

# Reducing Sampling Error in Batch Temporal Difference Learning

**Brahma S. Pavse**<sup>1</sup>, Ishan Durugkar<sup>1</sup>, Josiah Hanna<sup>2</sup>, Peter Stone<sup>1 3</sup>

<sup>1</sup>The University of Texas at Austin

<sup>2</sup>The University of Edinburgh

<sup>3</sup>Sony AI

ICML July 2020

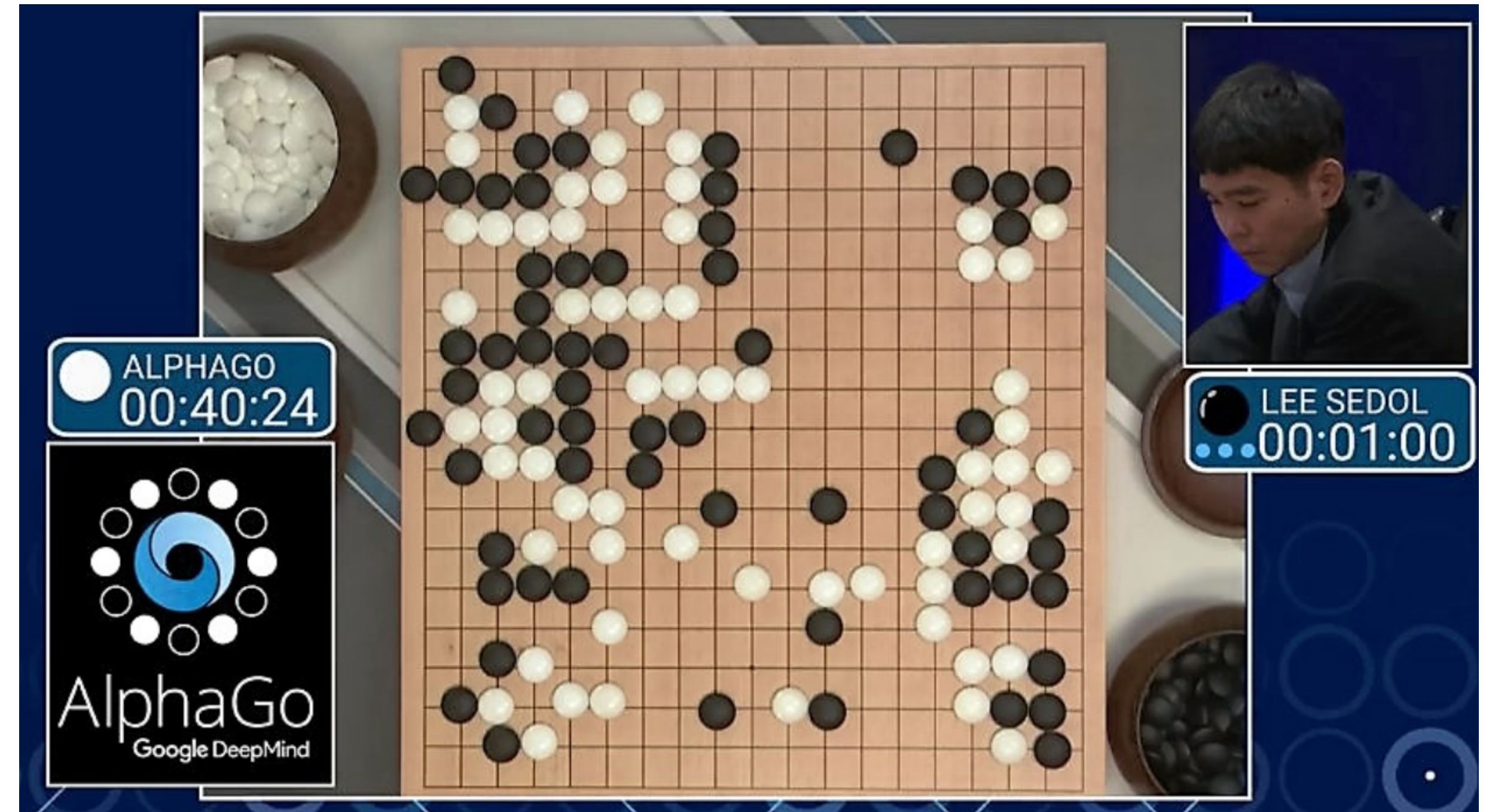
[brahmasp@cs.utexas.edu](mailto:brahmasp@cs.utexas.edu)

# Reinforcement Learning Successes

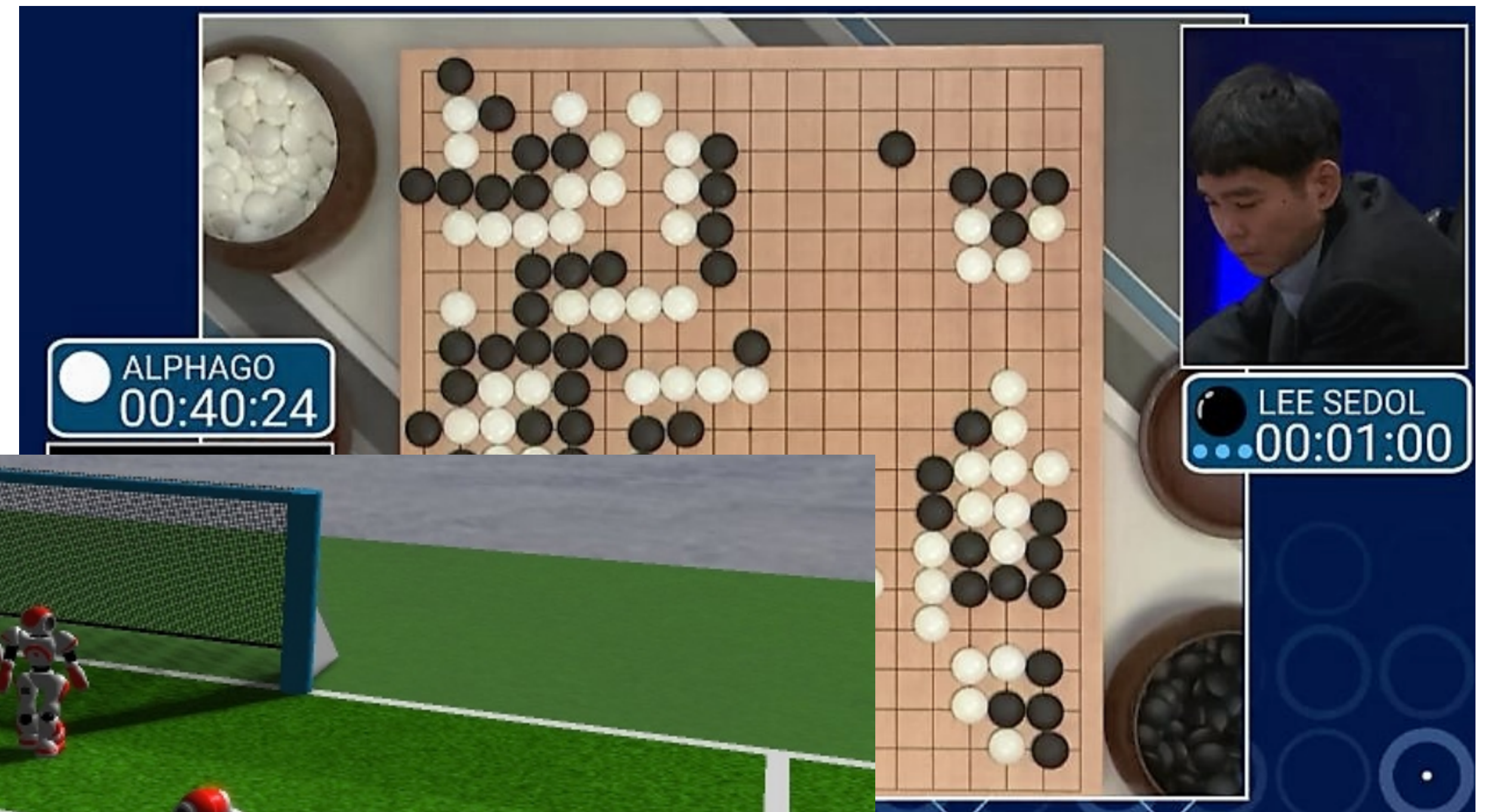
# Reinforcement Learning Successes



# Reinforcement Learning Successes



# Reinforcement Learning Successes



How can RL agents make the most from a finite amount of experience?

How can RL agents make the most from a finite amount of experience?

Learning an accurate estimation of the **value function** with **finite amount data**.

# Spotlight Overview



# Spotlight Overview

- With **finite** batch of data, **on-policy** single-step temporal difference learning converges to the value function for the **wrong** policy.

# Spotlight Overview

- With **finite** batch of data, **on-policy** single-step temporal difference learning converges to the value function for the **wrong** policy.
- Propose and prove that a **more efficient** estimator converges to the value function for the **true** policy.

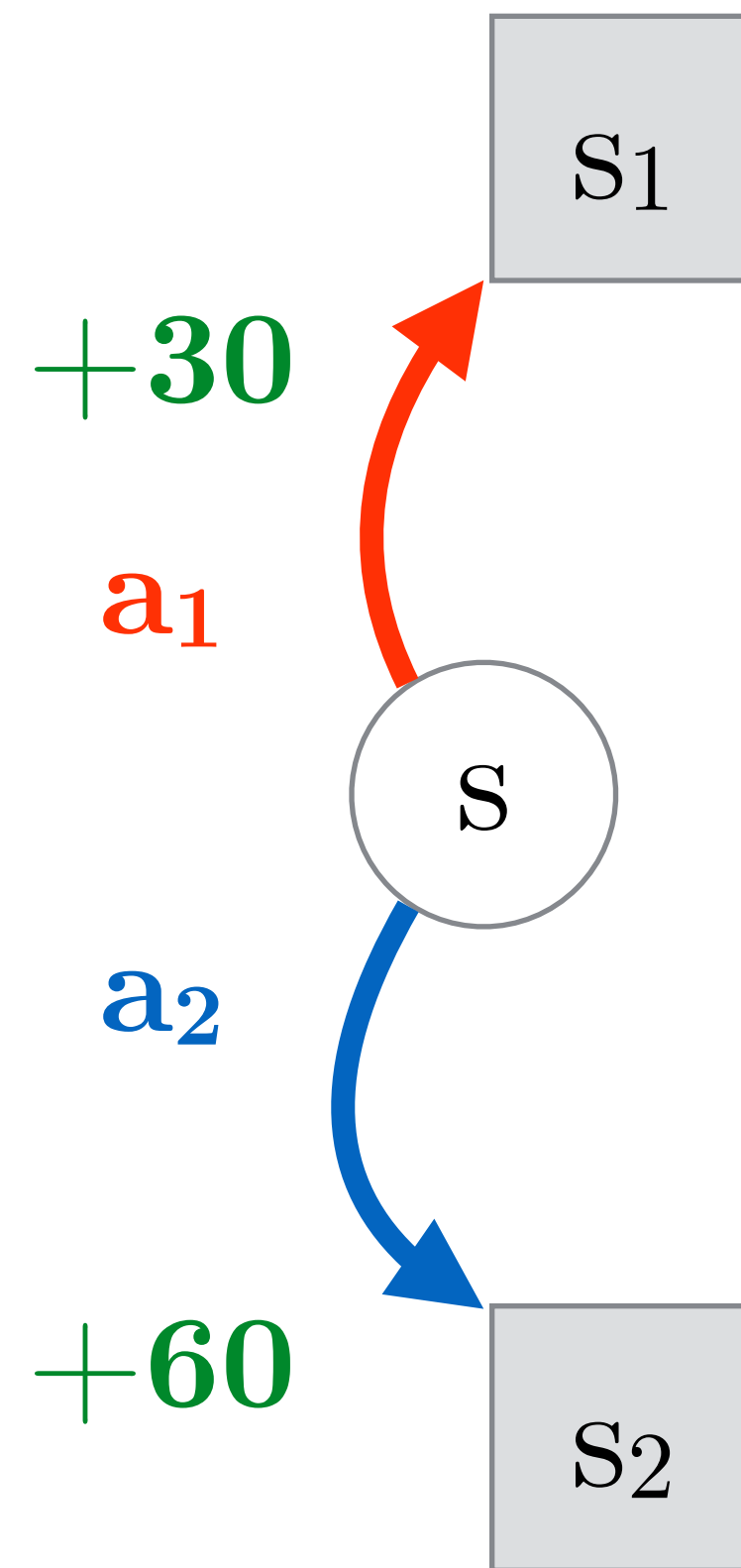
# Spotlight Overview: Flaw in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



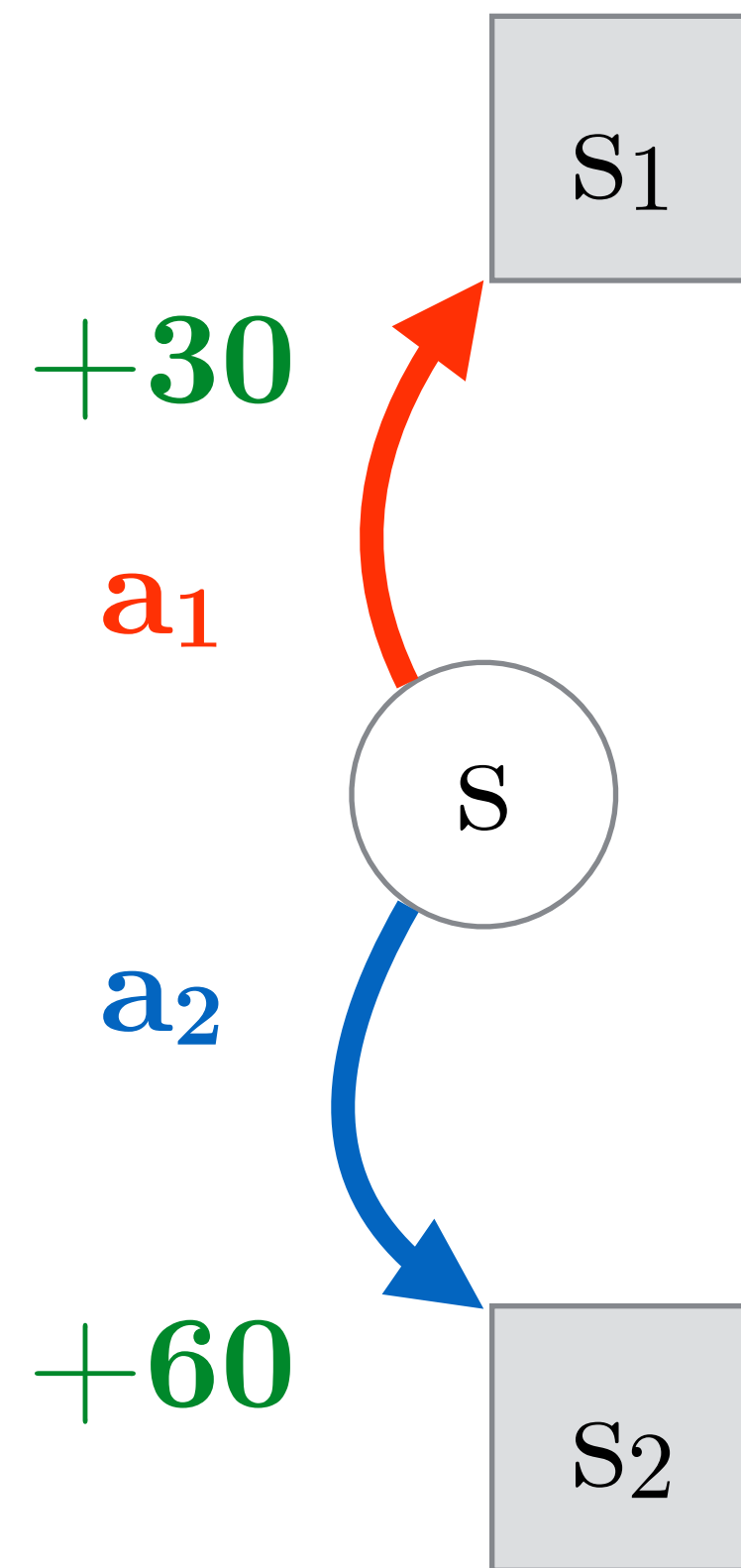
# Spotlight Overview: Flaw in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

