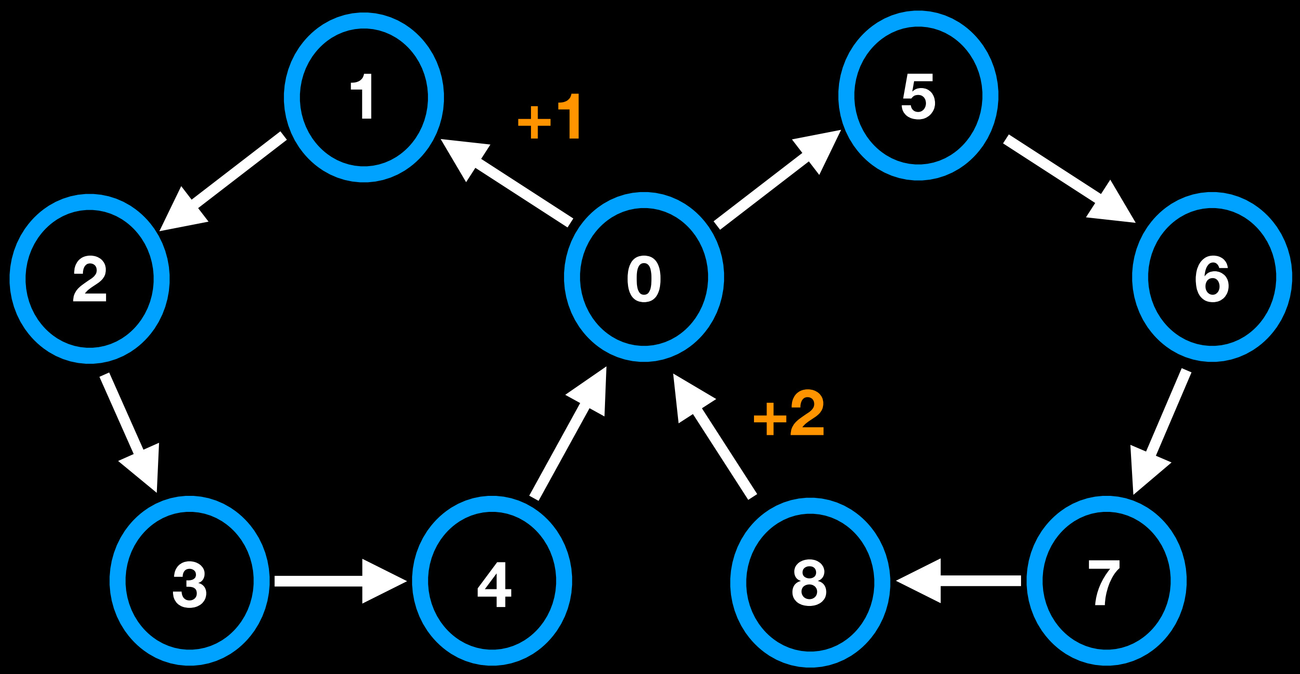
Theorem 3 (informal)

If the previous assumptions hold, then the Centered Differential TD-learning algorithm converges a.s.: $\bar{R}_t$ to $r(\pi)$, $V_t - \bar{V}_t e$ to the centered differential value function

