Efficient Algorithms for Linearly Solvable Markov Decision Processes

Bachelor's Thesis

David Pérez Carrasco

Supervisor : Anders Jonsson

Universitat Pompeu Fabra

July 1st, 2024

Motivation

Addressing computational challenges in large state spaces of MDPs.

Motivation

Addressing computational challenges in large state spaces of MDPs.

Leveraging LMDPs for improved performance and scalability.

Motivation

Addressing computational challenges in large state spaces of MDPs.

Leveraging LMDPs for improved performance and scalability.

Necessity for precise embeddings to compare MDPs and LMDPs effectively.

Objectives

01

Benchmark traditional RL against LMDP-based algorithms.

Objectives

01

Benchmark traditional RL against LMDP-based algorithms.

02

Develop robust embeddings for any MDP setting.

Objectives

01

Benchmark traditional RL against LMDP-based algorithms.

02

Develop robust embeddings for any MDP setting.

03

Optimize decision-making with accurate value approximations and efficient exploration strategies.

Markov Decision Processes

$$\mathcal{M} = (\mathcal{S}, \mathcal{S}^{-}, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

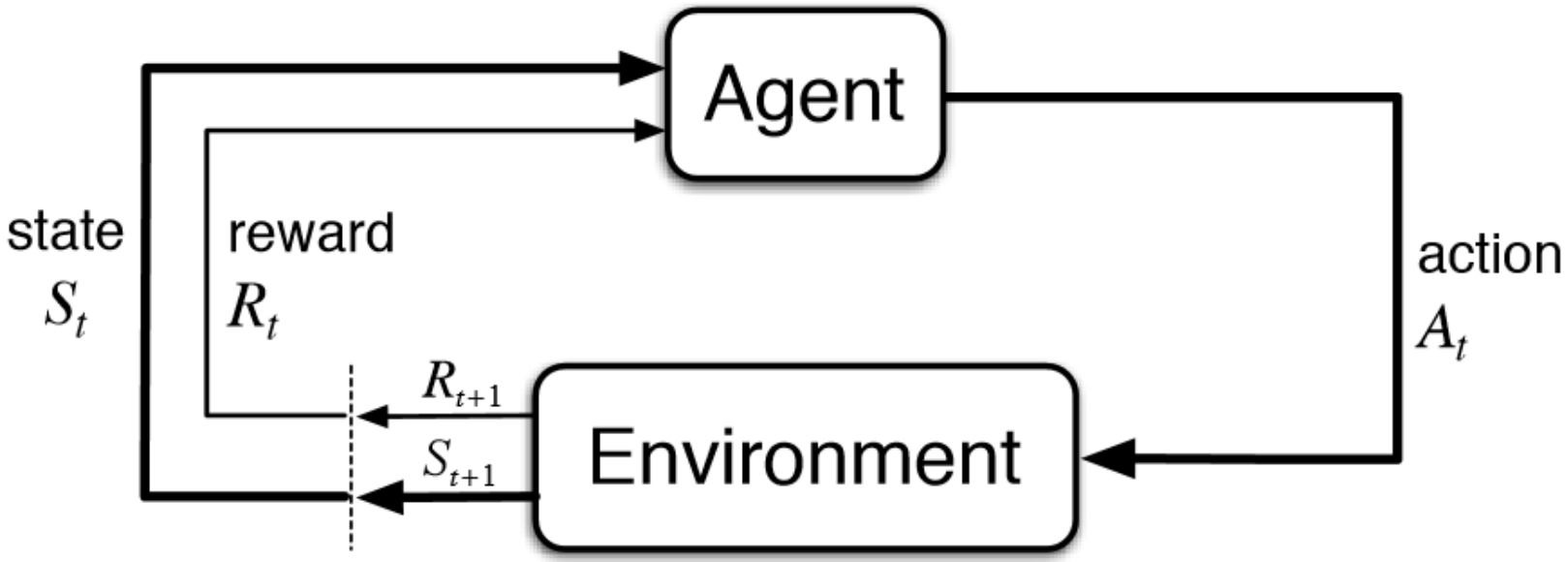

Figure 1: Agent-Environment interaction in MDPs.[1]

[1] Source: Sutton and Barto (2018)

Markov Decision Processes

$$\mathcal{M} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

- $\mathcal{S}$: Set of states.

Markov Decision Processes

$$\mathcal{M} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.

Markov Decision Processes

$$\mathcal{M} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.

Markov Decision Processes

$$\mathcal{M} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.
- $\mathcal{A}$: Set of actions.

Markov Decision Processes

$$\mathcal{M} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.
- $\mathcal{A}$: Set of actions.
- $\mathcal{P} : \mathcal{S}^- \times \mathcal{A} \to \Delta(\mathcal{S})$: State transition probability distribution.

Markov Decision Processes

$$\mathcal{M} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.
- $\mathcal{A}$: Set of actions.
- $\mathcal{P} : \mathcal{S}^- \times \mathcal{A} \to \Delta(\mathcal{S})$: State transition probability distribution.
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: Reward function.

$$\mathcal{R}(s, a) \to (-\infty, 0) \quad \forall s \in \mathcal{S}^- \; \forall a \in \mathcal{A}$$

$$\mathcal{R}(s, a) \to (-\infty, 0] \quad \forall s \in \mathcal{T} \; \forall a \in \mathcal{A}$$

Markov Decision Processes

$$\mathcal{M} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.
- $\mathcal{A}$: Set of actions.
- $\mathcal{P} : \mathcal{S}^- \times \mathcal{A} \to \Delta(\mathcal{S})$: State transition probability distribution.
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: Reward function.
- $\gamma : [0, 1]$: Discount factor.

Linearly-solvable Markov Decision Processes

$$\mathcal{L} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{P}, \mathcal{R}, \lambda)$$

- $\mathcal{S}$: Set of states.

Linearly-solvable Markov Decision Processes

$$\mathcal{L} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{P}, \mathcal{R}, \lambda)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.

Linearly-solvable Markov Decision Processes

$$\mathcal{L} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{P}, \mathcal{R}, \lambda)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.

Linearly-solvable Markov Decision Processes

$$\mathcal{L} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{P}, \mathcal{R}, \lambda)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.
- $\mathcal{P} : \mathcal{S}^- \to \Delta(\mathcal{S})$: Passive Dynamics.

Linearly-solvable Markov Decision Processes

$$\mathcal{L} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{P}, \mathcal{R}, \lambda)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.
- $\mathcal{P} : \mathcal{S}^- \to \Delta(\mathcal{S})$: Passive Dynamics.
- $\mathcal{R} : \mathcal{S} \to \mathbb{R}$: Reward function.

$$\mathcal{R}(s) \to (-\infty, 0) \quad \forall s \in \mathcal{S}^-$$

$$\mathcal{R}(s) \to (-\infty, 0] \quad \forall s \in \mathcal{T}$$

Linearly-solvable Markov Decision Processes

$$\mathcal{L} = (\mathcal{S}, \mathcal{S}^-, \mathcal{T}, \mathcal{P}, \mathcal{R}, \lambda)$$

- $\mathcal{S}$: Set of states.
- $\mathcal{S}^-$: Set of non-terminal states.
- $\mathcal{T}$: Set of terminal states.
- $\mathcal{P} : \mathcal{S}^- \to \Delta(\mathcal{S})$: Passive Dynamics.
- $\mathcal{R} : \mathcal{S} \to \mathbb{R}$: Reward function.
- λ: Temperature Parameter.