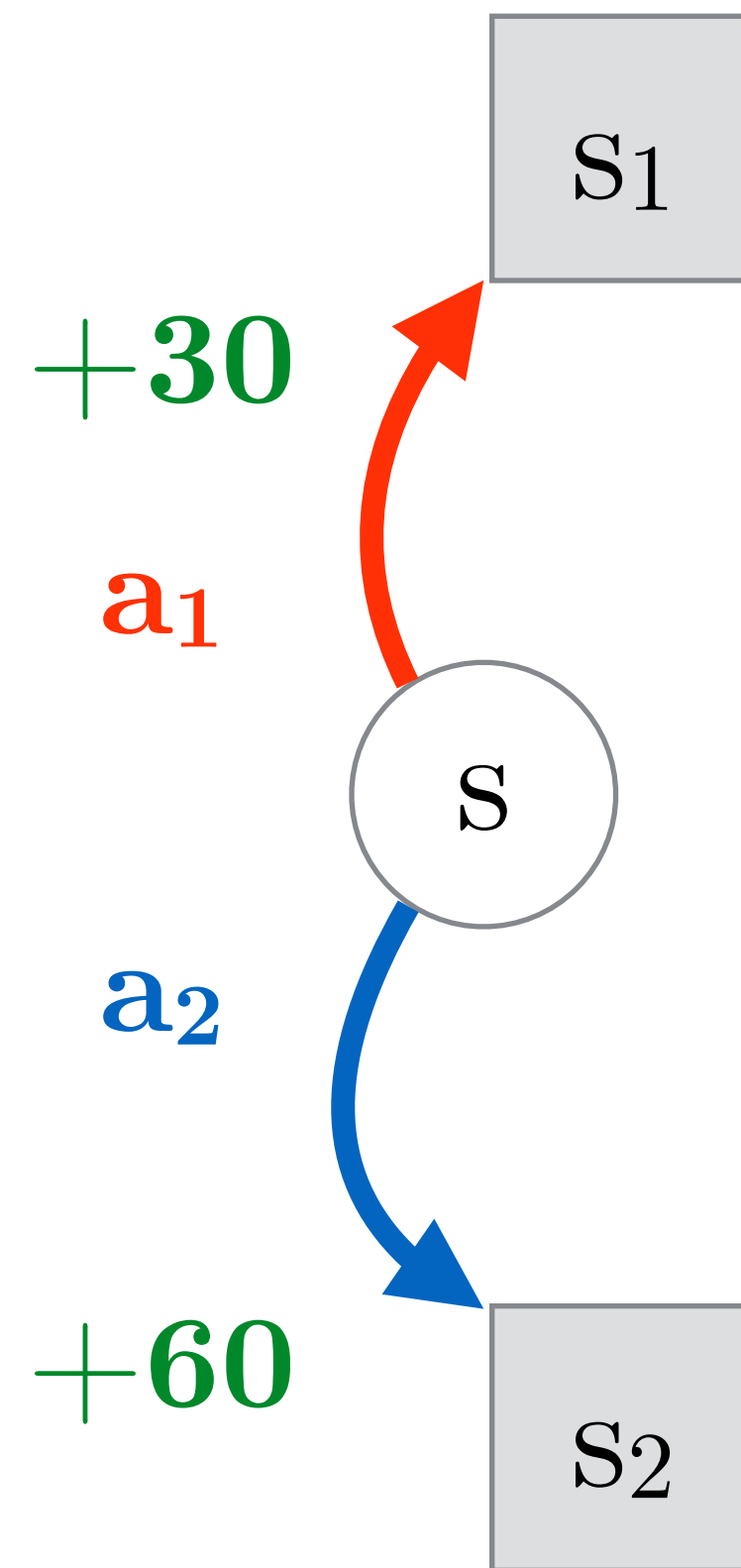
# Spotlight Overview: Flaw in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

$$v^\pi \neq \hat{v}$$

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

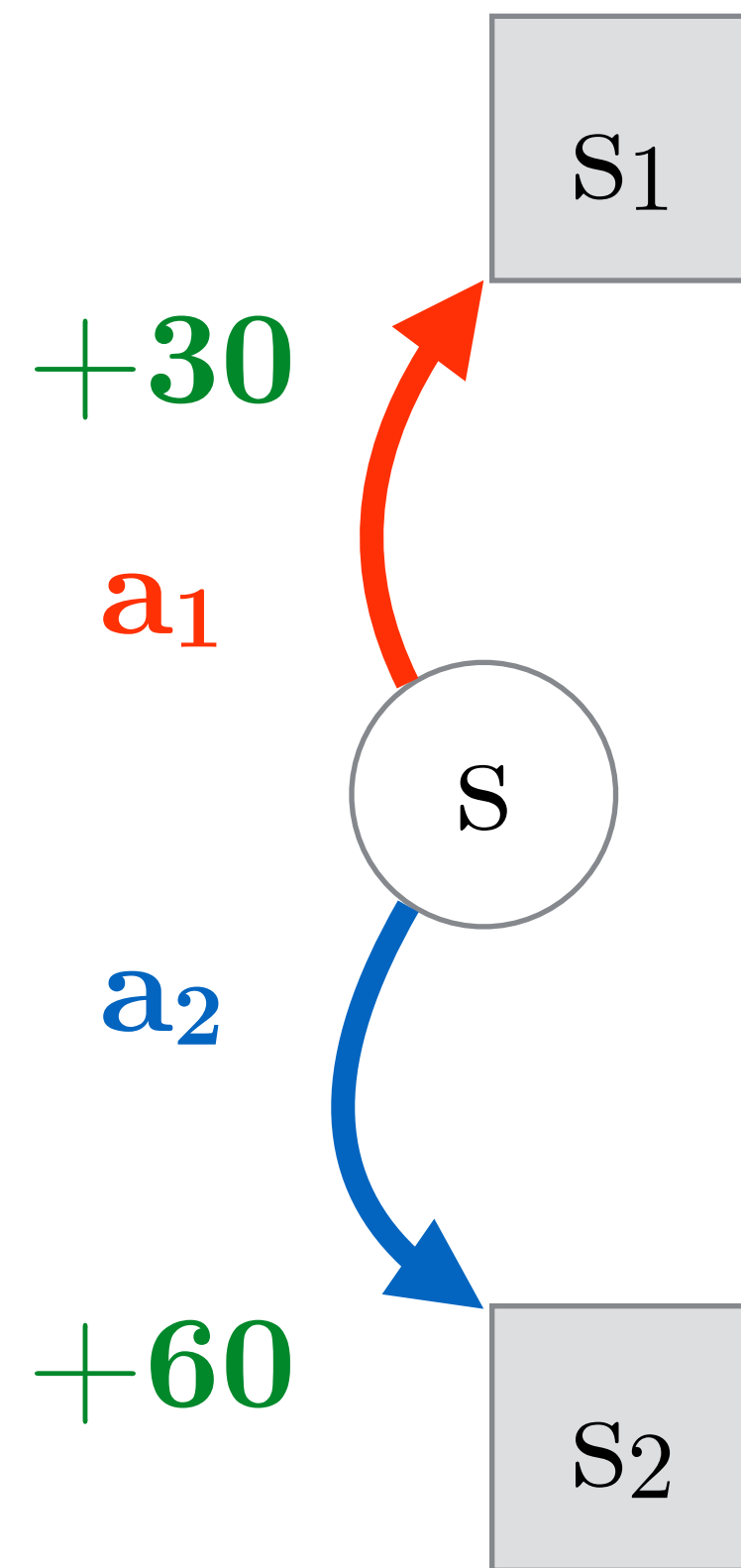
# Spotlight Overview: Flaw in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

$$v^\pi \neq \hat{v}$$

Batch TD(0) estimates value function for the **wrong** policy!

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

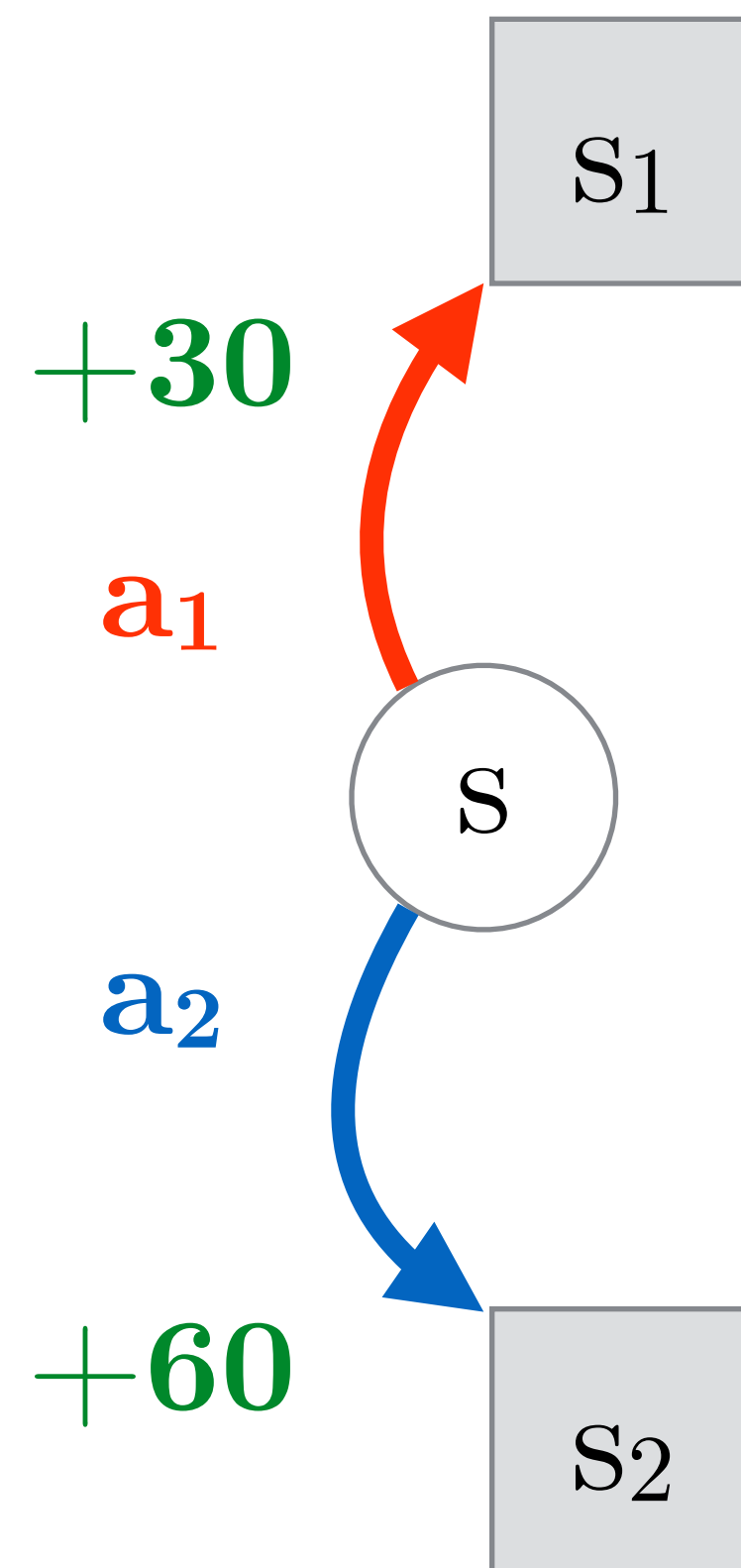
# Spotlight Overview: Flaw in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

$$v^\pi \neq \hat{v}$$

Batch TD(0) estimates value function for the **wrong** policy!

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

Our estimator will estimate value function for the **true** policy

# Batch Linear\* Value Function Learning

\*Empirical analysis also considers non-linear TD(0)



# Batch Linear\* Value Function Learning

Policy and environment transition dynamics:

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1] \quad P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$$

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* Value Function Learning

Policy and environment transition dynamics:

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1] \quad P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$$

Generates batch of  $m$  episodes:

$$\mathcal{D} := \{\tau_i\}_{i=1}^m \quad \text{where} \quad \tau := (s_1, a_1, r_1, \dots, s_{L_\tau}, a_{L_\tau}, r_{L_\tau})$$

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* Value Function Learning

Policy and environment transition dynamics:

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1] \quad P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$$

Generates batch of  $m$  episodes:

$$\mathcal{D} := \{\tau_i\}_{i=1}^m \quad \text{where} \quad \tau := (s_1, a_1, r_1, \dots, s_{L_\tau}, a_{L_\tau}, r_{L_\tau})$$

Estimate value function:

$$v^\pi(s) := \mathbf{E}_\pi \left[ \sum_{k=0}^L \gamma^k R_{t+k+1} \mid S_t = s \right], \forall s \in \mathcal{S}$$

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* Value Function Learning

Policy and environment transition dynamics:

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1] \quad P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$$

Generates batch of  $m$  episodes:

$$\mathcal{D} := \{\tau_i\}_{i=1}^m \quad \text{where} \quad \tau := (s_1, a_1, r_1, \dots, s_{L_\tau}, a_{L_\tau}, r_{L_\tau})$$

Estimate value function:

$$v^\pi(s) := \mathbf{E}_\pi \left[ \sum_{k=0}^L \gamma^k R_{t+k+1} \mid S_t = s \right], \forall s \in \mathcal{S}$$

$$\hat{v}(s) := \mathbf{w}^T \mathbf{x}(s)$$

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* Value Function Learning

Policy and environment transition dynamics:

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1] \quad P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$$

Generates batch of  $m$  episodes:

$$\mathcal{D} := \{\tau_i\}_{i=1}^m \quad \text{where} \quad \tau := (s_1, a_1, r_1, \dots, s_{L_\tau}, a_{L_\tau}, r_{L_\tau})$$

Estimate value function:

$$v^\pi(s) := \mathbf{E}_\pi \left[ \sum_{k=0}^L \gamma^k R_{t+k+1} \mid S_t = s \right], \forall s \in \mathcal{S}$$

$$\hat{v}(s) := \mathbf{w}^T \mathbf{x}(s)$$

Assumptions:

1.  $\pi$  is known (policy we want to learn about).
2.  $P$  is unknown (model-free).
3. Reward function is unknown.
4. On-policy (focus of talk).

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* TD(0)

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* TD(0)

---

**Algorithm 1** Batch Linear TD(0) to estimate  $v^\pi$

---

- 1: Input: batch  $\mathcal{D}$ , step-size  $\alpha > 0$ , convergence threshold  $\epsilon > 0$
  - 2: Initialize: weight vector  $\mathbf{w}_0$  arbitrarily (e.g.:  $\mathbf{w}_0 := \mathbf{0}$ ), update aggregation vector  $\mathbf{u} := \mathbf{0}$ , batch process counter,  $i = 0$
  - 3: **while**  $|\mathbf{w}_{i+1} - \mathbf{w}_i| \geq \epsilon \mathbf{1}$  **do**
  - 4:   **for** each episode,  $\tau \in \mathcal{D}$  **do**
  - 5:     **for** each transition,  $(s, a, r, s') \in \tau$  **do**
  - 6:        $\mathbf{u} \leftarrow \mathbf{u} + [r + \gamma \mathbf{w}_i^T \mathbf{x}(s') - \mathbf{w}_i^T \mathbf{x}(s)] \mathbf{x}(s)$
  - 7:     **end for**
  - 8:   **end for**
  - 9:    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha \mathbf{u}$  {batch update}
  - 10:    $\mathbf{u} \leftarrow \mathbf{0}$  {clear aggregation}
  - 11:    $i \leftarrow i + 1$  {update batch process counter}
  - 12: **end while**
- 

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* TD(0)

---

**Algorithm 1** Batch Linear TD(0) to estimate  $v^\pi$

---

- 1: **Input:** batch  $\mathcal{D}$ , step-size  $\alpha > 0$ , convergence threshold  $\epsilon > 0$
- 2: Initialize: weight vector  $\mathbf{w}_0$  arbitrarily (e.g.:  $\mathbf{w}_0 := \mathbf{0}$ ), update aggregation vector  $\mathbf{u} := \mathbf{0}$ , batch process counter,  $i = 0$
- 3: **while**  $|\mathbf{w}_{i+1} - \mathbf{w}_i| \geq \epsilon \mathbf{1}$  **do**
- 4:   **for** each episode,  $\tau \in \mathcal{D}$  **do**
- 5:     **for** each transition,  $(s, a, r, s') \in \tau$  **do**
- 6:        $\mathbf{u} \leftarrow \mathbf{u} + [r + \gamma \mathbf{w}_i^T \mathbf{x}(s') - \mathbf{w}_i^T \mathbf{x}(s)] \mathbf{x}(s)$
- 7:     **end for**
- 8:   **end for**
- 9:    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha \mathbf{u}$  {batch update}
- 10:    $\mathbf{u} \leftarrow \mathbf{0}$  {clear aggregation}
- 11:    $i \leftarrow i + 1$  {update batch process counter}
- 12: **end while**

fixed finite batch as input

---

\*Empirical analysis also considers non-linear TD(0)