DOMAIN

CENTERING

- Two Loop Task



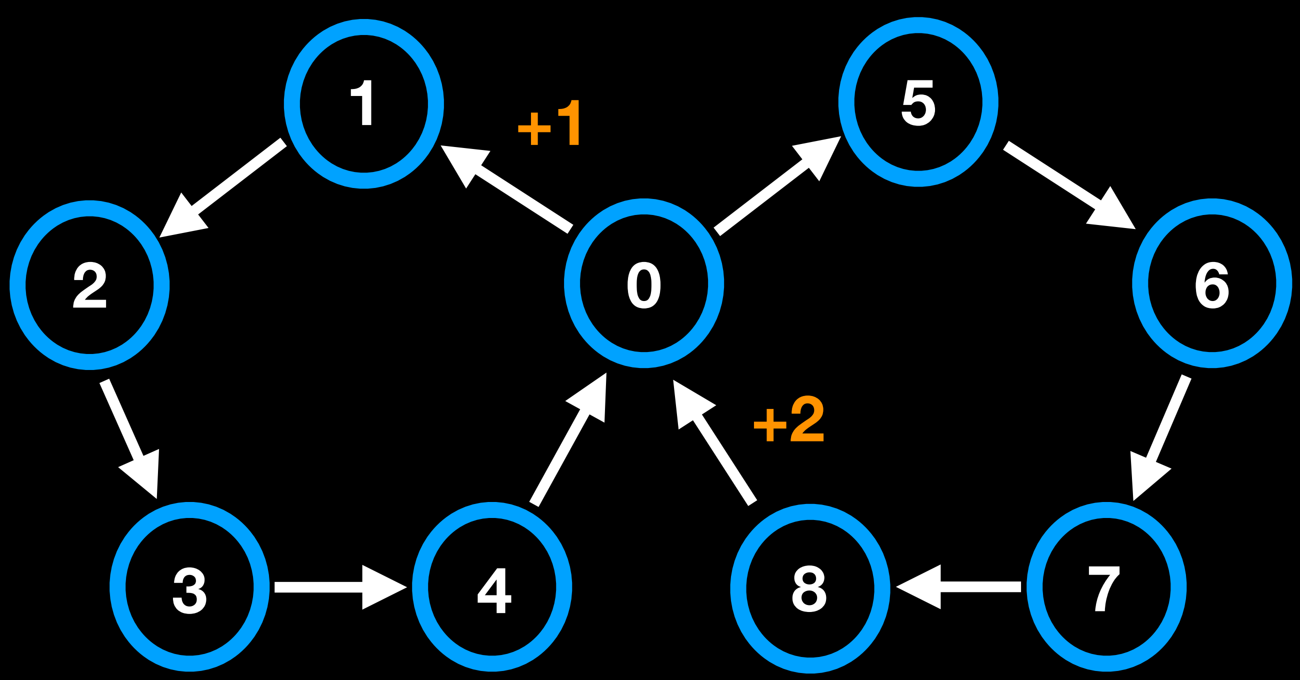
EXPERIMENT

- $\beta \in \{0.025,\ 0.05,\ 0.1,\ 0.2,\ 0.4\}$
- $\kappa \in \{0.125,\ 0.25,\ 0.5,\ 1,\ 2\}$
- $\epsilon = 0.1$
- 10,000 steps
- 30 runs