Linearly-solvable Markov Decision Processes

LMDP Dynamics

$$\mathbf{u} \in \mathbb{R}^{|\mathcal{S}|}$$

$$\mathcal{P}_{\mathbf{u}}(s'|s) = \mathcal{P}(s'|s)e^{u_{s'}} \tag{1}$$

Constraints

$$\mathcal{P}(s'|s) \neq 0 \implies \mathcal{P}_{\boldsymbol{u}}(s'|s) \neq 0 \quad \forall s \in \mathcal{S}^- \ \forall s' \in \mathcal{S}$$

$$\sum_{s' \in \mathcal{S}} \mathcal{P}_{\boldsymbol{u}}(s'|s) = 1 \quad \forall s \in \mathcal{S}^-$$

Todorov (2006).

Linearly-solvable Markov Decision Processes

Kullback-Leibler divergence

$$
\begin{aligned}
R(s, \mathbf{u}) &= \mathcal{R}(s) - \lambda \cdot KL\left(\mathcal{P}_{\mathbf{u}}(\cdot|s) \middle\| \mathcal{P}(\cdot|s)\right) \\
&= \mathcal{R}(s) - \lambda \cdot \sum_{s' \in S : \mathcal{P}(s'|s) \neq 0} \mathcal{P}_{\mathbf{u}}(s'|s) \log \frac{\mathcal{P}_{\mathbf{u}}(s'|s)}{\mathcal{P}(s'|s)} \ .
\end{aligned} \tag{2}
$$

Jonsson and Gómez (2016).

Linearly-solvable Markov Decision Processes

Bellman Optimality Equation

$$\frac{1}{\lambda}v(s) = \frac{1}{\lambda}\max_{\mathbf{u}\in\mathcal{U}(s)}\left(R(s,\mathbf{u}) + \sum_{s'\in S}\mathcal{P}_{\mathbf{u}}(s'|s)v(s')\right)$$
$$= \frac{1}{\lambda}\mathcal{R}(s) + \max_{\mathbf{u}\in\mathcal{U}(s)}\sum_{s'\in\mathcal{S}}\mathcal{P}_{\mathbf{u}}(s'|s)\left[\frac{1}{\lambda}v(s') - \log\frac{\mathcal{P}_{\mathbf{u}}(s'|s)}{\mathcal{P}(s'|s)}\right]. \quad (3)$$

Jonsson and Gómez (2016).

Linearly-solvable Markov Decision Processes

Definition

Exponential Transformation

$$z(s) = e^{\frac{v(s)}{\lambda}} \quad \forall s \in \mathcal{S},$$

where $v(s) = \mathcal{R}(s) \quad \forall s \in \mathcal{T}$.

$$\mathcal{R}(s) \to (-\infty, 0) \implies z(s) \to (0, 1) \quad \forall s \in \mathcal{S}^-$$

$$\mathcal{R}(s) \to (-\infty, 0] \implies z(s) \to (0, 1] \quad \forall s \in \mathcal{T}$$

Linearly-solvable Markov Decision Processes

Bellman Optimality Equation

$$\begin{aligned}\frac{1}{\lambda}v(s) &= \frac{1}{\lambda}\mathcal{R}(s) + \max_{\mathbf{u}\in\mathcal{U}(s)} \sum_{s'\in\mathcal{S}} \mathcal{P}_{\mathbf{u}}(s'|s) \left[-\log \frac{\mathcal{P}_{\mathbf{u}}(s'|s)}{\mathcal{P}(s'|s)z(s')}\right] \\ &= \frac{1}{\lambda}\mathcal{R}(s) - \min_{\mathbf{u}\in\mathcal{U}(s)} \sum_{s'\in\mathcal{S}} \mathcal{P}_{\mathbf{u}}(s'|s) \left[\log \frac{\mathcal{P}_{\mathbf{u}}(s'|s)}{\mathcal{P}(s'|s)z(s')}\right] .\end{aligned} \tag{4}$$

Jonsson and Gómez (2016).

Linearly-solvable Markov Decision Processes

$$\mathcal{G}\left[z\right](s) = \sum_{s' \in \mathcal{S}} \mathcal{P}(s' \mid s) z(s')$$

Bellman Optimality Equation

$$\begin{aligned}
\frac{1}{\lambda} v(s) &= \frac{1}{\lambda} \mathcal{R}(s) - \min_{\mathbf{u} \in \mathcal{U}(s)} \sum_{s' \in \mathcal{S}} \mathcal{P}_{\mathbf{u}}(s'|s) \left[\log \frac{\mathcal{P}_{\mathbf{u}(s'|s)} \mathcal{G}\left[z\right](s)}{\mathcal{P}(s'|s) z(s') \mathcal{G}\left[z\right](s)} \right] \\
&= \frac{1}{\lambda} \mathcal{R}(s) + \log \mathcal{G}\left[z\right](s) \\
&\quad - \min_{\mathbf{u} \in \mathcal{U}(s)} KL\left(\mathcal{P}_{\mathbf{u}}(\cdot|s) \middle\| \frac{\mathcal{P}(\cdot|s) z(\cdot)}{\mathcal{G}\left[z\right](s)} \right) .
\end{aligned} \tag{5}$$

Jonsson and Gómez (2016).

Linearly-solvable Markov Decision Processes

Optimally-controlled transition probabilities

$$\mathcal{P}_{\mathbf{u}^*}(s'|s) = \frac{\mathcal{P}(s'|s)z(s')}{\sum_{s''\in\mathcal{S}}\mathcal{P}(s''|s)z(s'')} \ . \tag{6}$$

Exponential Optimal Bellman Equation

$$z(s) = e^{\mathcal{R}(s)/\lambda}\sum_{s'\in\mathcal{S}}\mathcal{P}(s'|s)z(s') \ . \tag{7}$$

$$\mathbf{z} = G\mathcal{P}\mathbf{z} \ . \tag{8}$$

Todorov (2006); Jonsson and Gómez (2016).

Power Iteration

1. Initialize:

$$\mathbf{z}^{(0)} = 1 \quad \forall s \in \mathcal{S}^-,\ \mathbf{z}^{(0)} = e^{\mathcal{R}(s)/\lambda} \quad \forall s \in \mathcal{T}$$

2. For $k = 0, 1, 2, \ldots$ until convergence:

$$\mathbf{z}^{(k+1)} = G\mathcal{P}\mathbf{z}^{(k)} \ . \tag{9}$$

3. Upon convergence:

$$\mathbf{z}^{(k)} \approx \mathbf{z} = G\mathcal{P}\mathbf{z}$$

Todorov (2006).

Z-learning

Naive Z-learning

$$\hat{z}(s_t) \leftarrow (1-\alpha)\hat{z}(s_t) + \alpha e^{r_t/\lambda}\hat{z}(s_{t+1}) \ . \tag{10}$$

Z-learning with importance sampling

$$\hat{z}(s_t) = (1-\alpha)\hat{z}(s_t) + \alpha e^{r_t/\lambda}\frac{\mathcal{P}\left(s_{t+1}|s_t\right)}{\hat{\mathcal{P}}_{\mathbf{u}}\left(s_{t+1}|s_t\right)}\hat{z}(s_{t+1}) \ . \tag{11}$$

Todorov (2006); Jonsson and Gómez (2016).