# Batch Linear\* TD(0)

---

**Algorithm 1** Batch Linear TD(0) to estimate  $v^\pi$

---

- 1: Input: batch  $\mathcal{D}$ , step-size  $\alpha > 0$ , convergence threshold  $\epsilon > 0$
- 2: Initialize: weight vector  $\mathbf{w}_0$  arbitrarily (e.g.:  $\mathbf{w}_0 := \mathbf{0}$ ), update aggregation vector  $\mathbf{u} := \mathbf{0}$ , batch process counter,  $i = 0$
- 3: **while**  $|\mathbf{w}_{i+1} - \mathbf{w}_i| > \epsilon \mathbf{1}$  **do**
- 4:   **for** each episode,  $\tau \in \mathcal{D}$  **do**
- 5:     **for** each transition,  $(s, a, r, s') \in \tau$  **do**
- 6:        $\mathbf{u} \leftarrow \mathbf{u} + [r + \gamma \mathbf{w}_i^T \mathbf{x}(s') - \mathbf{w}_i^T \mathbf{x}(s)] \mathbf{x}(s)$
- 7:     **end for**
- 8:   **end for**
- 9:    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha \mathbf{u}$  {batch update}
- 10:    $\mathbf{u} \leftarrow \mathbf{0}$  {clear aggregation}
- 11:    $i \leftarrow i + 1$  {update batch process counter}
- 12: **end while**

for each transition

---

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* TD(0)

---

**Algorithm 1** Batch Linear TD(0) to estimate  $v^\pi$

---

```
1: Input: batch  $\mathcal{D}$ , step-size  $\alpha > 0$ , convergence threshold  
    $\epsilon > 0$   
2: Initialize: weight vector  $\mathbf{w}_0$  arbitrarily (e.g.:  $\mathbf{w}_0 :=$   
    $\mathbf{0}$ ), update aggregation vector  $\mathbf{u} := \mathbf{0}$ , batch process  
   counter,  $i = 0$   
3: while  $|\mathbf{w}_{i+1} - \mathbf{w}_i| \geq \epsilon \mathbf{1}$  do  
4:   for each episode,  $\tau \in \mathcal{D}$  do  
5:     for each transition,  $(s, a, r, s') \in \tau$  do  
6:        $\mathbf{u} \leftarrow \mathbf{u} + [r + \gamma \mathbf{w}_i^T \mathbf{x}(s') - \mathbf{w}_i^T \mathbf{x}(s)] \mathbf{x}(s)$   
7:     end for  
8:   end for  
9:    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha \mathbf{u}$  {batch update}  
10:   $\mathbf{u} \leftarrow \mathbf{0}$  {clear aggregation}  
11:   $i \leftarrow i + 1$  {update batch process counter}  
12: end while
```

accumulate computed TD error

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* TD(0)

---

**Algorithm 1** Batch Linear TD(0) to estimate  $v^\pi$

---

- 1: Input: batch  $\mathcal{D}$ , step-size  $\alpha > 0$ , convergence threshold  $\epsilon > 0$
- 2: Initialize: weight vector  $\mathbf{w}_0$  arbitrarily (e.g.:  $\mathbf{w}_0 := \mathbf{0}$ ), update aggregation vector  $\mathbf{u} := \mathbf{0}$ , batch process counter,  $i = 0$
- 3: **while**  $|\mathbf{w}_{i+1} - \mathbf{w}_i| \geq \epsilon \mathbf{1}$  **do**
- 4:   **for** each episode,  $\tau \in \mathcal{D}$  **do**
- 5:     **for** each transition,  $(s, a, r, s') \in \tau$  **do**
- 6:        $\mathbf{u} \leftarrow \mathbf{u} + [r + \gamma \mathbf{w}_i^T \mathbf{x}(s') - \mathbf{w}_i^T \mathbf{x}(s)] \mathbf{x}(s)$
- 7:     **end for**
- 8:   **end for**
- 9:    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha \mathbf{u}$  {batch update}
- 10:    $\mathbf{u} \leftarrow \mathbf{0}$  {clear aggregation}
- 11:    $i \leftarrow i + 1$  {update batch process counter}
- 12: **end while**

make aggregated update to weights

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* TD(0)

---

**Algorithm 1** Batch Linear TD(0) to estimate  $v^\pi$

---

```
1: Input: batch  $\mathcal{D}$ , step-size  $\alpha > 0$ , convergence threshold  
    $\epsilon > 0$   
2: Initialize: weight vector  $\mathbf{w}_0$  arbitrarily (e.g.:  $\mathbf{w}_0 :=$   
    $\mathbf{0}$ ), update aggregation vector  $\mathbf{u} := \mathbf{0}$ , batch process  
   counter,  $i = 0$   
3: while  $|\mathbf{w}_{i+1} - \mathbf{w}_i| \geq \epsilon \mathbf{1}$  do  
4:   for each episode,  $\tau \in \mathcal{D}$  do  
5:     for each transition,  $(s, a, r, s') \in \tau$  do  
6:        $\mathbf{u} \leftarrow \mathbf{u} + [r + \gamma \mathbf{w}_i^T \mathbf{x}(s') - \mathbf{w}_i^T \mathbf{x}(s)] \mathbf{x}(s)$   
7:     end for  
8:   end for  
9:    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha \mathbf{u}$  {batch update}  
10:   $\mathbf{u} \leftarrow \mathbf{0}$  {clear aggregation}  
11:   $i \leftarrow i + 1$  {update batch process counter}  
12: end while
```

clear aggregation

\*Empirical analysis also considers non-linear TD(0)

# Batch Linear\* TD(0)

---

**Algorithm 1** Batch Linear TD(0) to estimate  $v^\pi$

---

- 1: Input: batch  $\mathcal{D}$ , step-size  $\alpha > 0$ , convergence threshold  $\epsilon > 0$
- 2: Initialize: weight vector  $\mathbf{w}_0$  arbitrarily (e.g.:  $\mathbf{w}_0 := \mathbf{0}$ ), update aggregation vector  $\mathbf{u} := \mathbf{0}$ , batch process counter,  $i = 0$
- 3: **while**  $|\mathbf{w}_{i+1} - \mathbf{w}_i| \geq \epsilon \mathbf{1}$  **do**
- 4:   **for** each episode,  $\tau \in \mathcal{D}$  **do**
- 5:     **for** each transition,  $(s, a, r, s') \in \tau$  **do**
- 6:        $\mathbf{u} \leftarrow \mathbf{u} + [r + \gamma \mathbf{w}_i^T \mathbf{x}(s') - \mathbf{w}_i^T \mathbf{x}(s)] \mathbf{x}(s)$
- 7:     **end for**
- 8:   **end for**
- 9:    $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha \mathbf{u}$  {batch update}
- 10:    $\mathbf{u} \leftarrow \mathbf{0}$  {clear aggregation}
- 11:    $i \leftarrow i + 1$  {update batch process counter}
- 12: **end while**

until convergence

---

\*Empirical analysis also considers non-linear TD(0)

# Batch TD(0) Value Function

finite-sized  $\mathcal{D}$   $\longrightarrow$  batch TD(0)  $\longrightarrow$   $\hat{v}(s)$

# Batch TD(0) Value Function

finite-sized  $\mathcal{D} \longrightarrow$  batch TD(0)  $\longrightarrow \hat{v}(s)$

certainty-equivalence estimate for MDP\*

$$\hat{v}(s_j) = \sum_{a \in \hat{\mathcal{A}}} \hat{\pi}(a|s_j) \left( \bar{R}(s_j, a) + \gamma \sum_{k \in \hat{\mathcal{S}}} \hat{P}(s_k|s_j, a) \hat{v}(s_k) \right), \forall s_j, s_k \in \hat{\mathcal{S}}$$

\*Sutton (1988) proved a similar result for a Markov reward process

# Batch TD(0) Value Function

finite-sized  $\mathcal{D}$   $\longrightarrow$  batch TD(0)  $\longrightarrow$   $\hat{v}(s)$

certainty-equivalence estimate for MDP\*

$$\hat{v}(s_j) = \sum_{a \in \hat{\mathcal{A}}} \hat{\pi}(a|s_j) \left( \bar{R}(s_j, a) + \gamma \sum_{k \in \hat{\mathcal{S}}} \hat{P}(s_k|s_j, a) \hat{v}(s_k) \right), \forall s_j, s_k \in \hat{\mathcal{S}}$$

maximum-likelihood estimates (MLE) computed from  $\mathcal{D}$

\*Sutton (1988) proved a similar result for a Markov reward process



# Batch TD(0) Value Function

finite-sized  $\mathcal{D}$   $\longrightarrow$  batch TD(0)  $\longrightarrow$   $\hat{v}(s)$

certainty-equivalence estimate for MDP\*

$$\hat{v}(s_j) = \sum_{a \in \hat{\mathcal{A}}} \hat{\pi}(a|s_j) \left( \bar{R}(s_j, a) + \gamma \sum_{k \in \hat{\mathcal{S}}} \hat{P}(s_k|s_j, a) \hat{v}(s_k) \right), \forall s_j, s_k \in \hat{\mathcal{S}}$$

maximum-likelihood estimates (MLE) computed from  $\mathcal{D}$

Problem!

\*Sutton (1988) proved a similar result for a Markov reward process

# Batch TD(0) Value Function

finite-sized  $\mathcal{D}$   $\longrightarrow$  batch TD(0)  $\longrightarrow$   $\hat{v}(s)$

certainty-equivalence estimate for MDP\*

$$\hat{v}(s_j) = \sum_{a \in \hat{\mathcal{A}}} \hat{\pi}(a|s_j) \left( \bar{R}(s_j, a) + \gamma \sum_{k \in \hat{\mathcal{S}}} \hat{P}(s_k|s_j, a) \hat{v}(s_k) \right), \forall s_j, s_k \in \hat{\mathcal{S}}$$

maximum-likelihood estimates (MLE) computed from  $\mathcal{D}$

Problem!  $\hat{\pi} \neq \pi$   $\hat{P} \neq P$

\*Sutton (1988) proved a similar result for a Markov reward process

# Batch TD(0) Value Function

finite-sized  $\mathcal{D}$   $\longrightarrow$  batch TD(0)  $\longrightarrow$   $\hat{v}(s)$

certainty-equivalence estimate for MDP\*

$$\hat{v}(s_j) = \sum_{a \in \hat{\mathcal{A}}} \hat{\pi}(a|s_j) \left( \bar{R}(s_j, a) + \gamma \sum_{k \in \hat{\mathcal{S}}} \hat{P}(s_k|s_j, a) \hat{v}(s_k) \right), \forall s_j, s_k \in \hat{\mathcal{S}}$$

maximum-likelihood estimates (MLE) computed from  $\mathcal{D}$

Problem!  $\hat{\pi} \neq \pi$   $\hat{P} \neq P$  policy and transition dynamics  
*sampling error*

\*Sutton (1988) proved a similar result for a Markov reward process

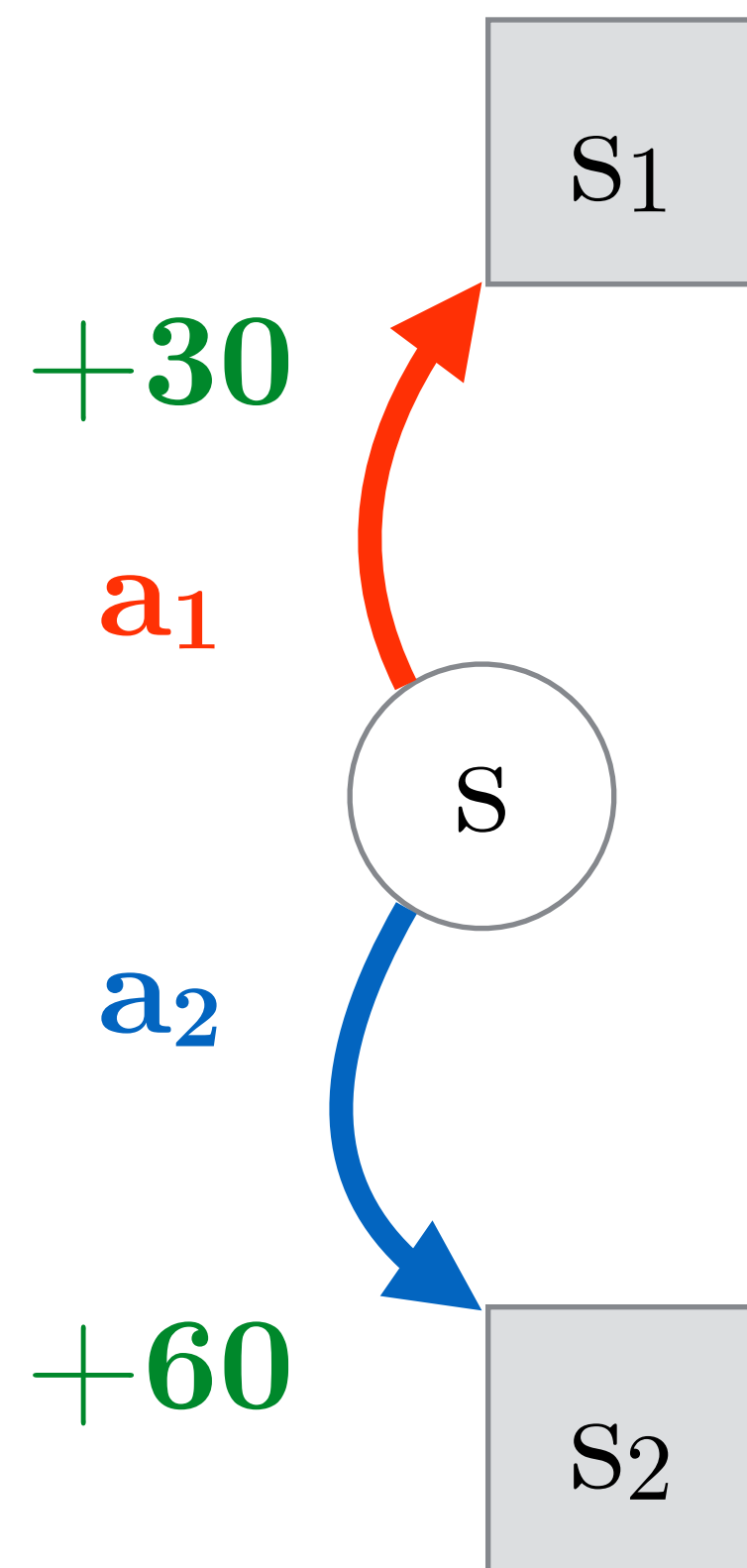
# Policy Sampling Error in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

Batch TD(0) estimates  
value function for the  
**wrong** policy!

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

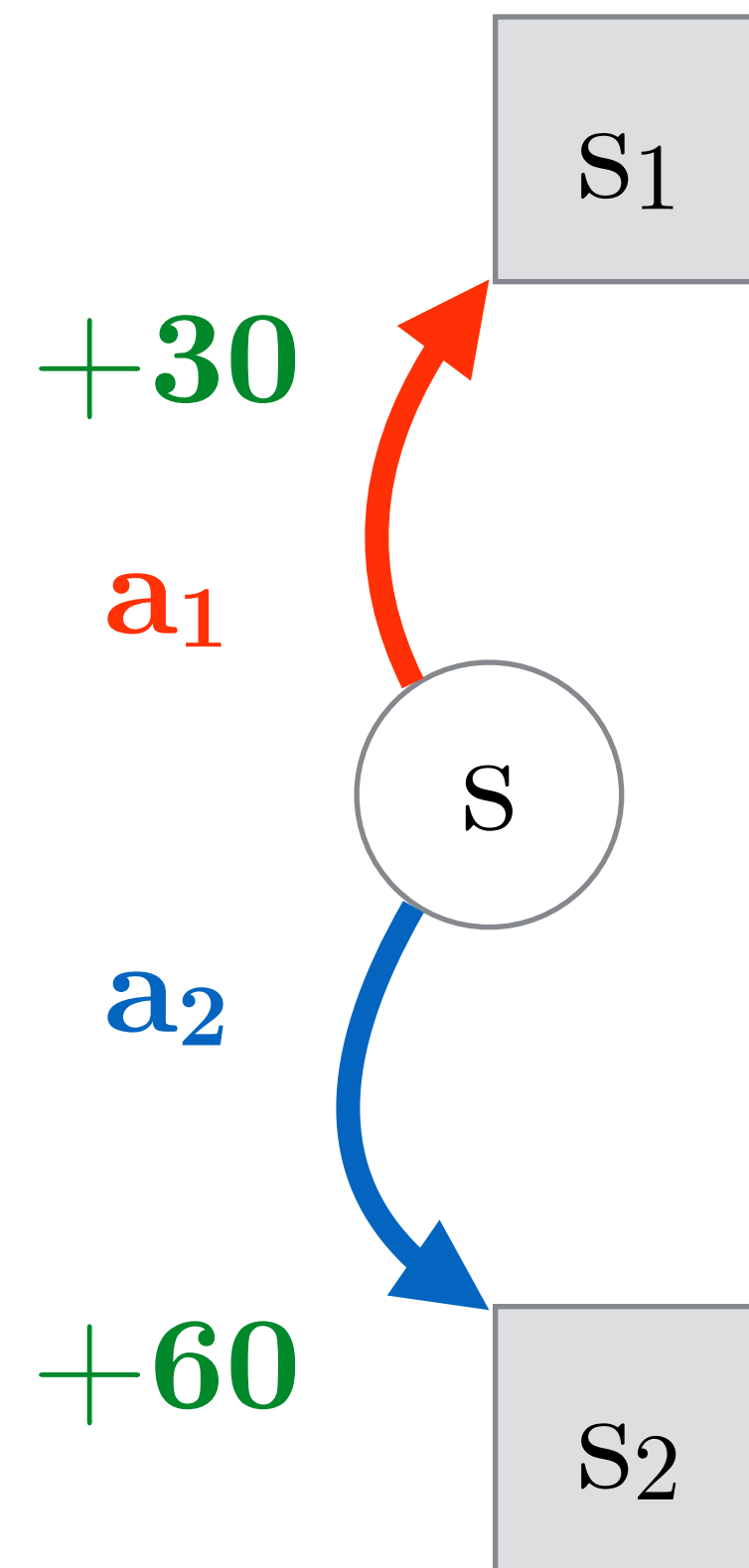
# Policy Sampling Error in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