DOMAIN

CENTERING

- Two Loop Task



EXPERIMENT

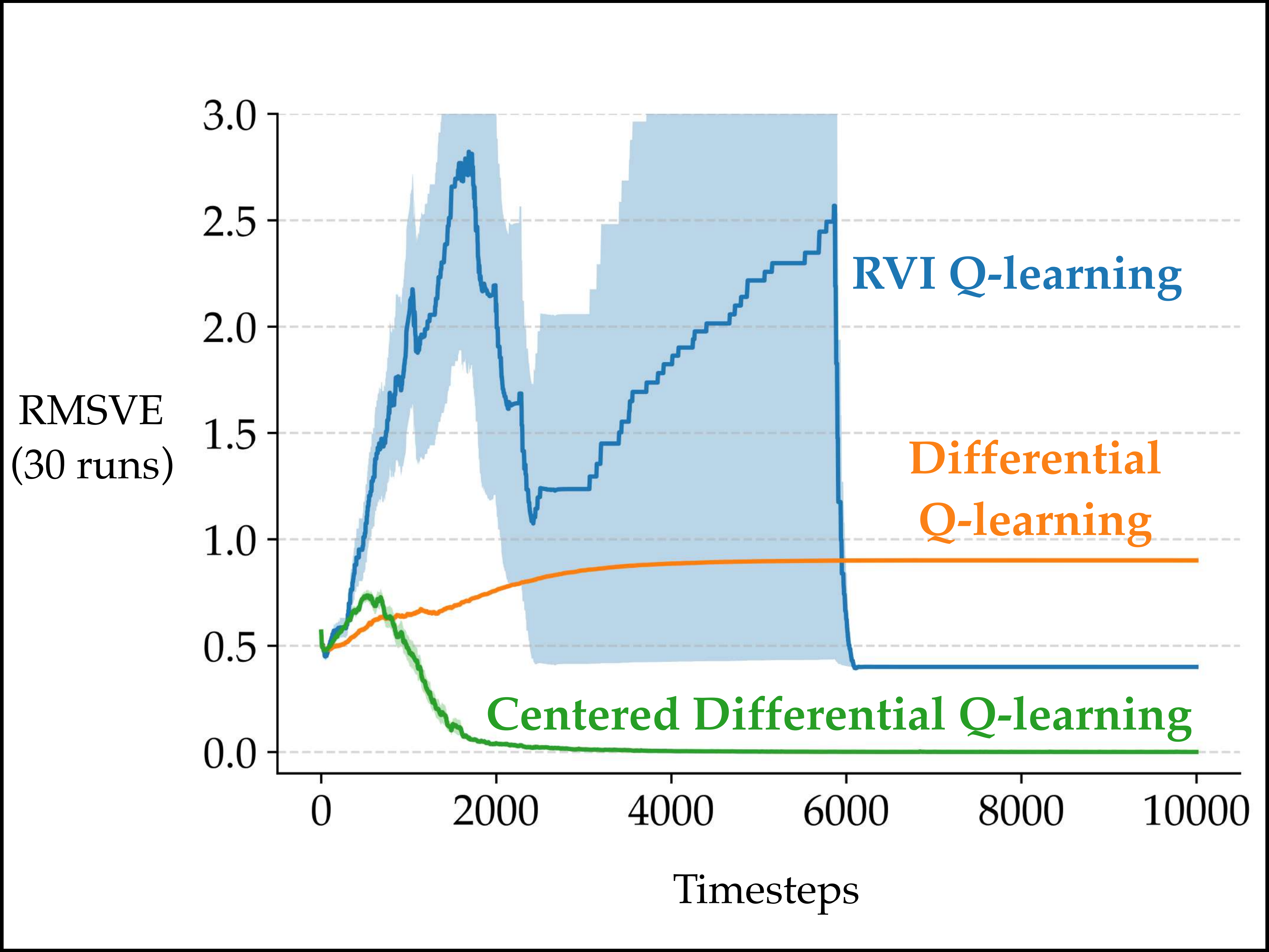
- $\beta \in \{0.025, 0.05, 0.1, 0.2, 0.4\}$
- $\kappa \in \{0.125, 0.25, 0.5, 1, 2\}$
- $\epsilon = 0.1$
- 10,000 steps
- 30 runs
- Evaluation metric:
  - RMSVE

$$\|v - v_\pi\|_{d_\pi}$$

(the usual one)