Z-learning

Z-learning with importance sampling

$$\begin{aligned}\hat{z}(s_t) &\leftarrow (1-\alpha)\,\hat{z}(s_t) + \alpha e^{r_t/\lambda}\frac{\mathcal{P}\left(s_{t+1}|s_t\right)}{\hat{\mathcal{P}}_{\mathbf{u}}\left(s_{t+1}|s_t\right)}\hat{z}\left(s_{t+1}\right)\\ &= (1-\alpha)\,\hat{z}(s_t) + \alpha e^{r_t/\lambda}\frac{\mathcal{P}\left(s_{t+1}|s_t\right)\hat{z}(s_{t+1})}{\mathcal{P}\left(s_{t+1}|s_t\right)\hat{z}(s_{t+1})}\sum_{s'\in S}\mathcal{P}\left(s'|s_t\right)\hat{z}(s')\\ &= (1-\alpha)\,\hat{z}(s_t) + \alpha e^{r_t/\lambda}\sum_{s'\in S}\mathcal{P}\left(s'|s_t\right)\hat{z}(s')\ .\end{aligned} \tag{12}$$

Jonsson and Gómez (2016).

Embedding of stochastic MDP

Definition

MDP Embedding

$$\mathcal{P}_{\mathbf{u}^a}(\cdot|s) = \tilde{\mathcal{P}}(\cdot|s,a),\ \mathcal{R}_{\mathbf{u}^a}(s) = \tilde{\mathcal{R}}(s,a) \quad \forall (s,a) \in (\mathcal{S}^-, \mathcal{A})$$

where $\mathbf{u}^a$ is the control vector corresponding to action a.

$$\mathcal{R}(s) - \sum_{s' \in \mathcal{S}} \tilde{\mathcal{P}}(s'|s,a) \log \frac{\tilde{\mathcal{P}}(s'|s,a)}{\mathcal{P}(s'|s)} = \tilde{\mathcal{R}}(s,a) \quad \forall s \in \mathcal{S}^-, a \in \mathcal{A}. \tag{13}$$

Todorov (2006); Jonsson and Gómez (2016).

Embedding of stochastic MDP

System of $|\mathcal{A}|$ equations for a fixed s

$$\boxed{\begin{aligned} m_{s'} &= \log \mathcal{P}(s'|s) \\ b_a &= \tilde{\mathcal{R}}(s,a) + \sum_{s' \in \mathcal{S}} \tilde{\mathcal{P}}(s'|s,a) \log \tilde{\mathcal{P}}(s'|s,a) \\ D_{as'} &= \tilde{\mathcal{P}}(s'|s,a) \end{aligned}} \tag{14}$$

$$\mathcal{R}\mathbf{1} + D\mathbf{m} = \mathbf{b}. \tag{15}$$

Todorov (2009).

Embedding of stochastic MDP

$$D\mathbf{1} = \mathbf{1}$$

Linear Problem

$$D\left(\mathcal{R}\mathbf{1} + \mathbf{m}\right) = \mathbf{b}. \tag{16}$$

$$\mathbf{c} = \mathcal{R}\mathbf{1} + \mathbf{m}$$

Todorov (2009).

Embedding of Stochastic MDP

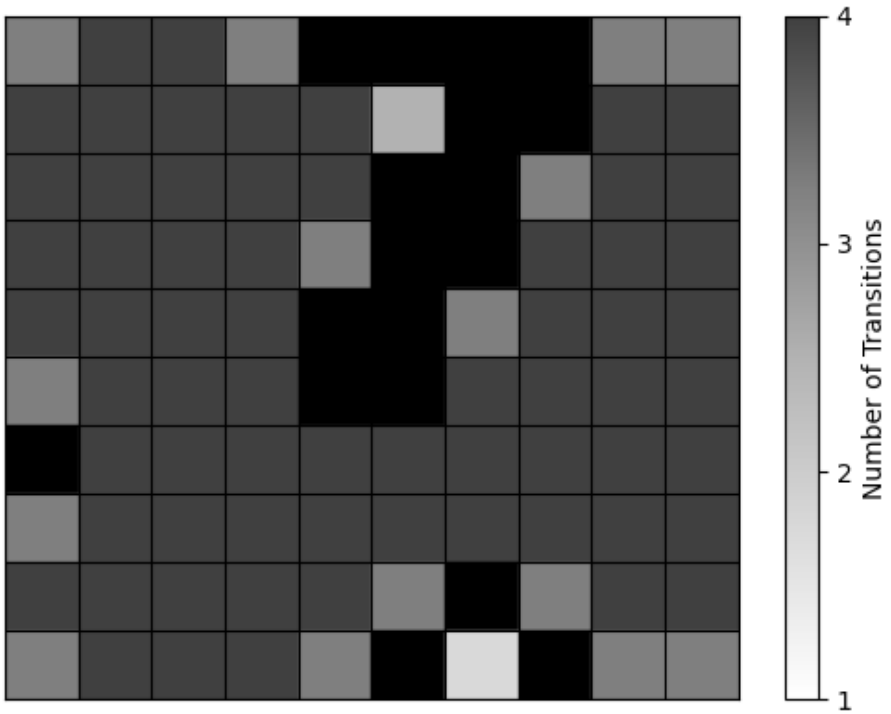

Figure 2: State transitions distribution in a grid world example.

Embedding of Stochastic MDP

When $|\mathcal{A}| \neq N(s)$:

1. Column-rank Deficient:

$$|\mathcal{A}| > N(s)$$

2. Row-Rank Deficient:

$$N(s) > |\mathcal{A}|$$

Todorov (2009).

Embedding of Stochastic MDP

Least Squares Solver

$$\mathbf{c} = \arg\min_{\mathbf{c}} \|D\mathbf{c} - \mathbf{b}\|^2$$

Moore-Penrose Pseudoinverse

$$\mathbf{c} = D^{\dagger}\mathbf{b}$$

Embedding of Stochastic MDP

Linear Problem Solution

$$\mathbf{m} = \mathbf{c} - \mathcal{R}\mathbf{1} \quad \forall \mathcal{R} \tag{17}$$

Constraint

$$\sum_{s' \in \mathcal{S}} e^{m_{s'}} = 1$$

$$\mathcal{R} = \log \sum_{s' \in \mathcal{S}} e^{-c_{s'}} \tag{18}$$

Todorov (2009).

Embedding of Stochastic MDP

Terminal States Embedding

$$\mathcal{R}(s) = \frac{\sum_{a' \in \mathcal{A}} \tilde{\mathcal{R}}(s, a)}{|\mathcal{A}|} \tag{19}$$

Embedding of Stochastic MDP

Algorithm 1 Vectorized Embedding of stochastic MDP into LMDP

input: MDP $\mathcal{M}$ with $\tilde{\mathcal{R}}$ and stochastic $\tilde{\mathcal{P}}$, small ϵ
output: LMDP $\mathcal{L}$ with $\mathcal{R}$ and $\mathcal{P}$
initialize $D = \mathcal{P}$
Identify the number of potential next states $N(s)$ from each state $s \in \mathcal{S}^-$
Obtain the set $\mathcal{N}$ by collecting the unique values of $N(s)$ for all states $s \in \mathcal{S}^-$
for $n \in \mathcal{N}$ **do**
    Identify state space $\mathcal{S}^n$ of states s with $N(s) = n$
    Obtain D' for $\mathcal{S}^n$
    Remove zero columns from D', keeping only possible transitions
    Replace zeros in remaining columns with ϵ and renormalize
    $\mathbf{B} \leftarrow \mathcal{R}[\mathcal{S}^n] + \sum_{s' \in \mathcal{S}} D' \log(D')$
    Calculate pseudo-inverse $D^\dagger$ of D
    $\mathbf{C} \leftarrow D^\dagger \mathbf{B}$
    $\mathbf{R} \leftarrow \log(\sum_{s' \in \mathcal{S}} e^{-\mathbf{C}})$
    $\mathbf{M} \leftarrow \mathbf{C} - \mathbf{R}$
    $\mathcal{R}[\mathcal{S}^n] \leftarrow \mathbf{R}$
    $\mathcal{P}[\mathcal{S}^n] \leftarrow e^{\mathbf{M}}$
end for
$\mathcal{R}(\mathcal{T}) \leftarrow \left(\sum_{a \in \mathcal{A}} \tilde{\mathcal{R}}(\mathcal{T}, a)\right) / |\mathcal{A}|$
return $\mathcal{L}$ with $\mathcal{R}$ and $\mathcal{P}$