MLE policy

$$\hat{\pi}(a_1|s) = \frac{2}{3}$$

$$\hat{\pi}(a_2|s) = \frac{1}{3}$$

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

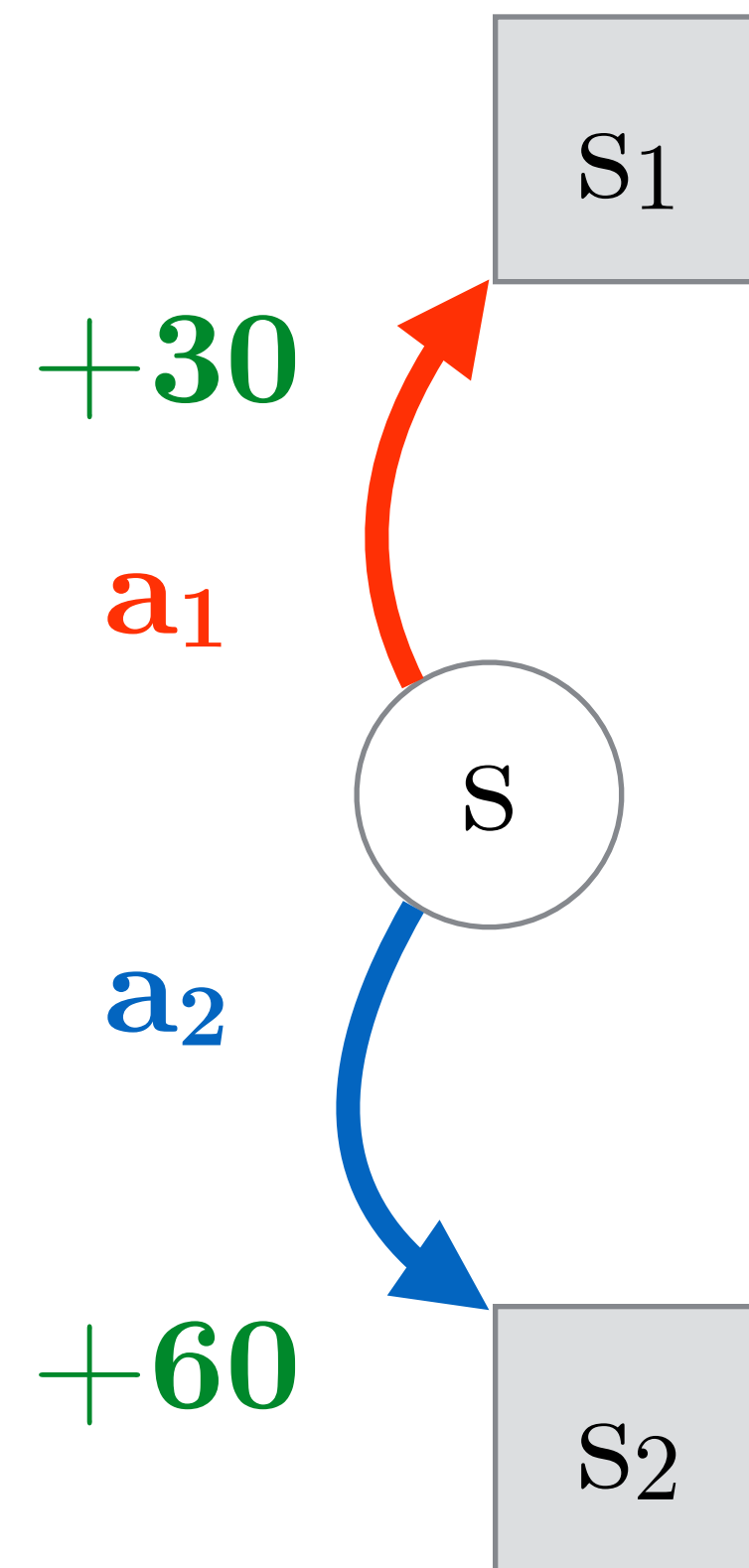
# Policy Sampling Error in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

MLE policy

$$\hat{\pi}(a_1|s) = \frac{2}{3}$$

$$\hat{\pi}(a_2|s) = \frac{1}{3}$$

$$\pi \neq \hat{\pi}$$

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

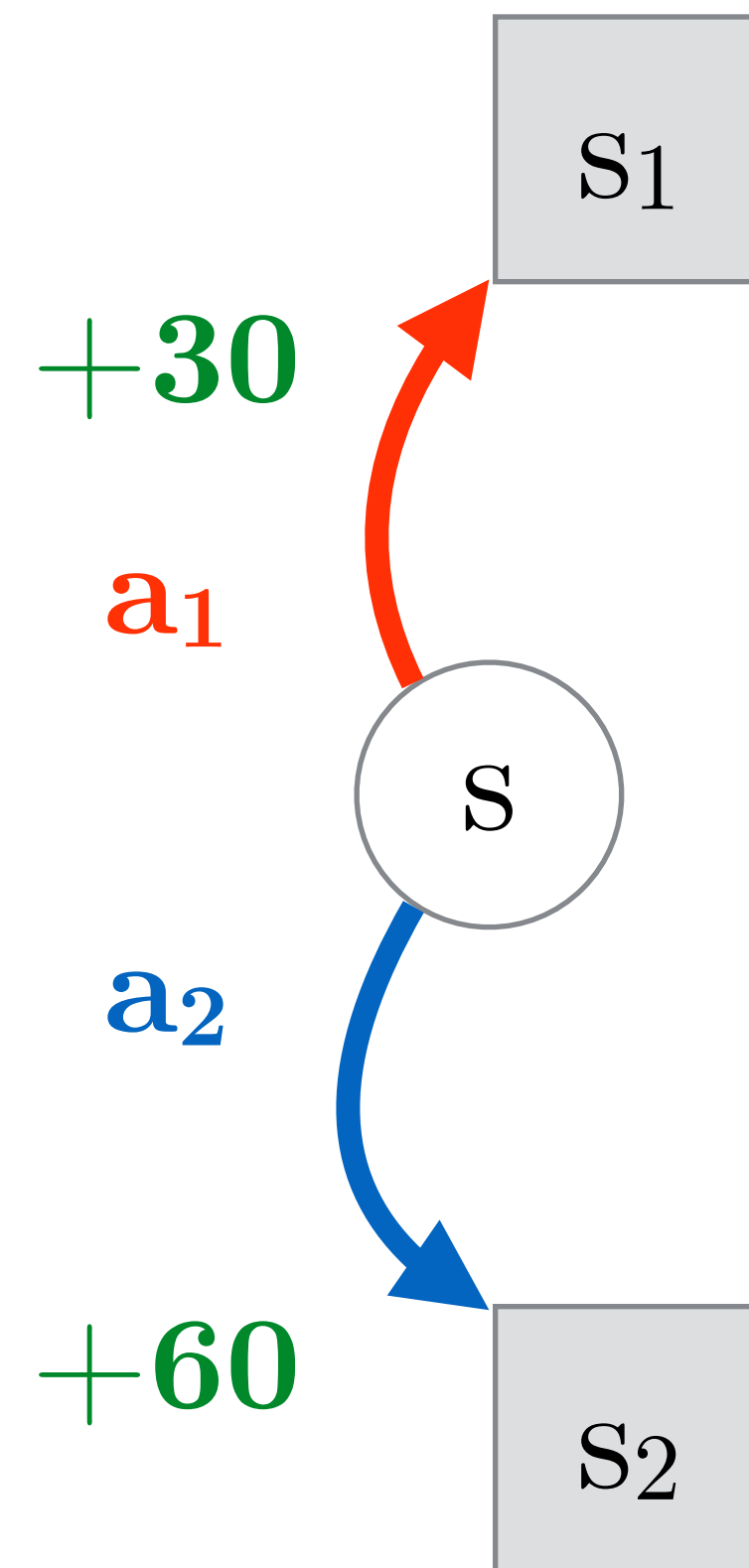
# Policy Sampling Error in Batch TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

MLE policy

$$\hat{\pi}(a_1|s) = \frac{2}{3}$$

$$\hat{\pi}(a_2|s) = \frac{1}{3}$$

$$\pi \neq \hat{\pi}$$

$$v^\pi \neq \hat{v}$$

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

# Policy Sampling Error Corrected-TD(0)



# Policy Sampling Error Corrected-TD(0)

**True** policy distribution is assumed to be known.

# Policy Sampling Error Corrected-TD(0)

**True** policy distribution is assumed to be known.

Correct learning from the **MLE** policy distribution to the **true** policy distribution.

# Policy Sampling Error Corrected-TD(0)

**True** policy distribution is assumed to be known.

Correct learning from the **MLE** policy distribution to the **true** policy distribution.

An **off-policy-styled** correction for an **on-policy** algorithm.

# Policy Sampling Error Corrected-TD(0)

**True** policy distribution is assumed to be known.

Correct learning from the **MLE** policy distribution to the **true** policy distribution.

An **off-policy-styled** correction for an **on-policy** algorithm.

PSEC ratio (importance sampling [Precup et al., 2000, Ghiassian et al., 2018]):

$$\mathbf{u} \leftarrow \mathbf{u} + [r + \gamma \mathbf{w}_i^T \mathbf{x}(s') - \mathbf{w}_i^T \mathbf{x}(s)] \mathbf{x}(s) \quad \mathbf{u} \leftarrow \mathbf{u} + \left[ \frac{\pi(a|s)}{\hat{\pi}(a|s)} (r + \gamma \mathbf{w}_i^T \mathbf{x}(s')) - \mathbf{w}_i^T \mathbf{x}(s) \right] \mathbf{x}(s)$$

**On-policy** TD(0) Update

**On-policy** PSEC-TD(0) Update

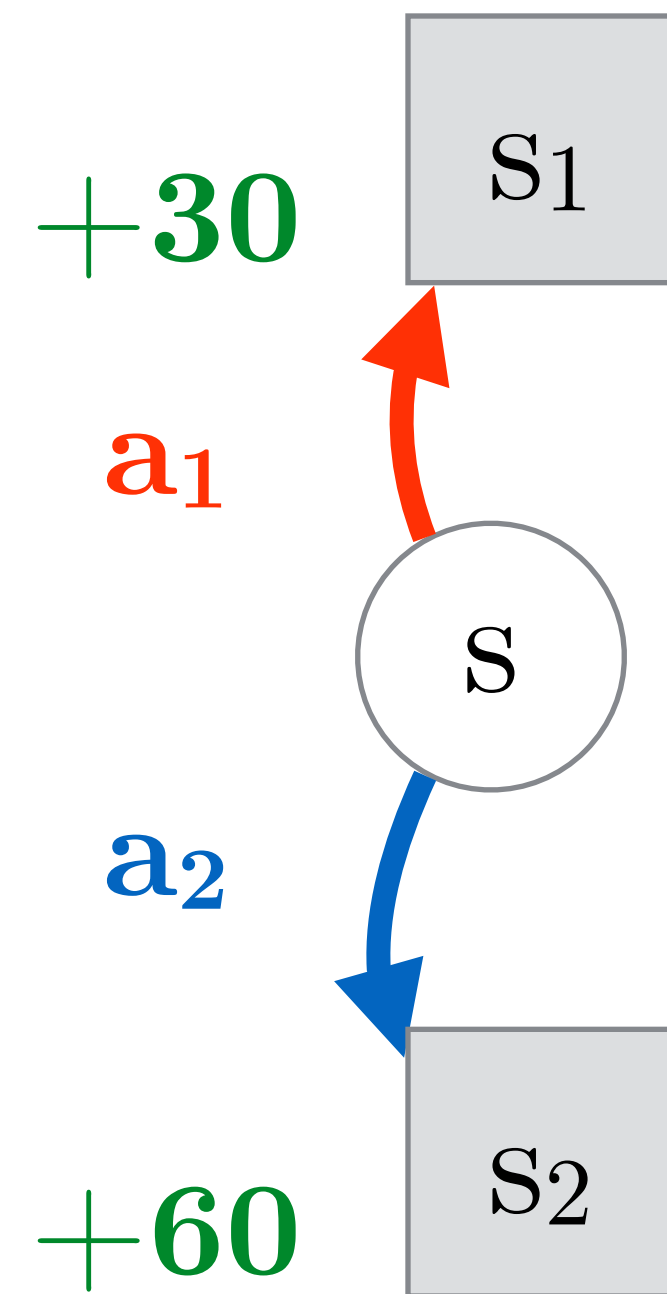
# Batch PSEC-TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

MLE policy

$$\hat{\pi}(a_1|s) = \frac{2}{3}$$

$$\hat{\pi}(a_2|s) = \frac{1}{3}$$

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

PSEC-TD(0) Update

$$\mathbf{u} \leftarrow \mathbf{u} + \left[ \frac{\pi(a|s)}{\hat{\pi}(a|s)} (r + \gamma \mathbf{w}_i^T \mathbf{x}(s')) - \mathbf{w}_i^T \mathbf{x}(s) \right] \mathbf{x}(s)$$

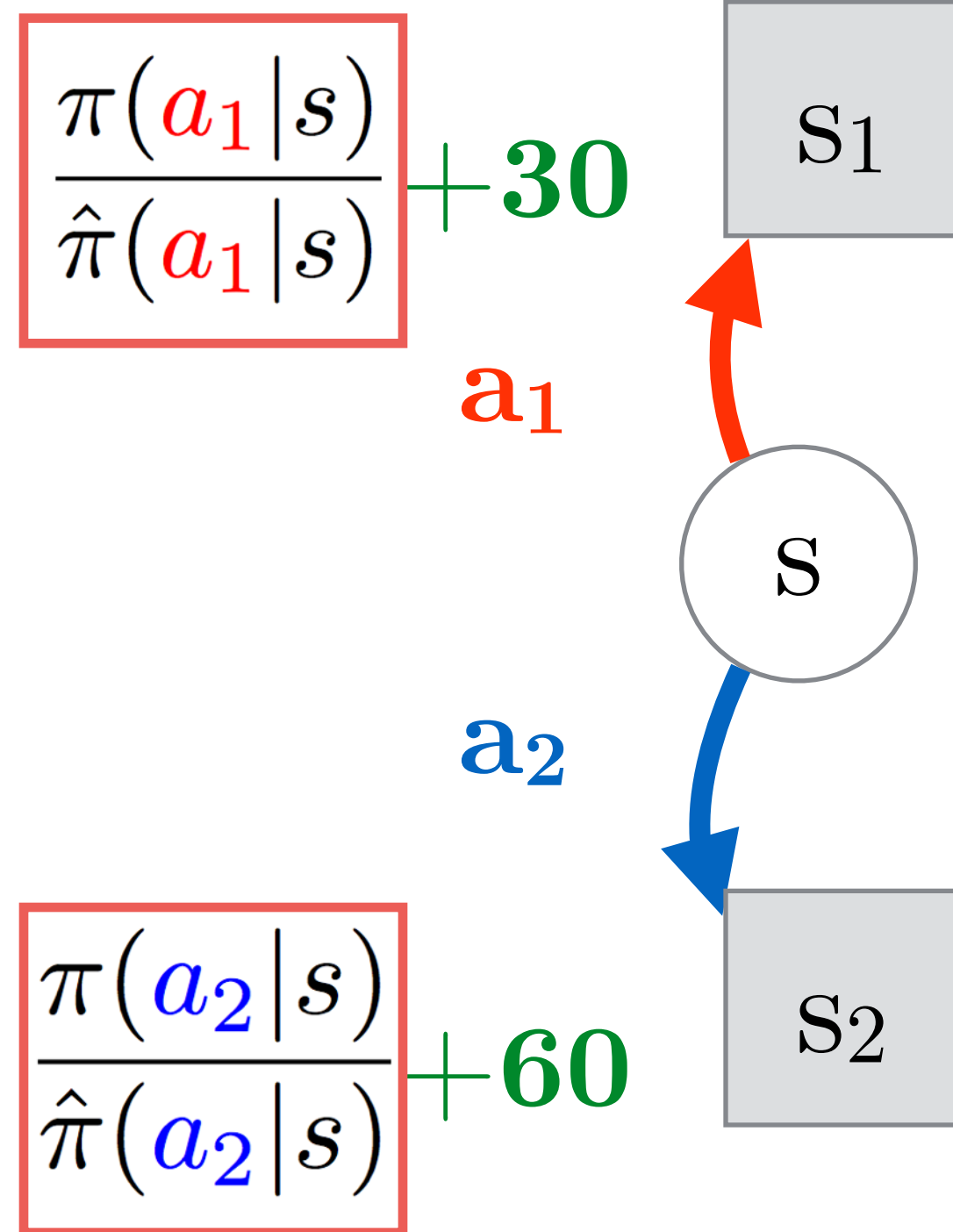
# Batch PSEC-TD(0)