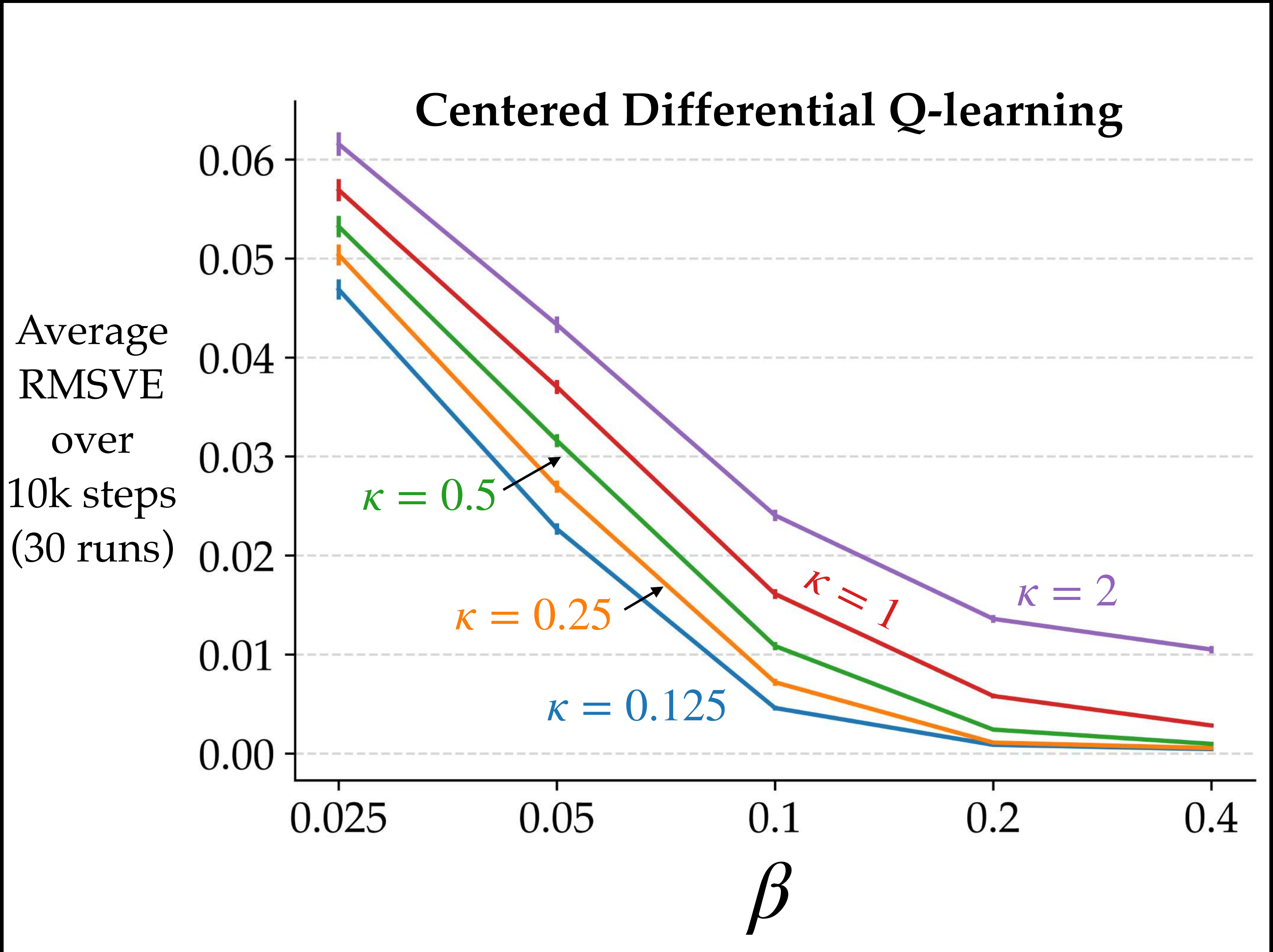
RESULTS

CENTERING



Learning curves

CENTERING



Sensitivity analysis

CONTRIBUTIONS

A family of average-reward learning and planning algorithms, including:

1. The first general proven-convergent off-policy model-free *control* algorithm without reference states

2. The first proven-convergent off-policy model-free *prediction* algorithm

3. A general technique to estimate the actual value function rather than the value function plus an offset

TAKEAWAY

TAKEAWAY

- The Differential family of methods for learning and planning in average-reward MDPs:
    - is guaranteed to converge,
    - results in good performance, and
    - is easy to use.

TAKEAWAY

- The Differential family of methods for learning and planning in average-reward MDPs:
    - is guaranteed to converge,
    - results in good performance, and
    - is easy to use.
- As a result, average-reward reinforcement learning is now more appealing and accessible.

FUTURE WORK

FUTURE WORK

- Theoretical extension of our tabular algorithms to function approximation

FUTURE WORK

- Theoretical extension of our tabular algorithms to function approximation
- Extension to SMDPs so they can be used with temporal abstractions like options

FUTURE WORK

- Theoretical extension of our tabular algorithms to function approximation
- Extension to SMDPs so they can be used with temporal abstractions like options
- Extension of our one-step algorithms to n-step and lambda returns, as well as eligibility traces

FUTURE WORK

- Theoretical extension of our tabular algorithms to function approximation
- Extension to SMDPs so they can be used with temporal abstractions like options
- Extension of our one-step algorithms to n-step and lambda returns, as well as eligibility traces
- Analysis of exploration techniques in the average-reward setting

THANK YOU

- Paper: *https://arxiv.org/abs/2006.16318*
- Code: *https://github.com/abhisheknaik96/average-reward-methods*