Embedding of Deterministic MDP

Todorov's Approach

$$\begin{aligned} m_{s'} &= \log \mathcal{P}(s'|s) \\ b_a &= \tilde{\mathcal{R}}(s,a) + \sum_{s' \in \mathcal{S}} \tilde{\mathcal{P}}(s'|s,a) \log \tilde{\mathcal{P}}(s'|s,a) \\ D_{as'} &= \tilde{\mathcal{P}}(s'|s,a) \end{aligned} \tag{14}$$

- $\mathcal{R}(s) \approx \frac{\sum_{a \in \mathcal{A}} \tilde{\mathcal{R}}(s,a)}{|\mathcal{A}|} + \log N(s) \quad \forall s \in \mathcal{S}$
- $\log N(s) > \left| \frac{\sum_{a \in \mathcal{A}} \tilde{\mathcal{R}}(s,a)}{|\mathcal{A}|} \right| \implies \mathcal{R}(s) > 0$

Todorov (2006).

Embedding of Deterministic MDP

Stochastic Policy Averaging

$$\mathcal{R}(s) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{R}}(s, a), \text{ for all } s \in \mathcal{S} \tag{20}$$

$$\mathcal{P}(s'|s) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{P}}(s'|s, a), \text{ for all } s \in \mathcal{S}^-, s' \in \mathcal{S} \tag{21}$$

Embedding of Deterministic MDP

Kullback-Leibler divergence

$$R(s, \mathbf{u}) = \mathcal{R}(s) - \lambda \cdot KL\left(\mathcal{P}_{\mathbf{u}}(\cdot|s) \middle\| \mathcal{P}(\cdot|s)\right) \quad . \tag{2}$$

$$\mathcal{R}(s) = \tilde{\mathcal{R}}(s, a) + \lambda KL\left(\mathcal{P}_{\mathbf{u}} \middle\| \mathcal{P}\right) \quad . \tag{22}$$

Jonsson and Gómez (2016).

Embedding of Deterministic MDP

Algorithm 2 Deterministic MDP Embedding through Stochastic Policy Averaging and KL divergence

1: **input:** MDP $\mathcal{M}$ with $\mathcal{A}$, $\mathcal{S}$, $\mathcal{T}$, $\mathcal{S}^-$, $\tilde{\mathcal{R}}$ and deterministic $\tilde{\mathcal{P}}$, temperature parameter λ
2: **output:** LMDP $\mathcal{L}$ with $\hat{\mathcal{R}}$ and $\mathcal{P}$
3: $\hat{\mathcal{R}}(\cdot) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{R}}(\cdot, a)$
4: $\mathcal{P}(\cdot|\cdot) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{P}}(\cdot|\cdot, a)$
5: Obtain Z from power iteration for LMDP $\mathcal{L}$ with $\mathcal{P}$ and $\hat{\mathcal{R}}$
6: $\mathcal{P}_{\mathbf{u}^*} \leftarrow \mathcal{P}Z / \sum_{s' \in \mathcal{S}} \mathcal{P}(s'|\cdot)Z(s')$
7: $\hat{\mathcal{R}} \leftarrow \hat{\mathcal{R}} + \lambda KL(\mathcal{P}_{\mathbf{u}^*} \| \mathcal{P})$
8: **return** $\mathcal{L}$ with $\hat{\mathcal{R}}$ and $\mathcal{P}$

Embedding of Deterministic MDP

- **Reward Approximation:** $\exists K$ such as:

$$\mathcal{R}(s) = K \cdot \hat{\mathcal{R}}(s) \ \forall s \in \mathcal{S}. \tag{23}$$

- **Objective function:** MSE between v^{π} and $v = \lambda \log z$.

$$g(K) = \sum_{s \in \mathcal{S}} \left(v_K(s) - v^*(s) \right)^2 ,$$

where $v_K(s) = \lambda \log z(s)$ for LMDP with $\mathcal{R}(s) = K \cdot \hat{\mathcal{R}}(s) \ \forall s$.

- **Global Minima:** Find K that minimizes $g(K)$ through search.

$$K = \arg\min_K g(K) \ .$$

Embedding of Deterministic MDP

Algorithm 3 Deterministic MDP Embedding through ternary search

input: MDP $\mathcal{M}$ with $\mathcal{A}$, $\mathcal{S}$, $\mathcal{T}$, $\mathcal{S}^-$, $\tilde{\mathcal{R}}$ and deterministic $\tilde{\mathcal{P}}$, temperature parameter λ, small ϵ, optimal value function V^* from $\mathcal{M}$
output: LMDP $\mathcal{L}$ with $\mathcal{R}$ and $\mathcal{P}$
$\hat{\mathcal{R}}(\cdot) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{R}}(\cdot, a)$
$\mathcal{P}(\cdot|\cdot) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{P}}(\cdot|\cdot, a)$
Obtain Z from power iteration method for LMDP $\mathcal{L}$ with $\mathcal{P}$ and $\hat{\mathcal{R}}$
$\mathcal{P}_{\mathbf{u}^*} \leftarrow \mathcal{P}Z / \sum_{s' \in \mathcal{S}} \mathcal{P}(s'|\cdot)Z(s')$
$\hat{\mathcal{R}} \leftarrow \hat{\mathcal{R}} + \lambda KL(\mathcal{P} \| \mathcal{P}_{\mathbf{u}^*})$
Set $K_{min} = 0$ and $K_{max} = 1$
while $K_{max} - K_{min} \geq \epsilon$ **do**
    $m_1 = K_{min} + (K_{max} - K_{min})/3$
    $R_1 = m_1 \cdot \hat{\mathcal{R}}$
    Obtain Z_1 from power iteration method for LMDP $\mathcal{L}$ with $\mathcal{P}$ and R_1
    $V_1 = \lambda \log Z_1$
    $MSE_1 = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} (V_1 - V^*)^2$
    $m_2 = K_{max} - (K_{max} - K_{min})/3$
    $R_2 = m_2 \cdot \hat{\mathcal{R}}$
    Obtain Z_2 from power iteration method for LMDP $\mathcal{L}$ with $\mathcal{P}$ and R_2
    $V_2 = \lambda \log Z_2$
    $MSE_2 = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} (V_2 - V^*)^2$
    if $MSE_1 > MSE_2$ **then**
        $K_{min} = m_1$
    else
        $K_{max} = m_2$
    end if
end while
$\mathcal{R} = K_{min} * \hat{\mathcal{R}}$
return $\mathcal{L}$ with $\mathcal{P}$ and $\mathcal{R}$