True policy

$$\pi(\cdot|s) = \frac{1}{2}$$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

MLE policy

$$\hat{\pi}(a_1|s) = \frac{2}{3}$$

$$\hat{\pi}(a_2|s) = \frac{1}{3}$$

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

PSEC-TD(0) Update

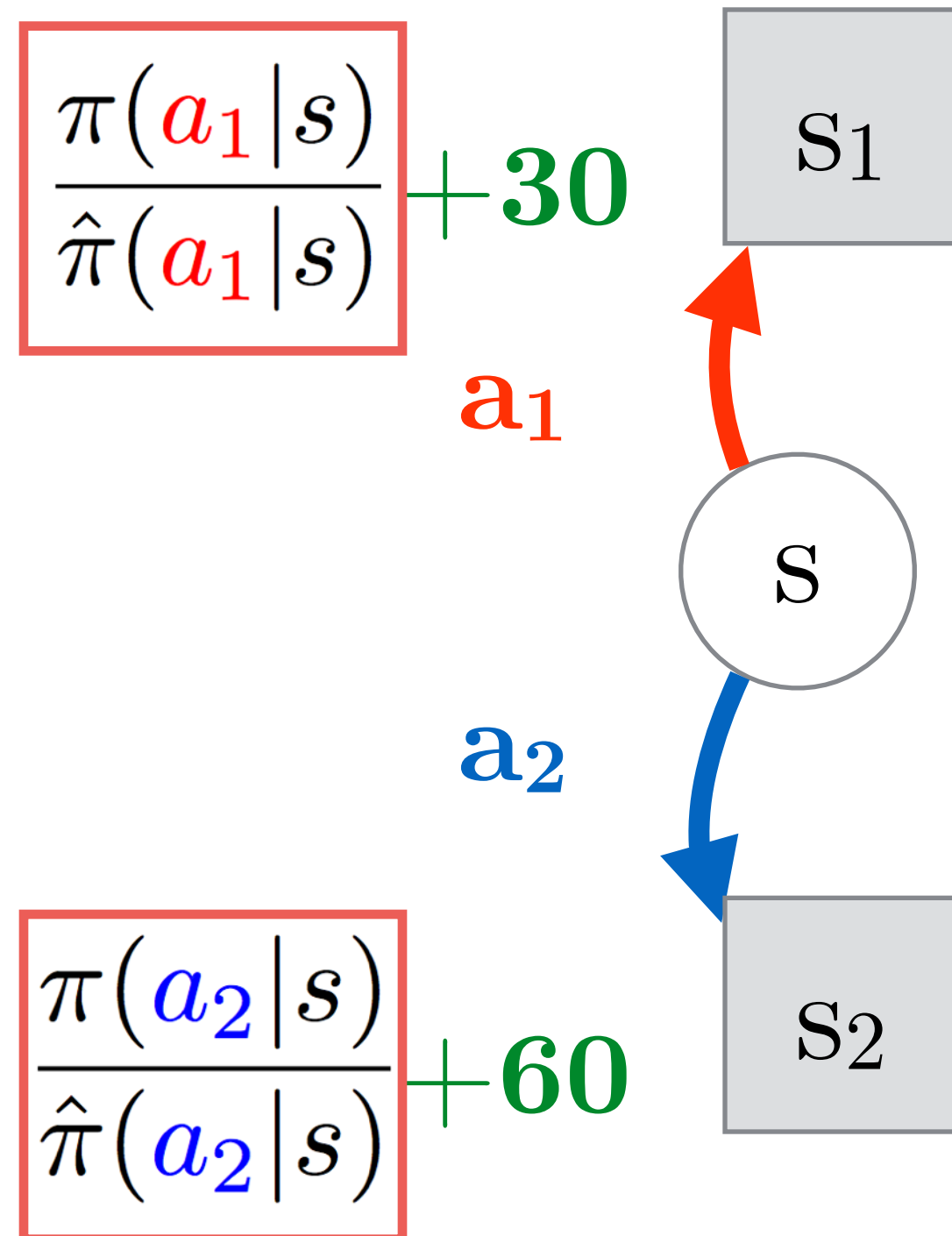
$$\mathbf{u} \leftarrow \mathbf{u} + \left[ \frac{\pi(a|s)}{\hat{\pi}(a|s)} (r + \gamma \mathbf{w}_i^T \mathbf{x}(s')) - \mathbf{w}_i^T \mathbf{x}(s) \right] \mathbf{x}(s)$$

# Batch PSEC-TD(0)

True policy  
 $\pi(\cdot|s) = \frac{1}{2}$

True value function

$$v^\pi(s) = 45$$



Batch TD(0) computes

$$\hat{v}(s) = 40$$

MLE policy

$$\hat{\pi}(a_1|s) = \frac{2}{3}$$

$$\hat{\pi}(a_2|s) = \frac{1}{3}$$

finite-sized batch

$$(s, a_1, s_1)$$

$$(s, a_1, s_1)$$

$$(s, a_2, s_2)$$

Batch PSEC-TD(0) computes

$$\hat{v}(s) = 45$$

PSEC-TD(0) Update

$$\mathbf{u} \leftarrow \mathbf{u} + \left[ \frac{\pi(a|s)}{\hat{\pi}(a|s)} (r + \gamma \mathbf{w}_i^T \mathbf{x}(s')) - \mathbf{w}_i^T \mathbf{x}(s) \right] \mathbf{x}(s)$$

# Batch PSEC-TD(0) Value Function

**Theorem 3** (Batch Linear PSEC-TD(0) Convergence). *For any batch whose observation vectors  $\{\mathbf{x}(s)|s \in \hat{\mathcal{S}}\}$  are linearly independent, there exists an  $\epsilon > 0$  such that, for all positive  $\alpha < \epsilon$  and for any initial weight vector, the predictions for linear PSEC-TD(0) converge under repeated presentations of the batch with weight updates after each complete presentation to the fixed-point (6).*

**Definition 3.** *PSEC Markov Decision Process Certainty Equivalence Estimate (PSEC-MDP-CEE) Value Function. The PSEC-MDP-CEE is the value function,  $\hat{v}^\pi$ , that,  $\forall s_j, s_k \in \hat{\mathcal{S}}$ , satisfies:*

$$\hat{v}^\pi(s_j) = \sum_{a \in \hat{\mathcal{A}}} \pi(a|s_j) [\bar{R}(s_j, a) + \gamma \sum_{k \in \hat{\mathcal{S}}} \hat{P}(s_k|s_j, a) \hat{v}^\pi(s_k)] \quad (6)$$



# Batch PSEC-TD(0) Value Function

**Theorem 3** (Batch Linear PSEC-TD(0) Convergence). *For any batch whose observation vectors  $\{\mathbf{x}(s) | s \in \hat{\mathcal{S}}\}$  are linearly independent, there exists an  $\epsilon > 0$  such that, for all positive  $\alpha < \epsilon$  and for any initial weight vector, the predictions for **linear PSEC-TD(0) converge** under repeated presentations of the batch with weight updates after each complete presentation to the fixed-point (6).*

**Definition 3.** *PSEC Markov Decision Process Certainty Equivalence Estimate (PSEC-MDP-CEE) Value Function. The PSEC-MDP-CEE is the value function,  $\hat{v}^\pi$ , that,  $\forall s_j, s_k \in \hat{\mathcal{S}}$ , satisfies:*

$$\hat{v}^\pi(s_j) = \sum_{a \in \hat{\mathcal{A}}} \pi(a | s_j) [\bar{R}(s_j, a) + \gamma \sum_{k \in \hat{\mathcal{S}}} \hat{P}(s_k | s_j, a) \hat{v}^\pi(s_k)] \quad (6)$$

# Batch PSEC-TD(0) Value Function

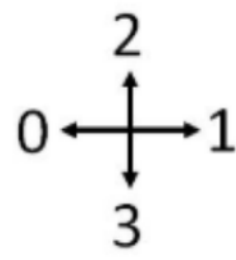
**Theorem 3** (Batch Linear PSEC-TD(0) Convergence). *For any batch whose observation vectors  $\{\mathbf{x}(s) | s \in \hat{\mathcal{S}}\}$  are linearly independent, there exists an  $\epsilon > 0$  such that, for all positive  $\alpha < \epsilon$  and for any initial weight vector, the predictions for linear PSEC-TD(0) converge under repeated presentations of the batch with weight updates after each complete presentation to the fixed-point (6).*

**Definition 3.** *PSEC Markov Decision Process Certainty Equivalence Estimate (PSEC-MDP-CEE) Value Function. The PSEC-MDP-CEE is the value function,  $\hat{v}^\pi$ , that,  $\forall s_j, s_k \in \hat{\mathcal{S}}$ , satisfies:*

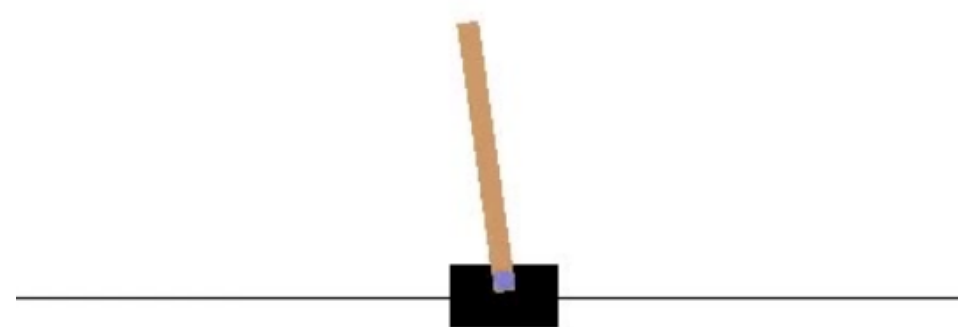
$$\hat{v}^\pi(s_j) = \sum_{a \in \hat{\mathcal{A}}} \pi(a|s_j) [\bar{R}(s_j, a) + \gamma \sum_{k \in \hat{\mathcal{S}}} \hat{P}(s_k | s_j, a) \hat{v}^\pi(s_k)] \quad (6)$$

# Experimental Results

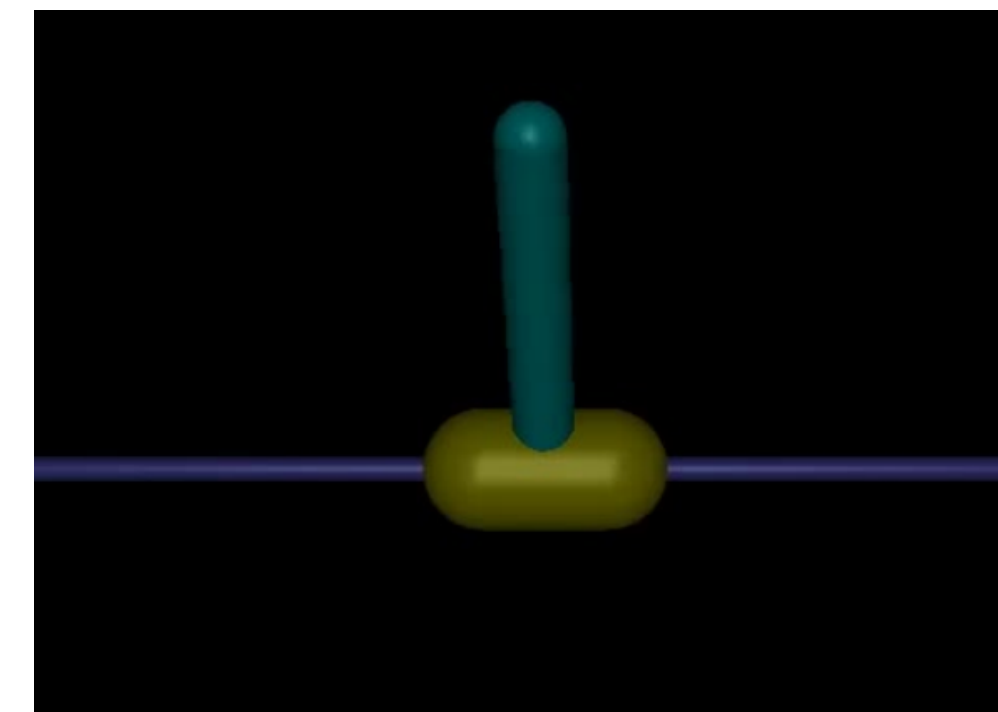
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15



Gridworld [[link](#)]  
**Discrete** States and Actions



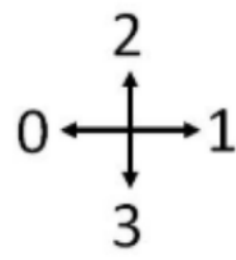
CartPole [Brockman et al., 2016]  
**Continuous** States and  
**Discrete** Actions



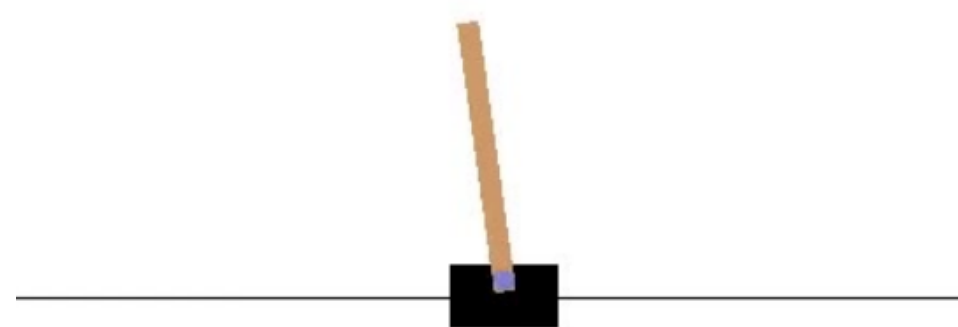
InvertedPendulum  
[Brockman et al., 2016,  
Todorov et al., 2012]  
**Continuous** States and  
Actions

# Experimental Results

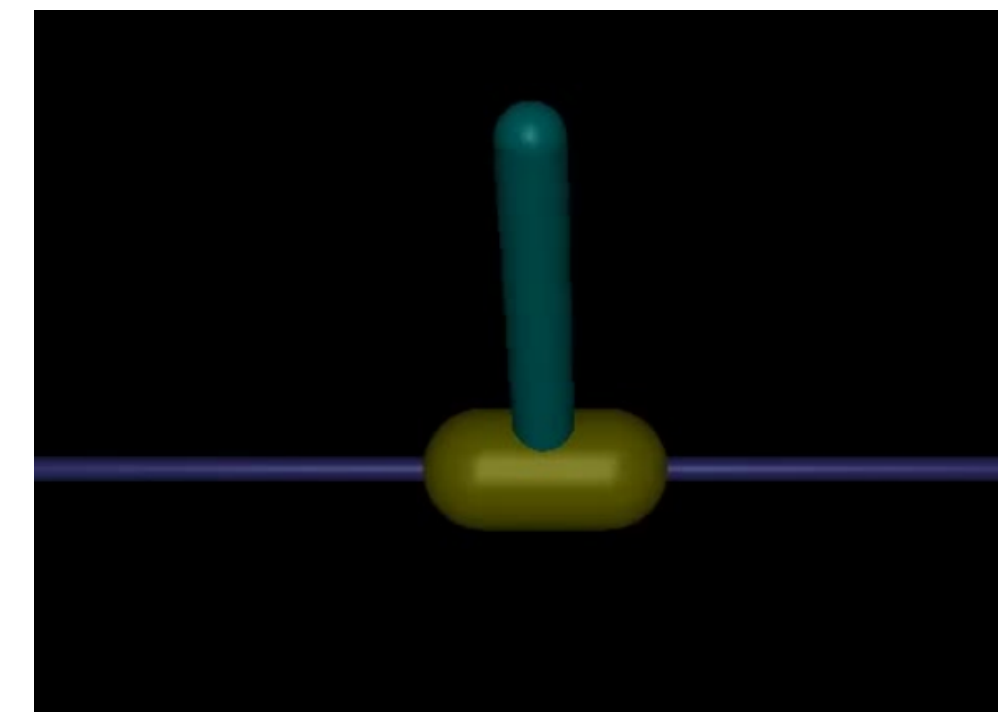
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15



Gridworld [[link](#)]  
**Discrete** States and Actions



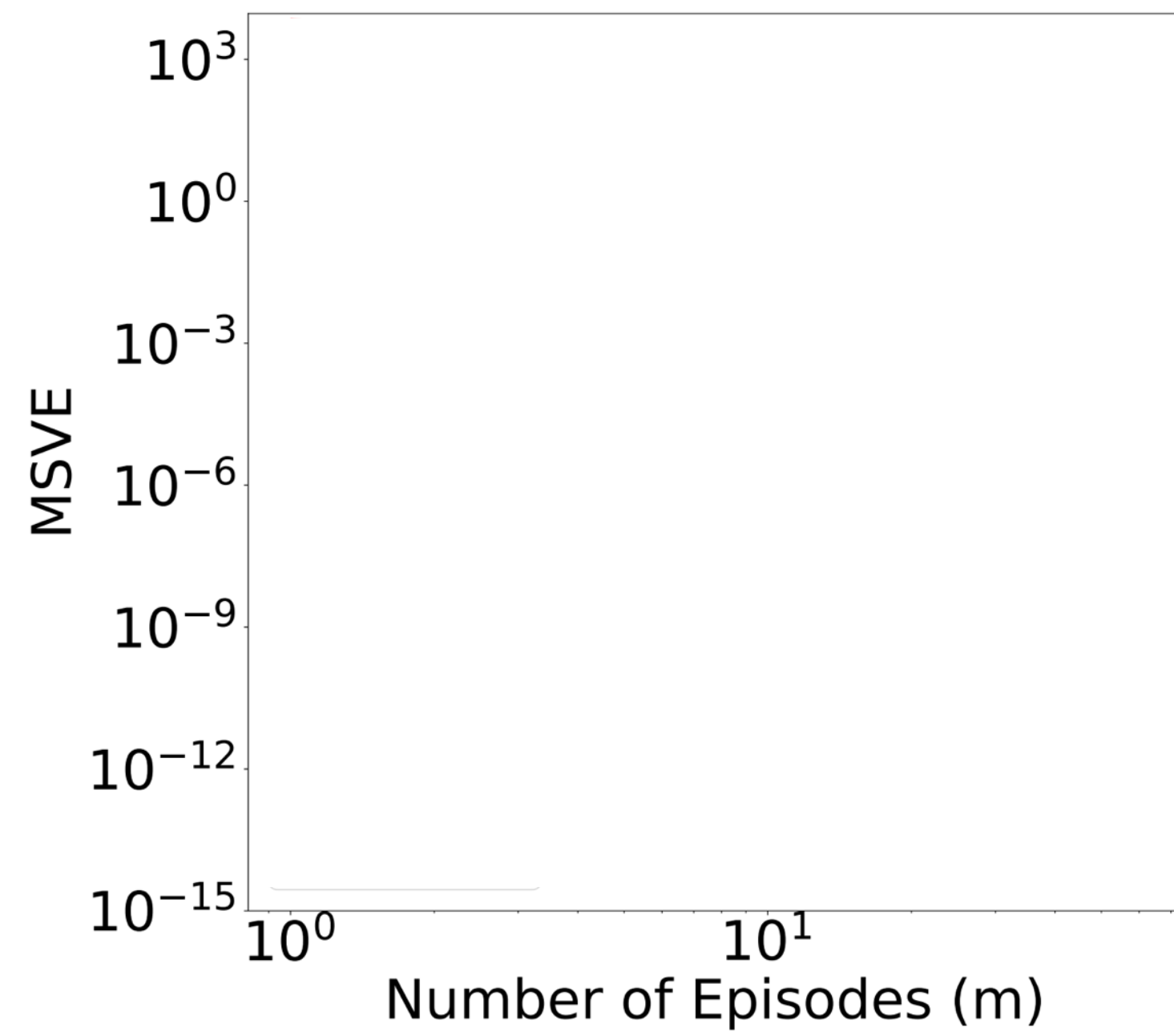
CartPole [Brockman et al., 2016]  
**Continuous** States and  
**Discrete** Actions



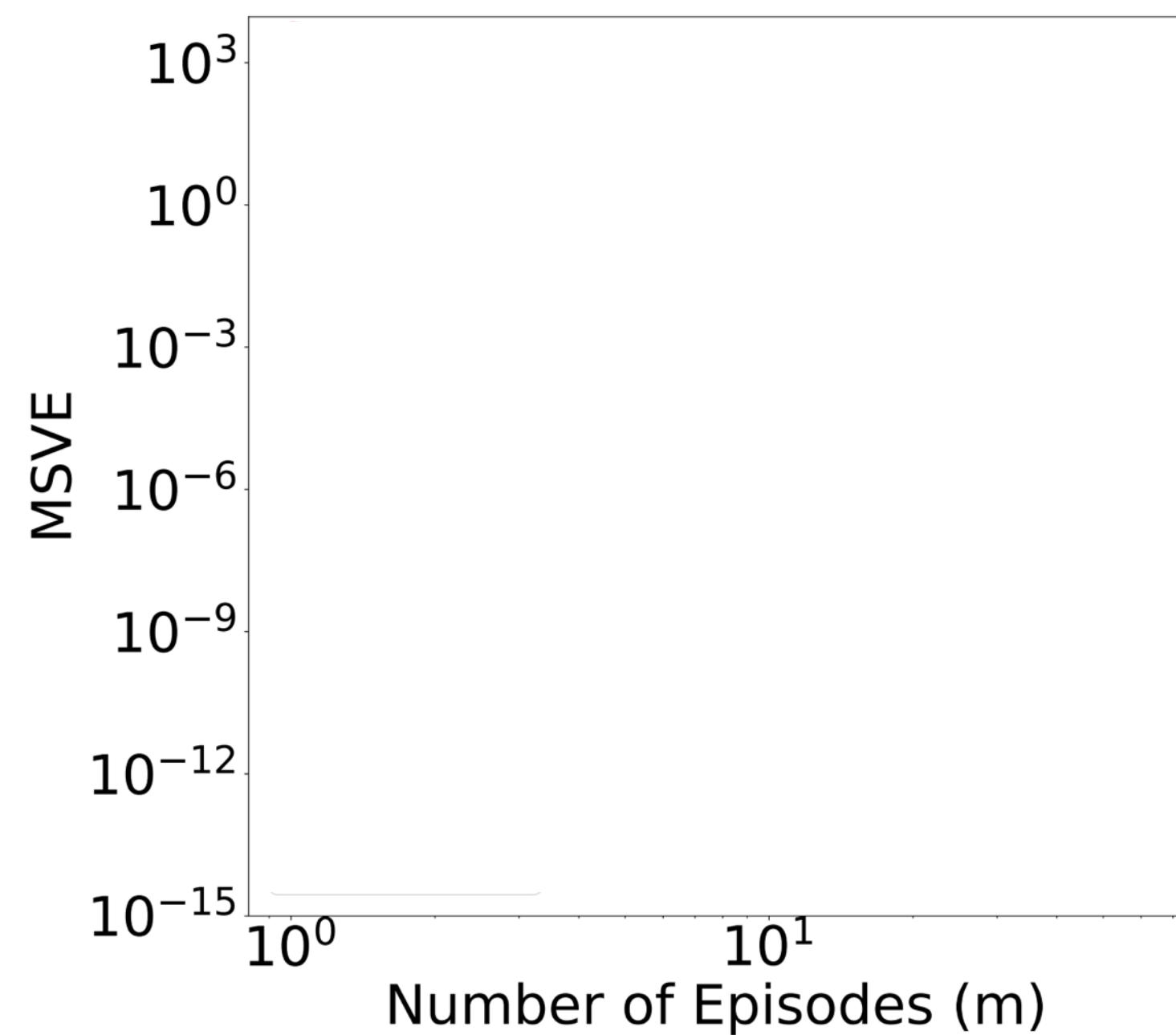
InvertedPendulum  
[Brockman et al., 2016,  
Todorov et al., 2012]  
**Continuous** States and  
Actions

Evaluation Metric  
(weighted error):

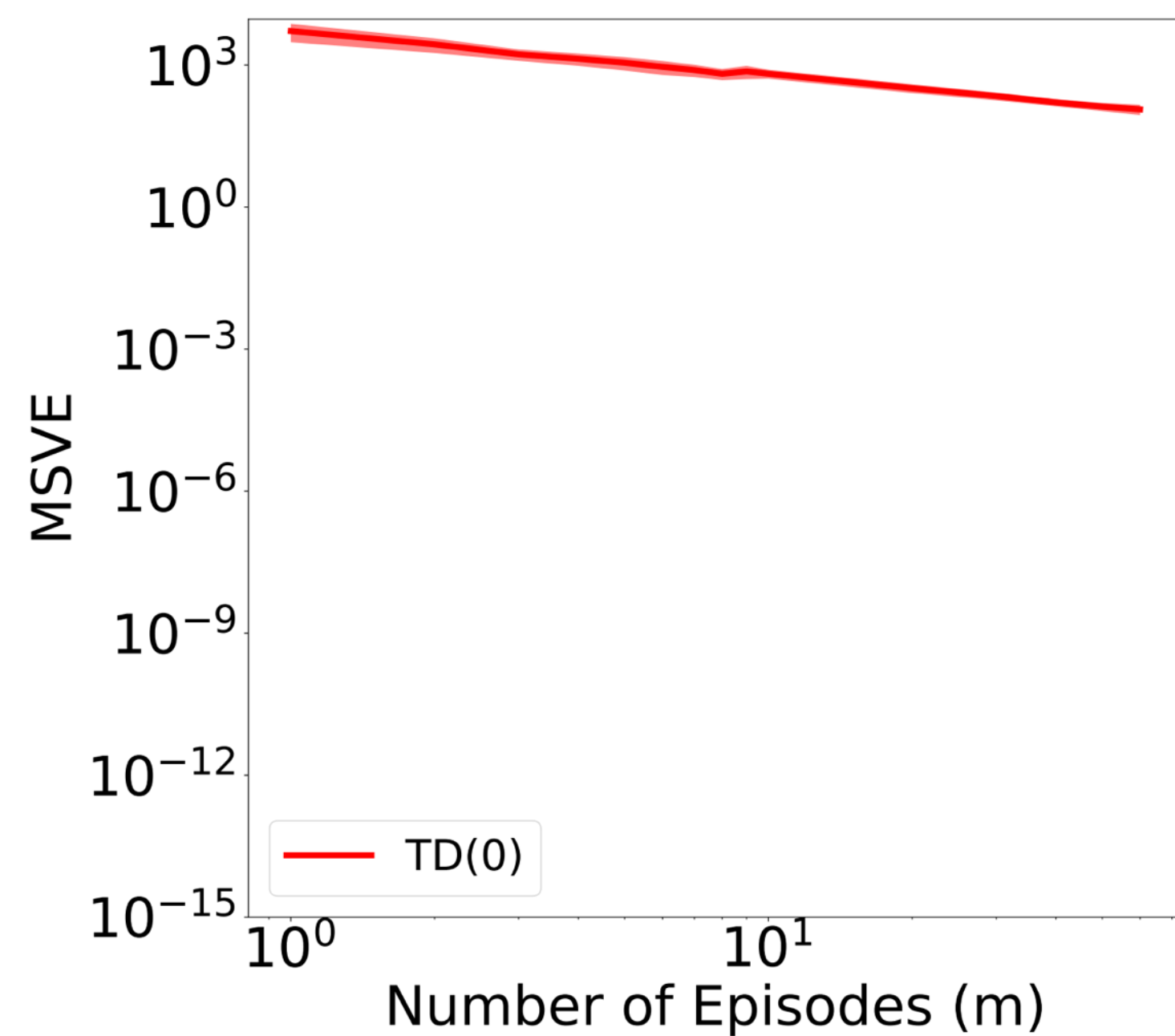
$$\text{MSVE}(\boldsymbol{w}) := \sum_{s \in \mathcal{S}} d_{\pi}(s) \left( v^{\pi}(s) - \hat{v}(s) \right)^2, \forall s \in \mathcal{S}$$



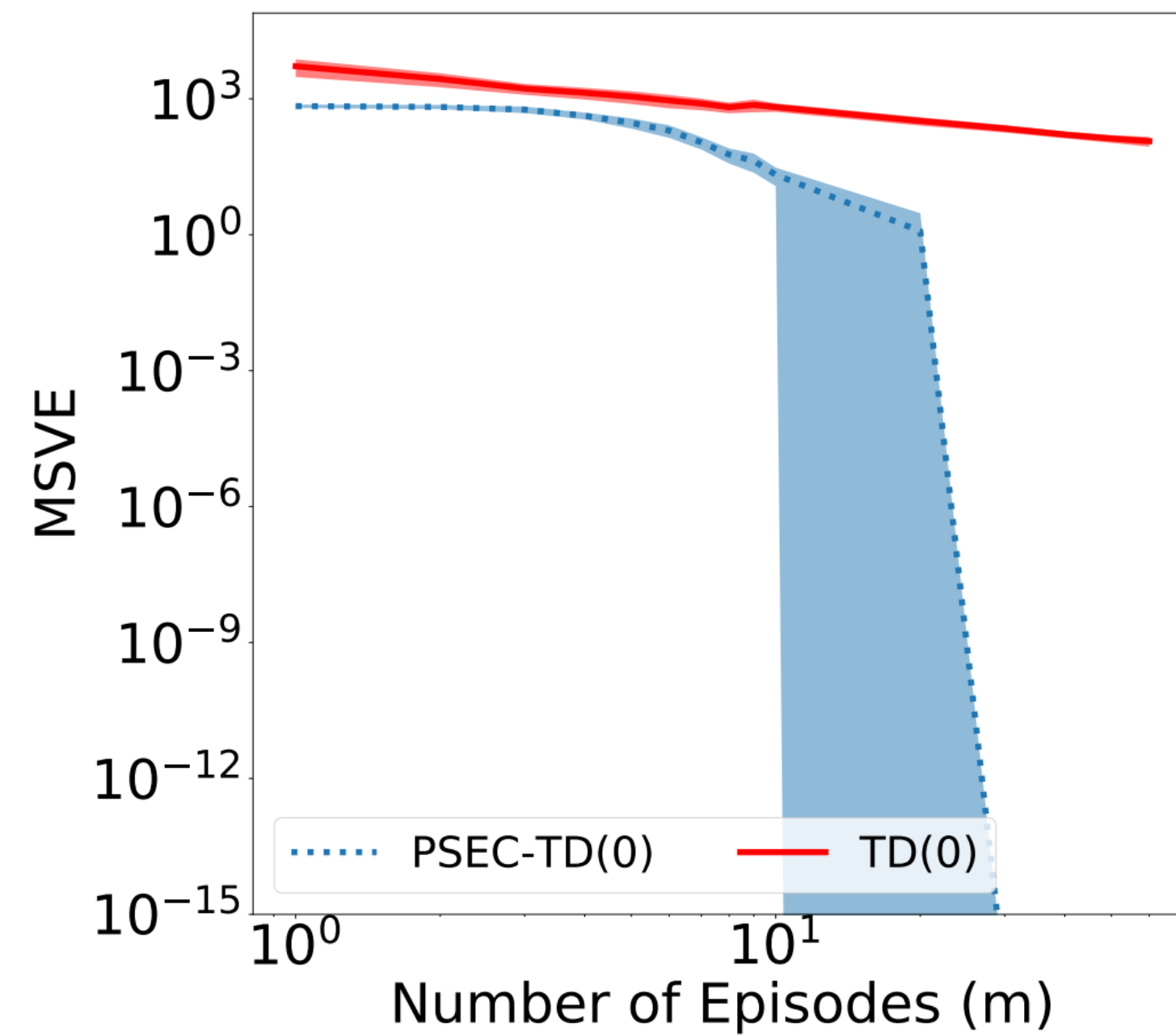
# Empirical Results: Deterministic Gridworld