Embedding of Deterministic MDP

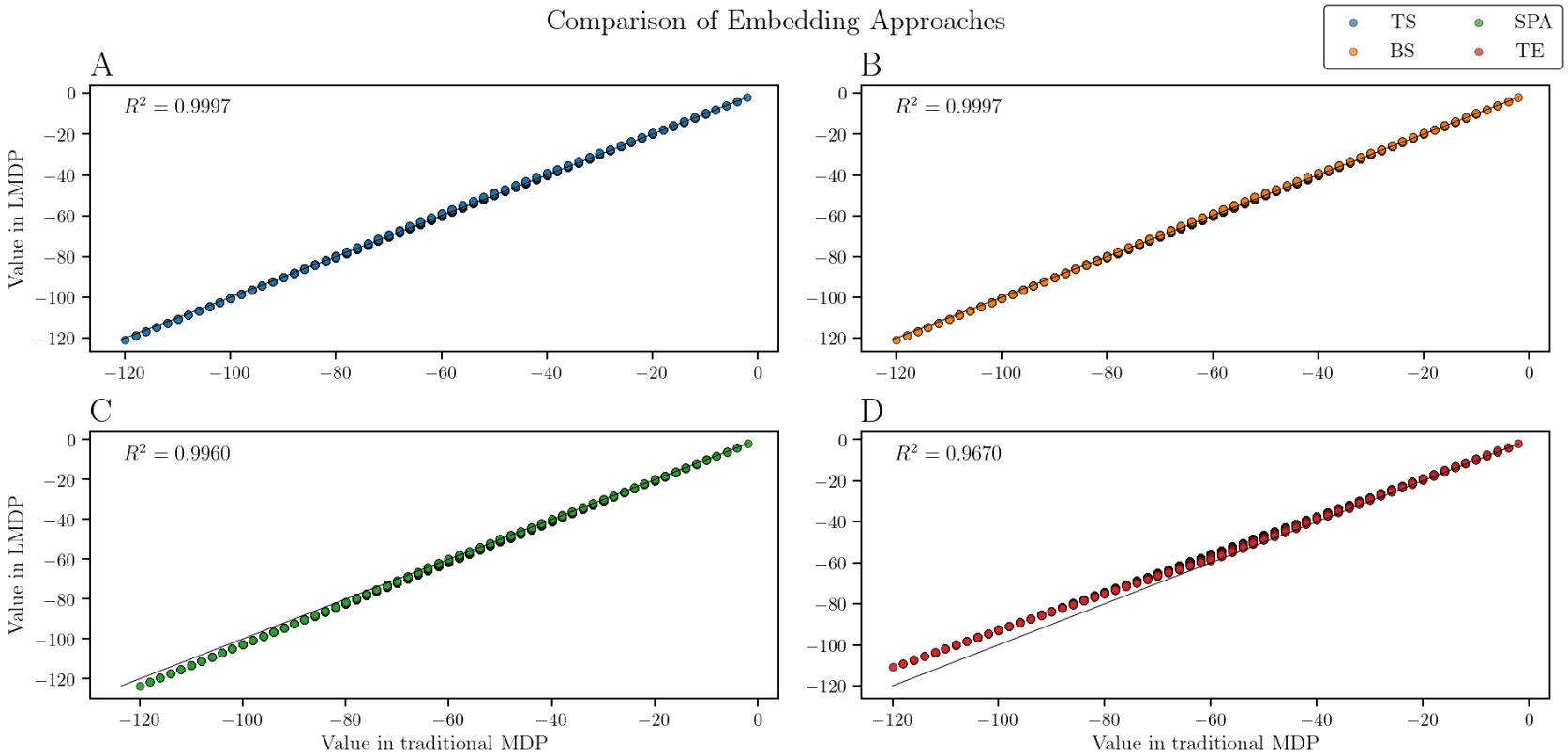

Figure 3: Embedding Approximations by different approaches.

Embedding of Deterministic MDP

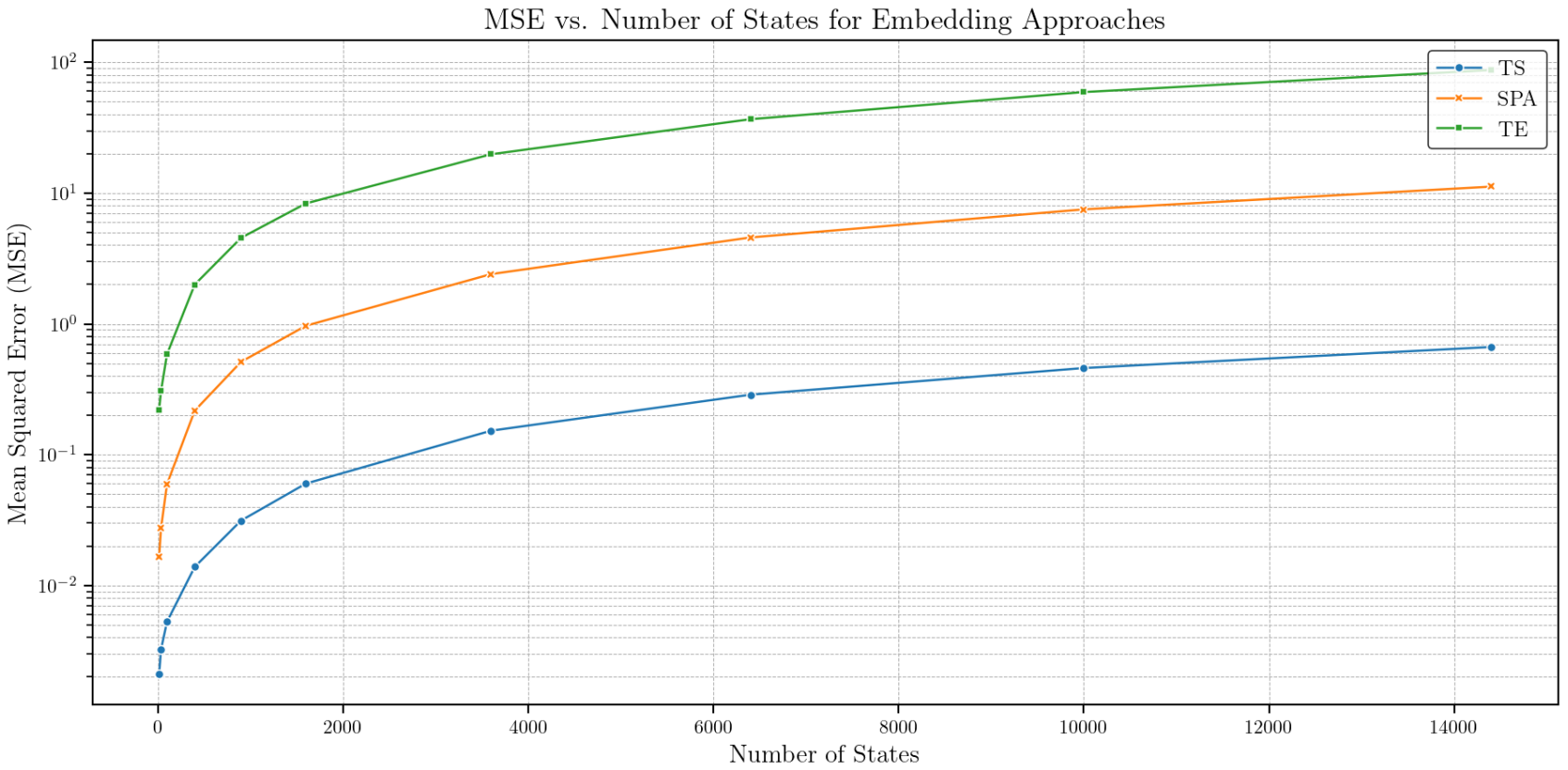

Figure 4: MSE vs. Number of States for Embedding Approaches.

Embedding of Deterministic MDP

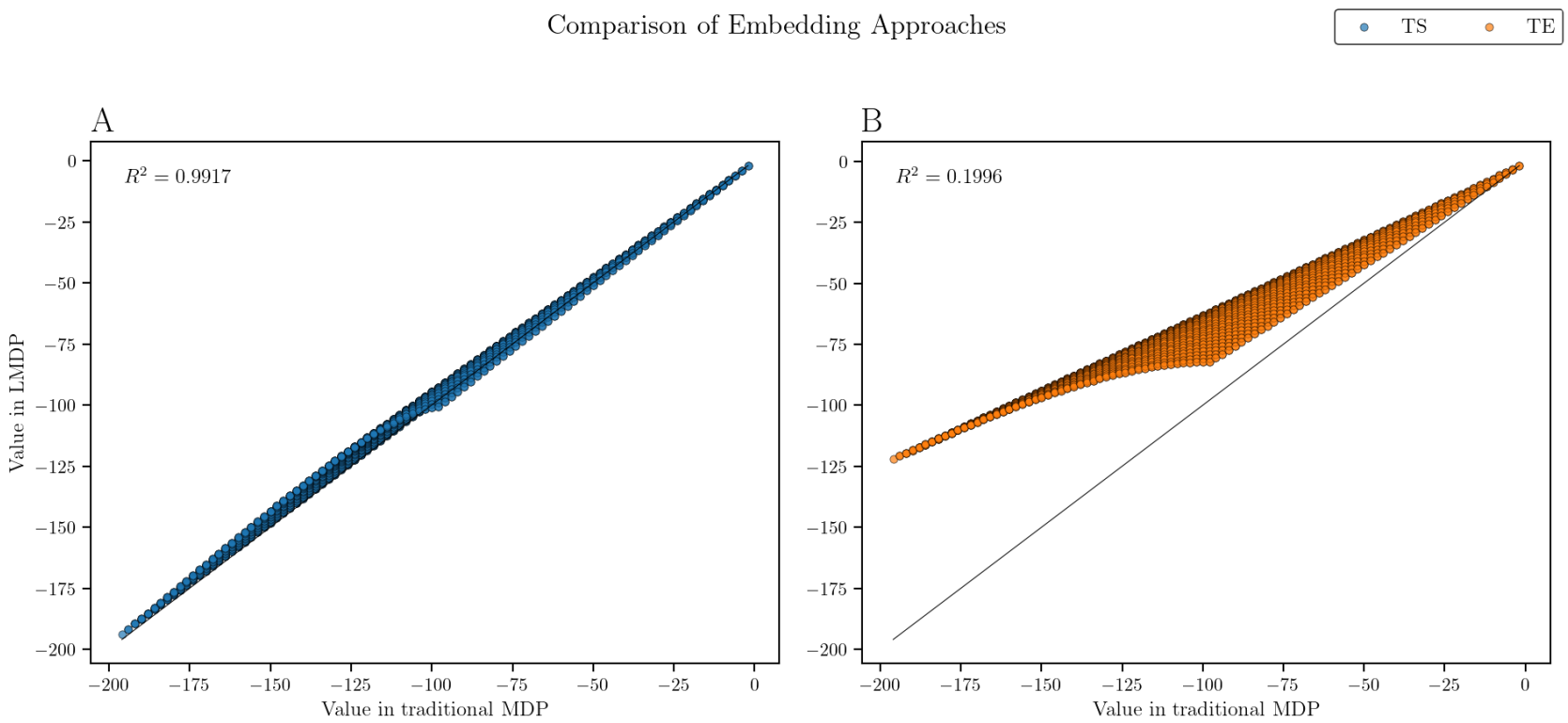

Figure 5: Embedding Approximation by TS and baseline methods.

Embedding of LMDP

Todorov's Embedding[2]

- **Reward Approximation** through KL divergence:

$$\tilde{\mathcal{R}}(s,a) = \mathcal{R}(s) - \lambda KL\left(\mathcal{P}_{\mathbf{u}} \middle\| \mathcal{P}\right) \quad \forall s \in \mathcal{S}\ \forall a \in \mathcal{A}\ . \tag{24}$$

- **Transition Probability Approximation** through circular shifting:

$$\tilde{\mathcal{P}}(s' \mid s,a) = \mathcal{P}^a_{\mathbf{u}}(s' \mid s) \quad \forall s \in \mathcal{S}^-\ \forall s' \in \mathcal{S}\ \forall a \in \mathcal{A}\ , \tag{25}$$

where $\mathcal{P}^a_{\mathbf{u}}$ is the a_{th} shift of non-zero elements of $\mathcal{P}_{\mathbf{u}}$.

[2] Todorov (2009)