# Empirical Results: Deterministic Gridworld



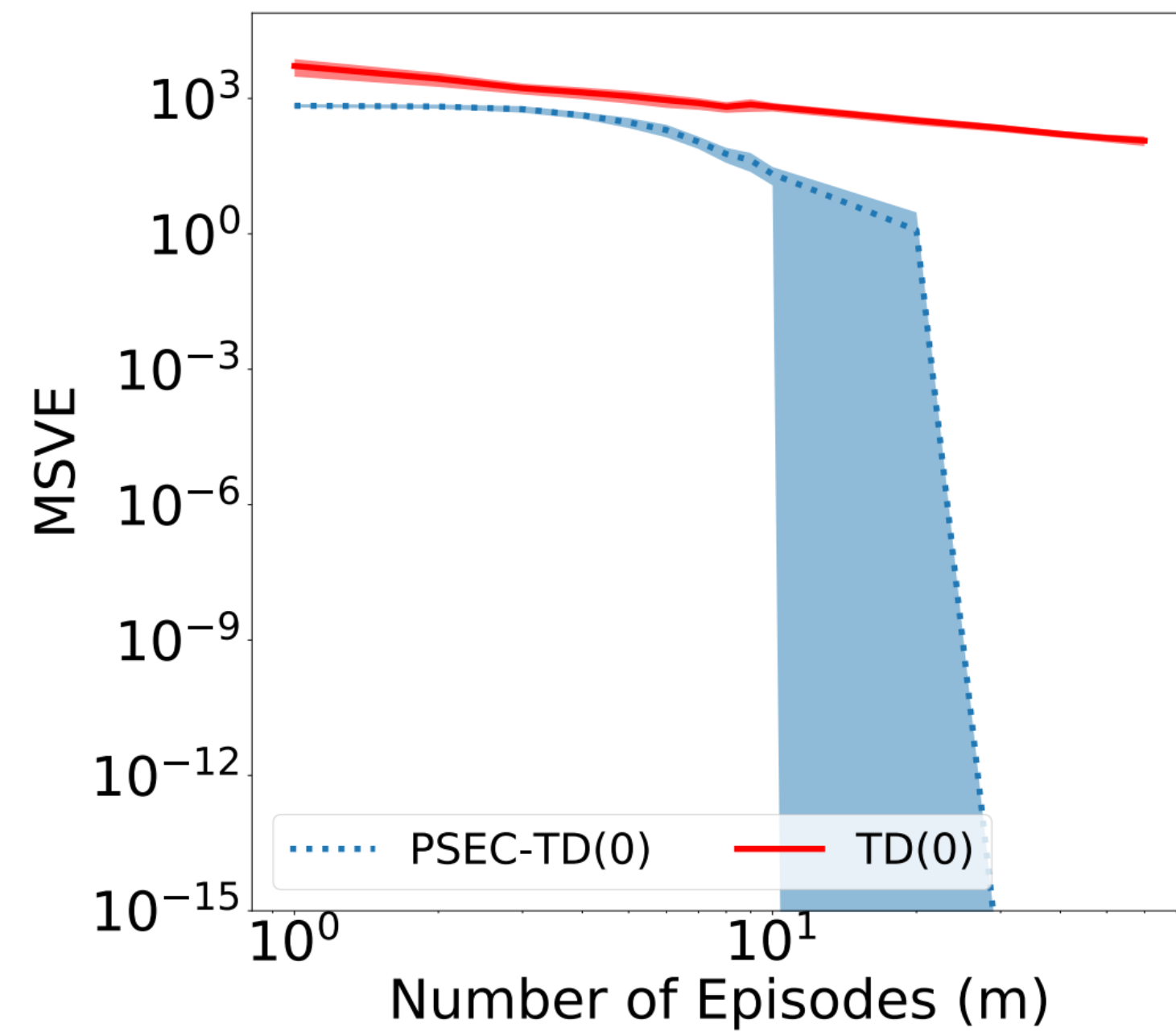
# Empirical Results: Deterministic Gridworld



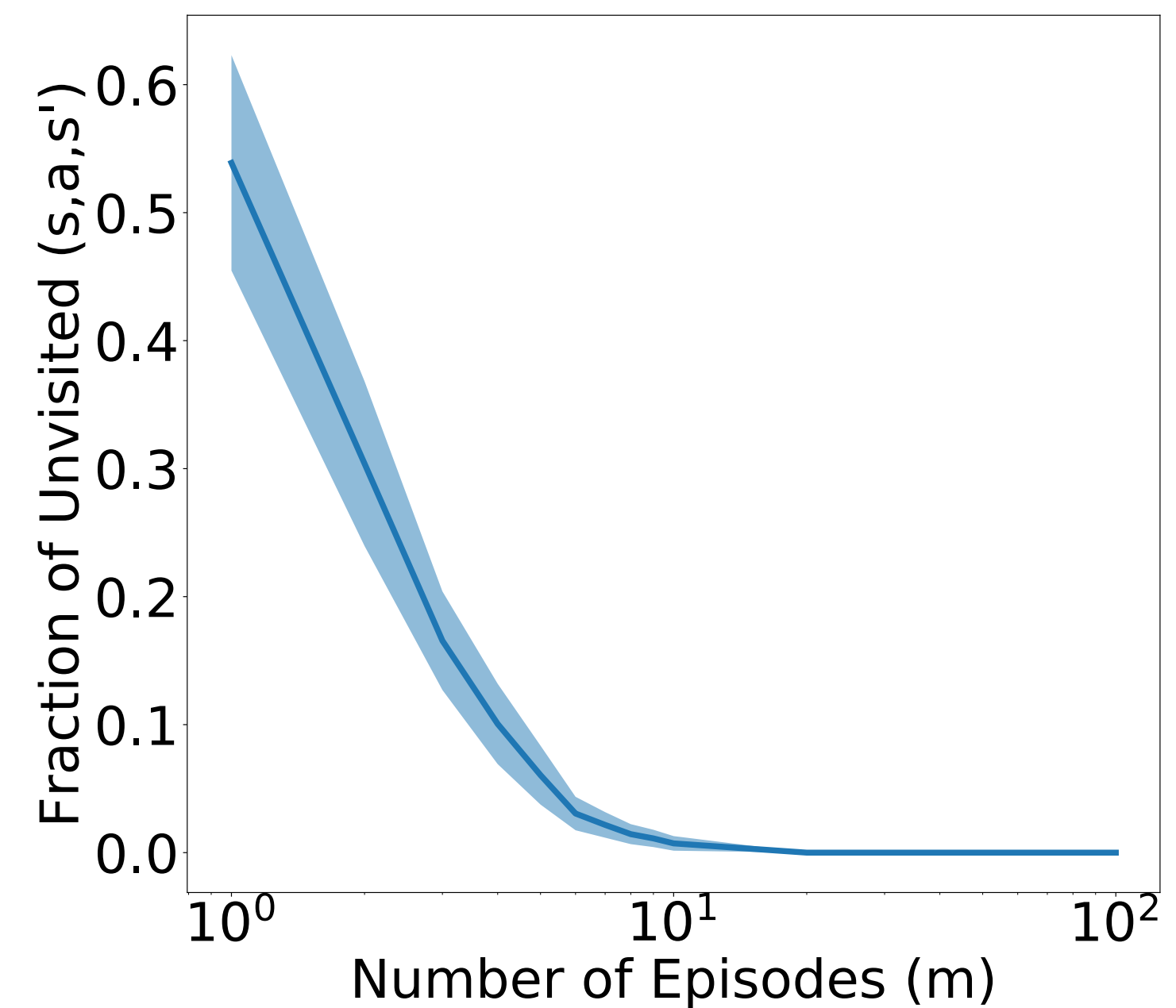
PSEC-TD(0) vs. TD(0)



# Empirical Results: Deterministic Gridworld



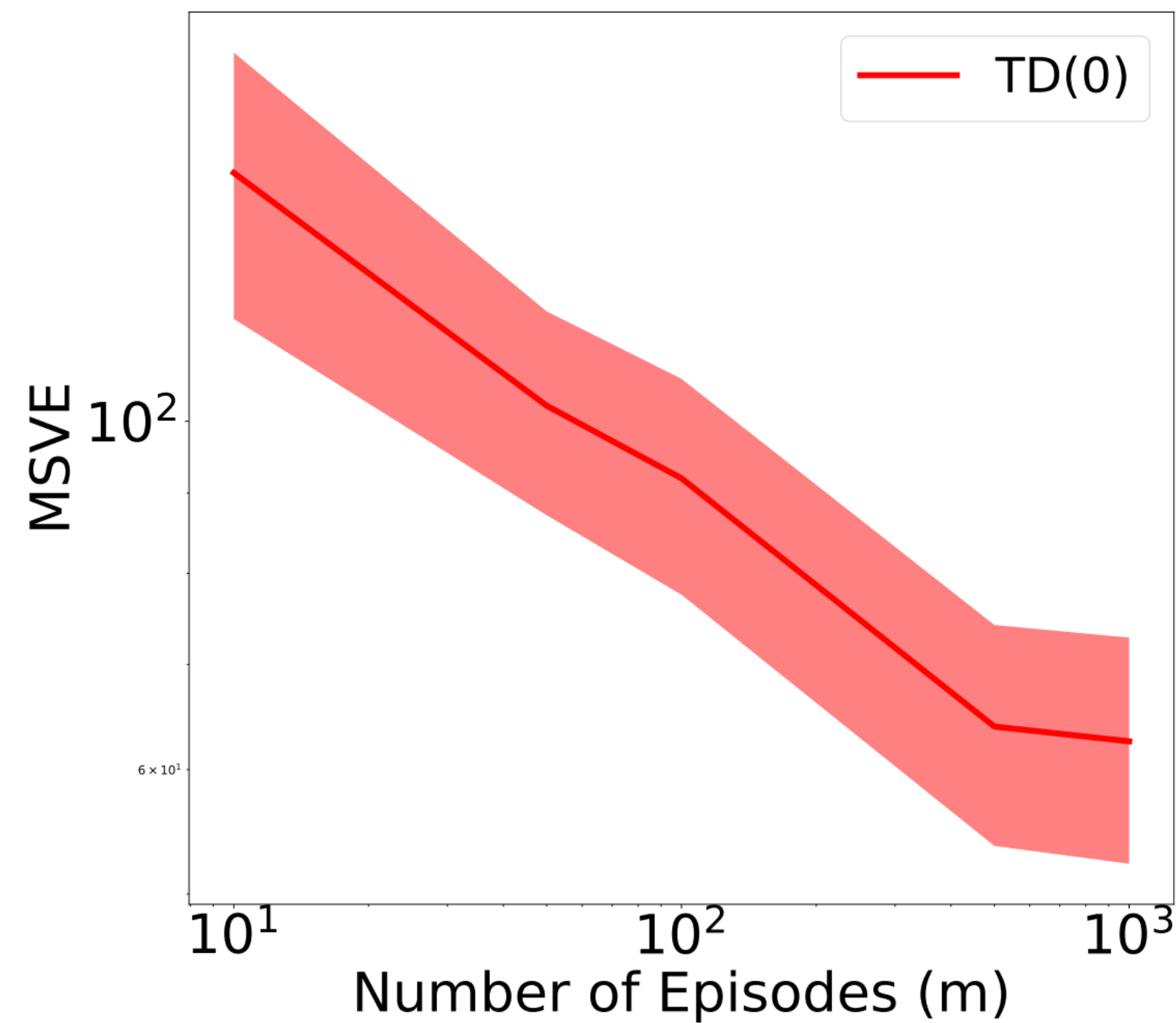
PSEC-TD(0) vs. TD(0)



Unvisited (s,a,s') tuples

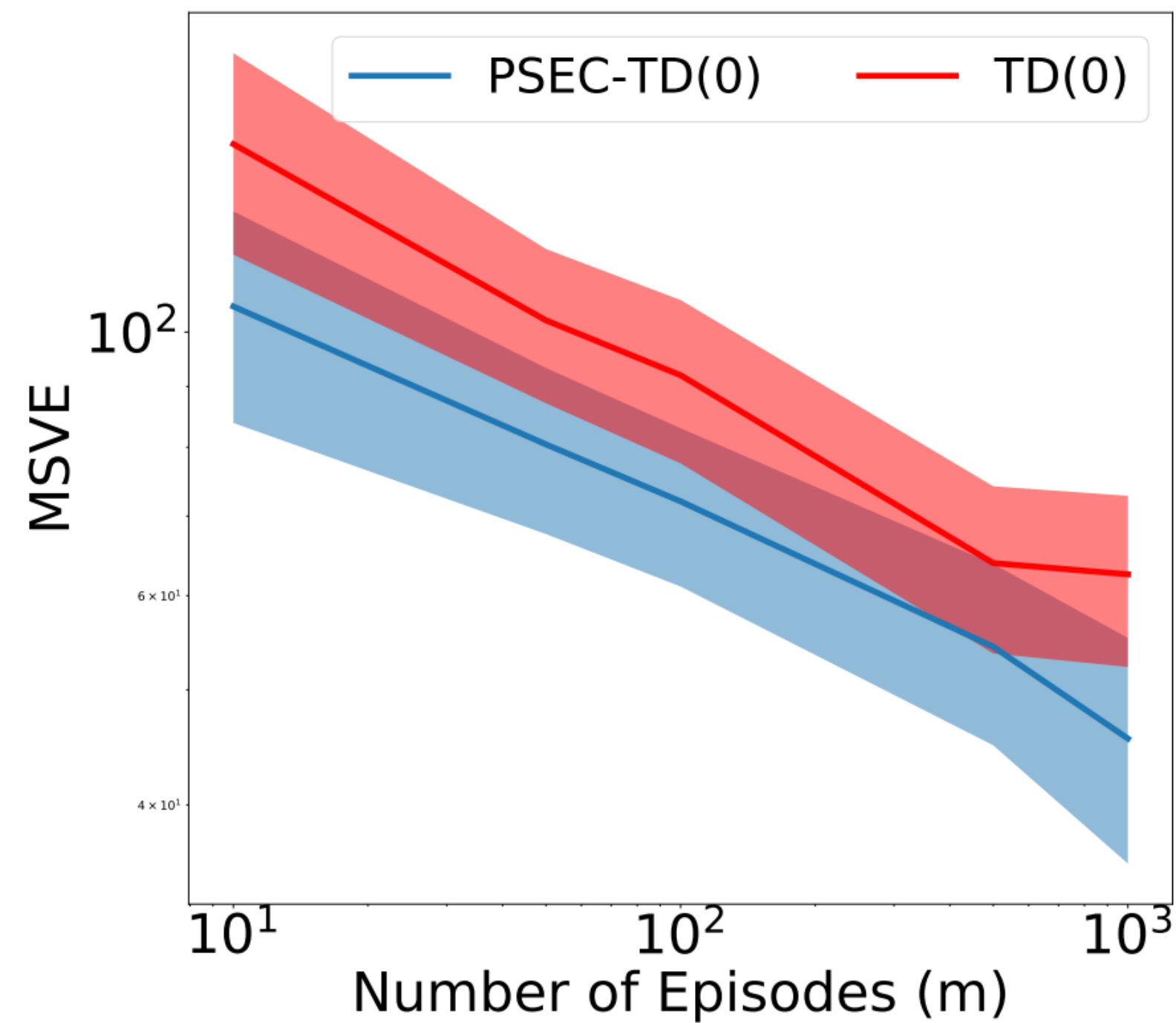
# Empirical Results: Function Approximation

# Empirical Results: Function Approximation



CartPole\*

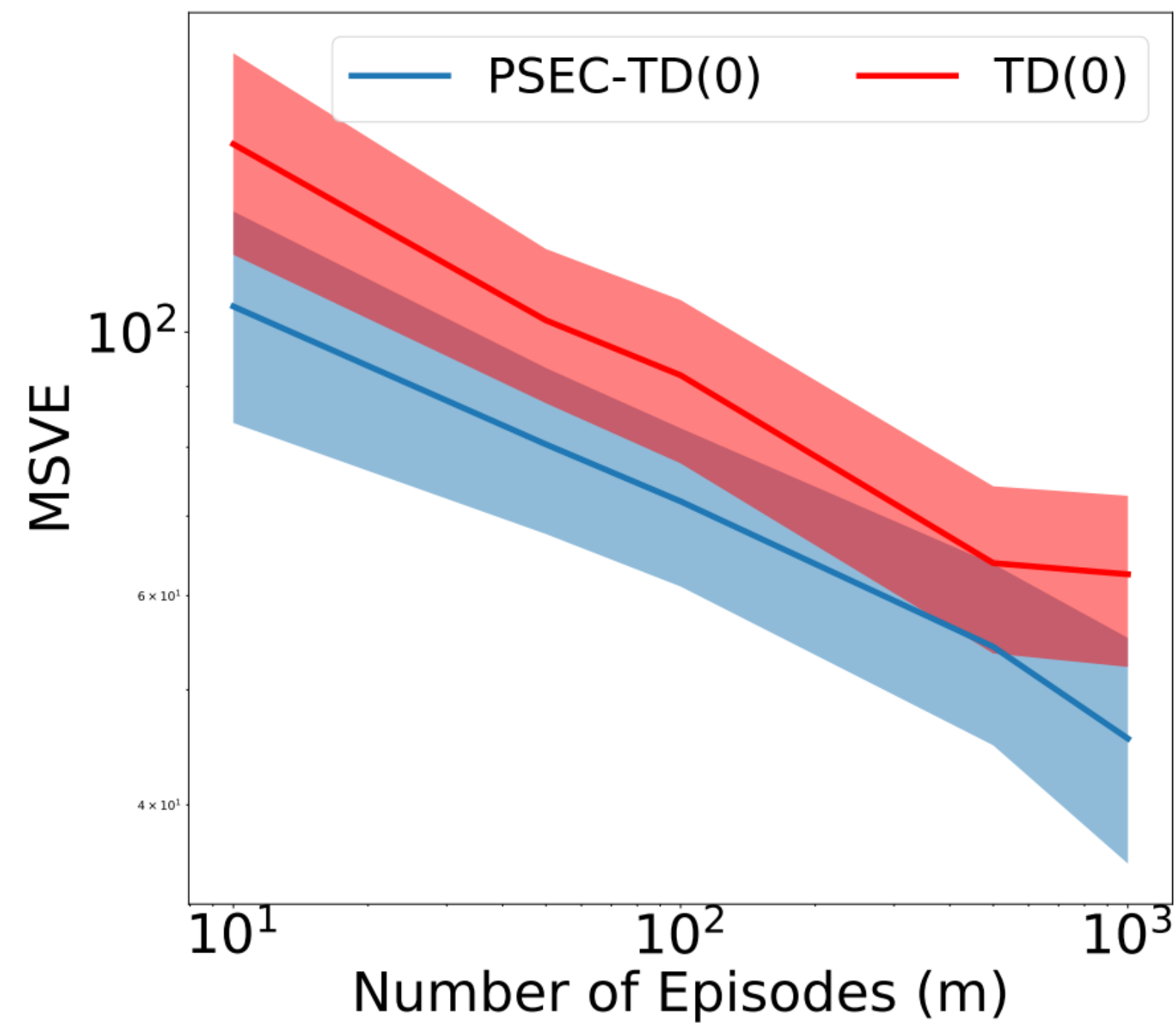
# Empirical Results: Function Approximation