Grid World Domain

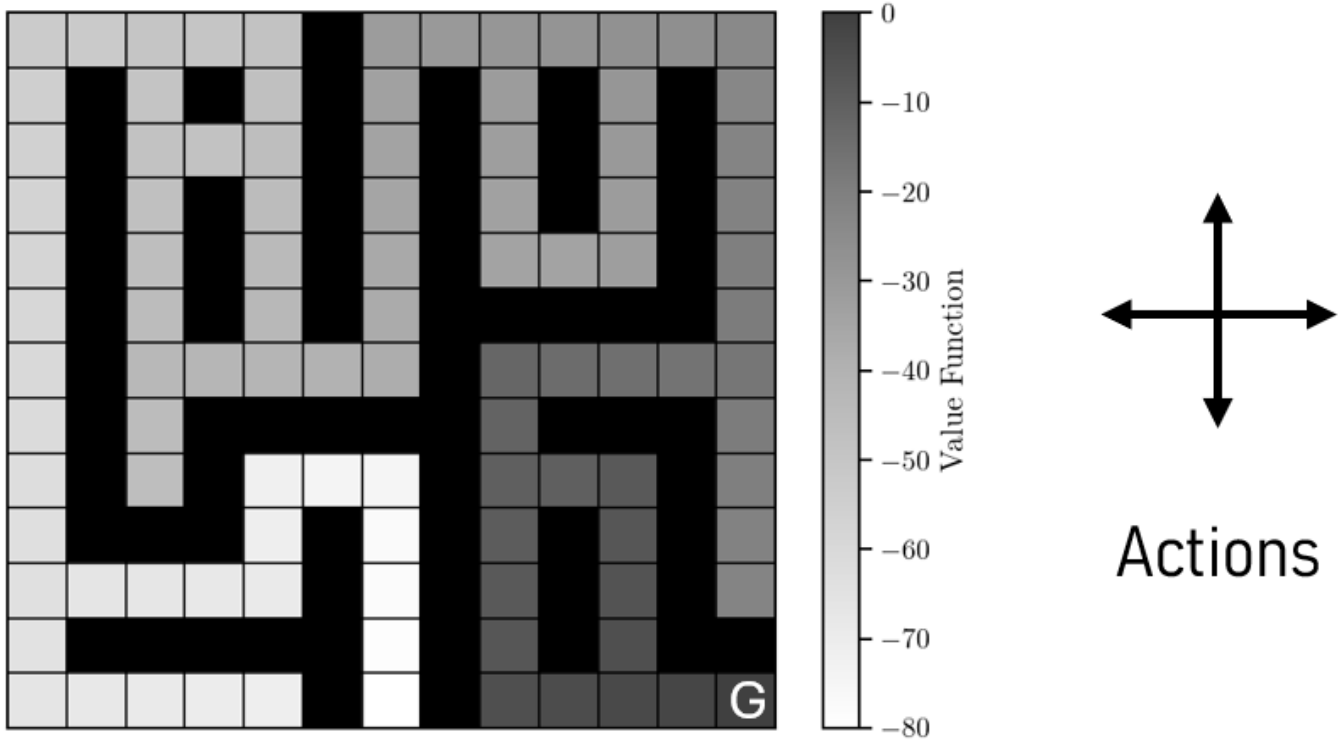

Figure 6: Simple Grid Environment

Grid World Domain

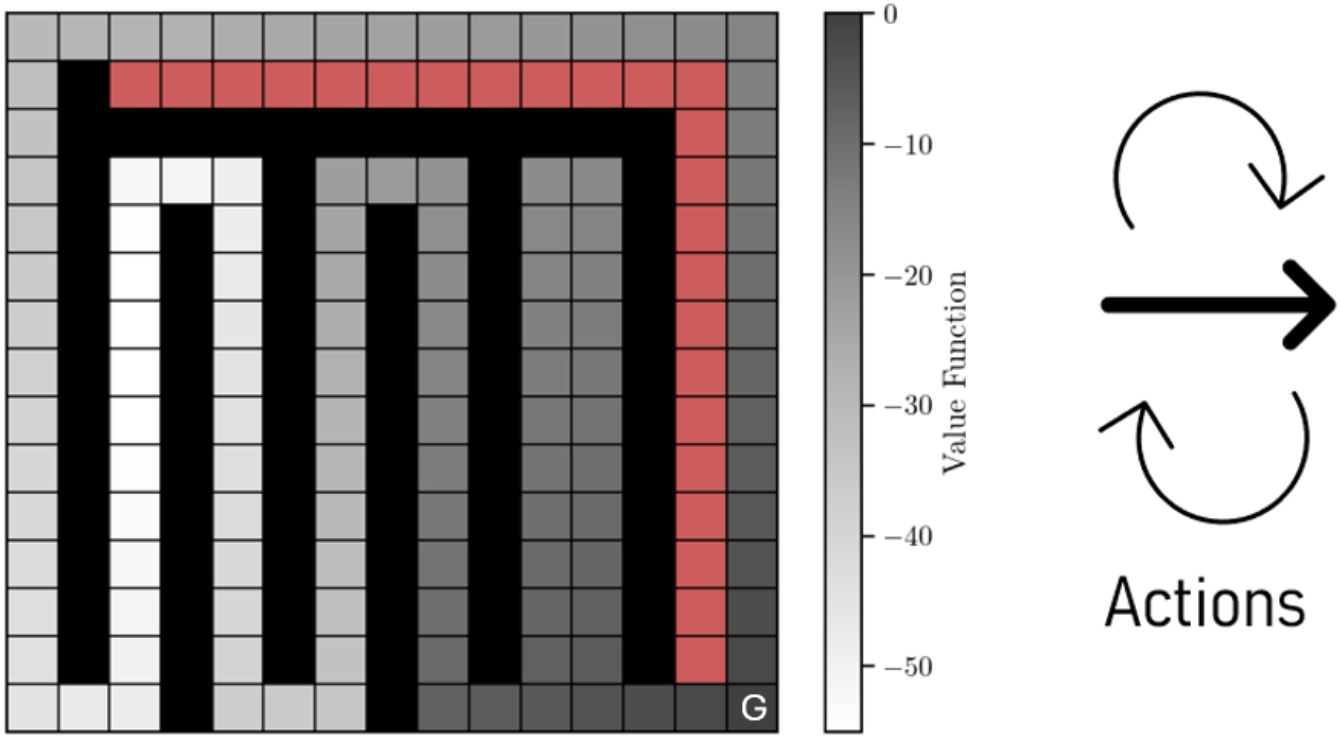

Figure 7: Directional Grid Environment

Exploration Strategy

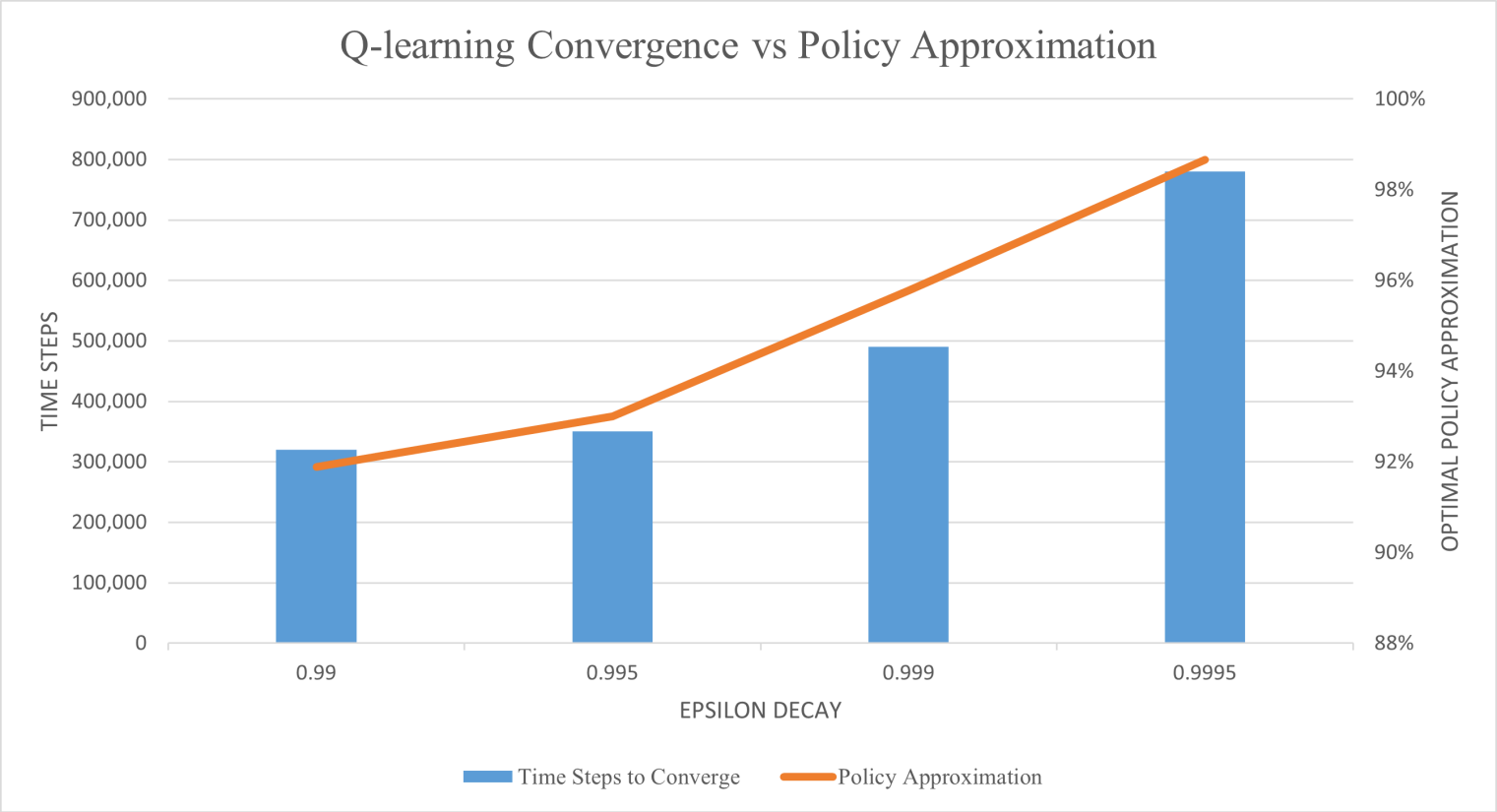

Figure 8: Q-learning convergence and policy approximation by ϵ-decay