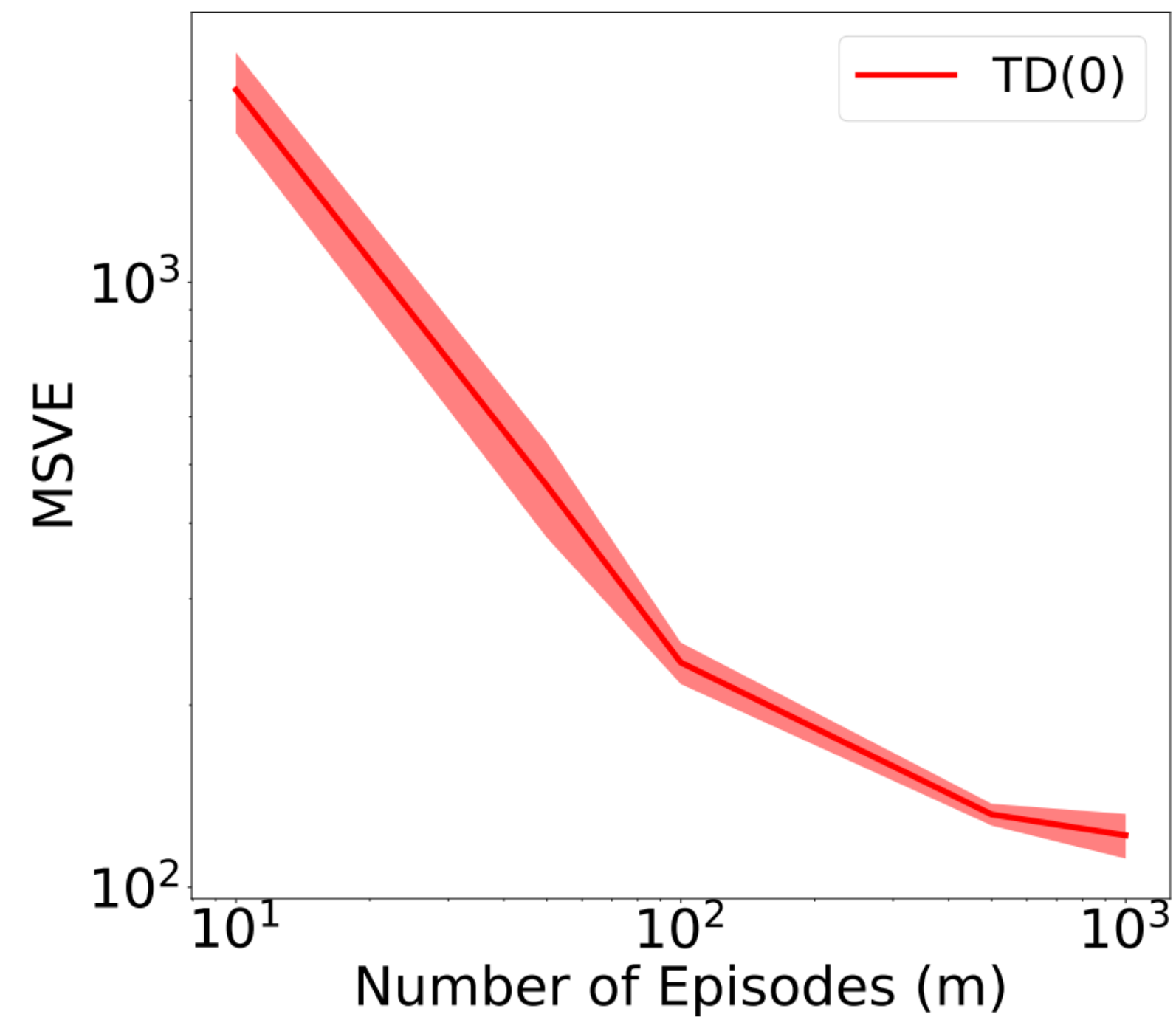
CartPole\*

\* Statistically significant according to Welch's test [Welch 1947]

# Empirical Results: Function Approximation



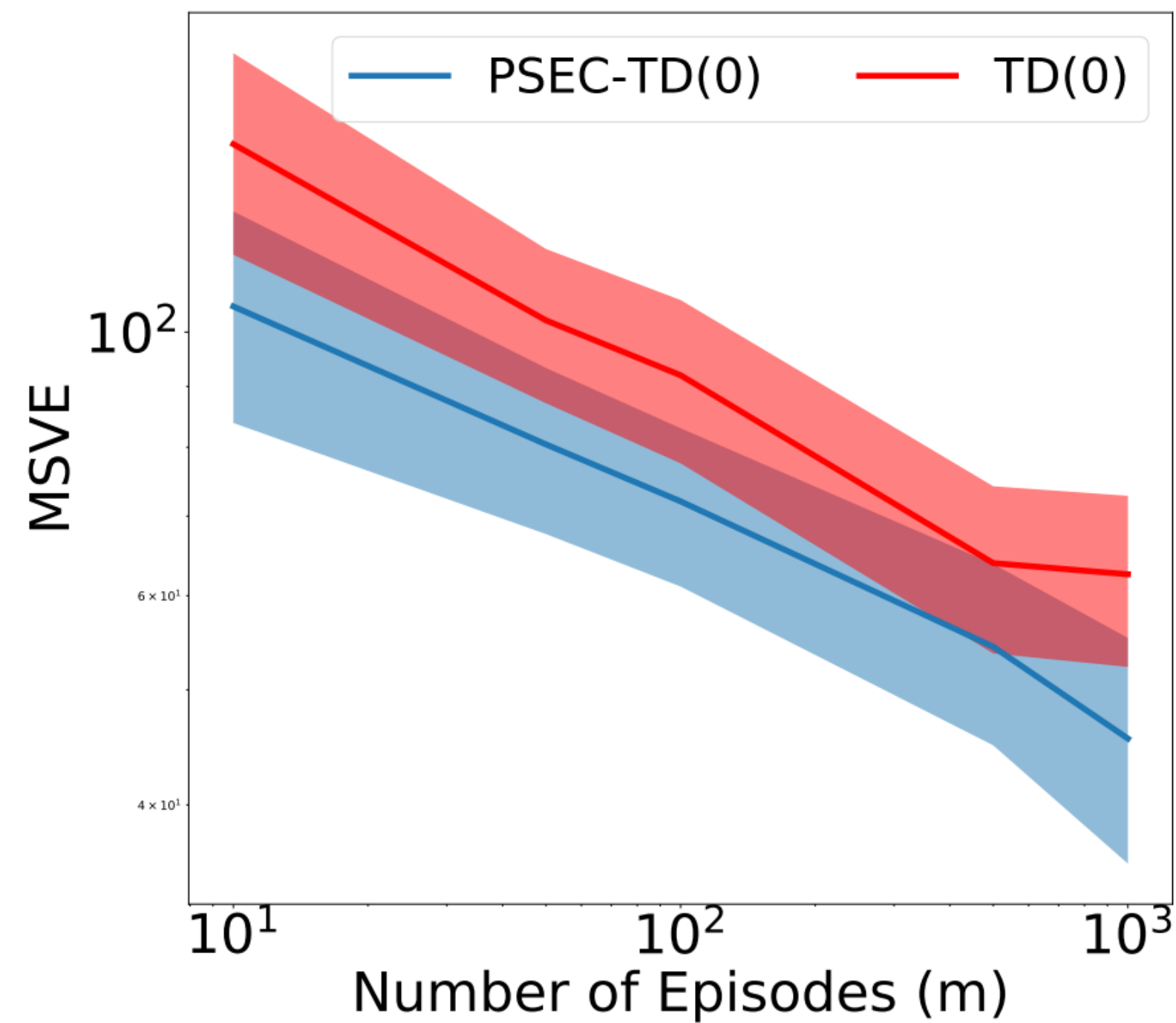
CartPole\*



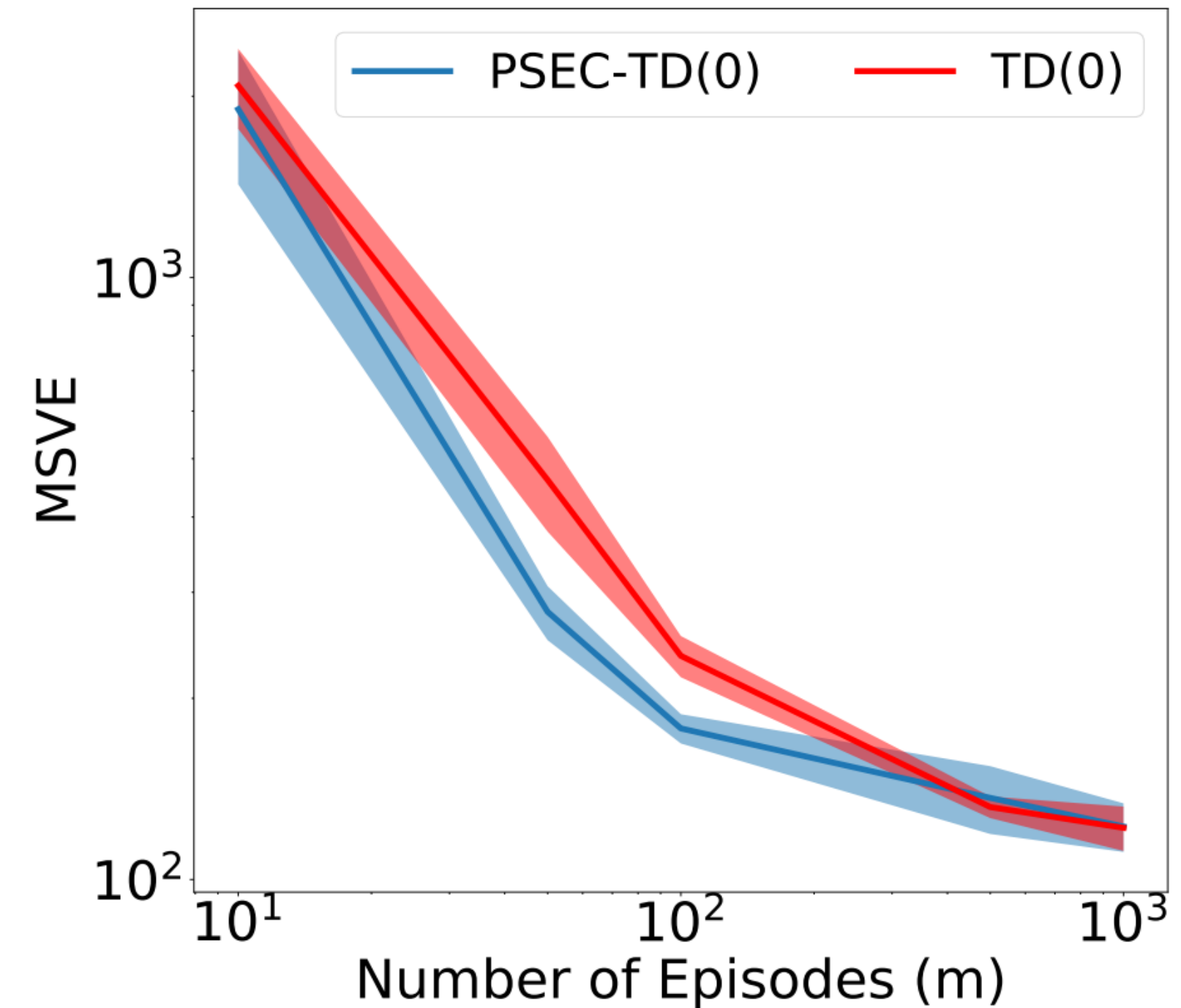
InvertedPendulum

\* Statistically significant according to Welch's test [Welch 1947]

# Empirical Results: Function Approximation



CartPole\*



InvertedPendulum

\* Statistically significant according to Welch's test [Welch 1947]

# Additional Results

# Additional Results

- Extend certainty-equivalence proof by Sutton (1988) from MRP to discounted, per-step reward MDP



# Additional Results

- Extend certainty-equivalence proof by Sutton (1988) from MRP to discounted, per-step reward MDP
- Answer the following (subset of many) questions:

# Additional Results

- Extend certainty-equivalence proof by Sutton (1988) from MRP to discounted, per-step reward MDP
- Answer the following (subset of many) questions:
  - Does expressiveness of the value function impact performance?

# Additional Results

- Extend certainty-equivalence proof by Sutton (1988) from MRP to discounted, per-step reward MDP
- Answer the following (subset of many) questions:
  - Does expressiveness of the value function impact performance?
  - Does expressiveness of the PSEC MLE policy impact performance?

# Additional Results

- Extend certainty-equivalence proof by Sutton (1988) from MRP to discounted, per-step reward MDP
- Answer the following (subset of many) questions:
  - Does expressiveness of the value function impact performance?
  - Does expressiveness of the PSEC MLE policy impact performance?
  - Does underfitting/overfitting the PSEC MLE policy to the batch impact performance?

# Additional Results

- Extend certainty-equivalence proof by Sutton (1988) from MRP to discounted, per-step reward MDP
- Answer the following (subset of many) questions:
  - Does expressiveness of the value function impact performance?
  - Does expressiveness of the PSEC MLE policy impact performance?
  - Does underfitting/overfitting the PSEC MLE policy to the batch impact performance?
  - Can PSEC be applied to off-policy TD(0)?

# Related Work

- Estimating the behavior policy from data [Li et al., 2015, Narita et al., 2018, Hirano et al., 2003].
- Reducing sampling error in policy evaluation [Hanna et al., 2019] and policy gradient learning [Hanna and Stone, 2019].
- Reducing sampling error in action-values [van Seijen et al., 2009, Precup et al., 2000]

# Open Questions

- Reduce sampling error in n-step and TD( $\lambda$ ).
- Evaluate actor-critic algorithms with an improved value function estimate.
- Extend *batch* PSEC to *online* TD(0).

# Takeaways



# Takeaways

- For **finite** batch of data, **batch TD(0)** converges to an **inaccurate value function**.

# Takeaways

- For **finite** batch of data, **batch TD(0)** converges to an **inaccurate value function**.
- Mismatch between the **MLE** policy distribution and **true** policy distribution can be viewed from an **off-policy perspective**.

# Takeaways

- For **finite** batch of data, **batch TD(0)** converges to an **inaccurate value function**.
- Mismatch between the **MLE** policy distribution and **true** policy distribution can be viewed from an **off-policy perspective**.
- PSEC-TD(0) is a **more efficient estimator** than TD(0).

# Takeaways

- For **finite** batch of data, **batch TD(0)** converges to an **inaccurate value function**.
- Mismatch between the **MLE** policy distribution and **true** policy distribution can be viewed from an **off-policy perspective**.
- PSEC-TD(0) is a **more efficient estimator** than TD(0).
- PSEC-TD(0) brings benefit to **discrete/continuous state/action spaces**.

# Takeaways

- For **finite** batch of data, **batch TD(0)** converges to an **inaccurate value function**.
- Mismatch between the **MLE** policy distribution and **true** policy distribution can be viewed from an **off-policy perspective**.
- PSEC-TD(0) is a **more efficient estimator** than TD(0).
- PSEC-TD(0) brings benefit to **discrete/continuous state/action spaces**.
- While primarily shown for **on-policy** TD(0), PSEC is also applicable in **off-policy** TD(0).

# Thank You!



**Brahma S. Pavse**

[brahmasp.github.io](http://brahmasp.github.io)



**Ishan Durugkar**

[idurugkar.github.io](http://idurugkar.github.io)



**Josiah Hanna**

[homepages.inf.ed.ac.uk/  
jhanna2/index.html](http://homepages.inf.ed.ac.uk/jhanna2/index.html)



**Peter Stone**

[cs.utexas.edu/~pstone/](http://cs.utexas.edu/~pstone/)

Recent extension: On Sampling Error in Batch Action-Value Prediction Algorithms