Exploration Strategy

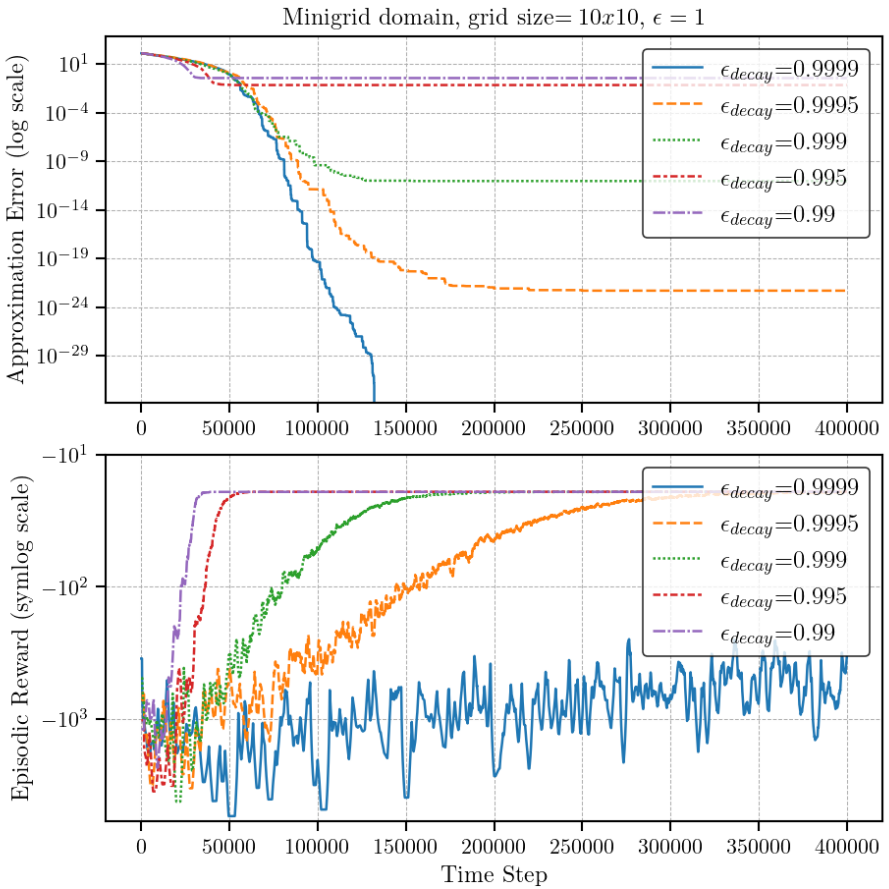

Figure 9: Q-learning convergence and approximation error by ϵ decay

Exploration Strategy

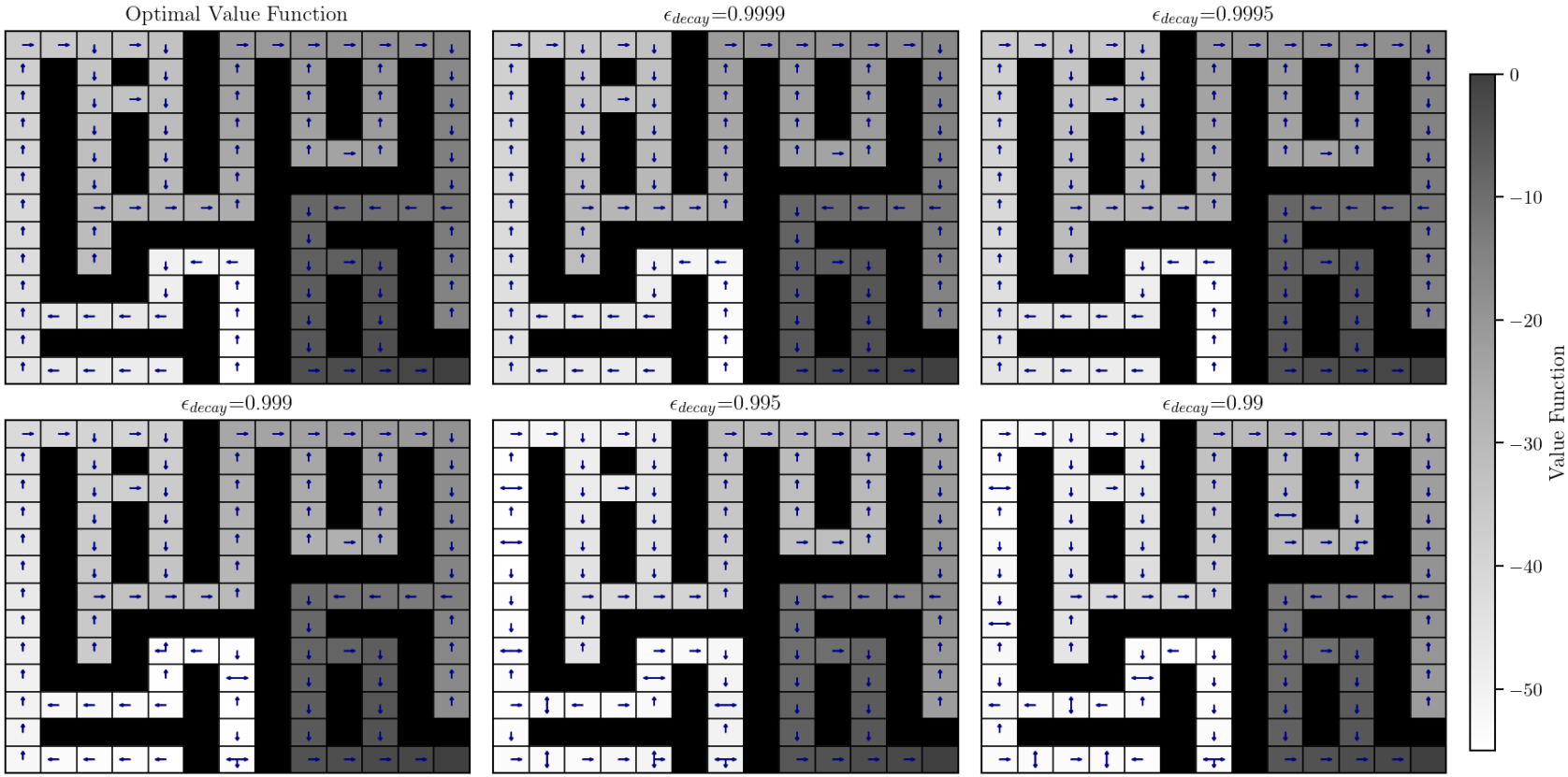

Figure 10: Value function approximation by ϵ decay

Benchmarking and Evaluation

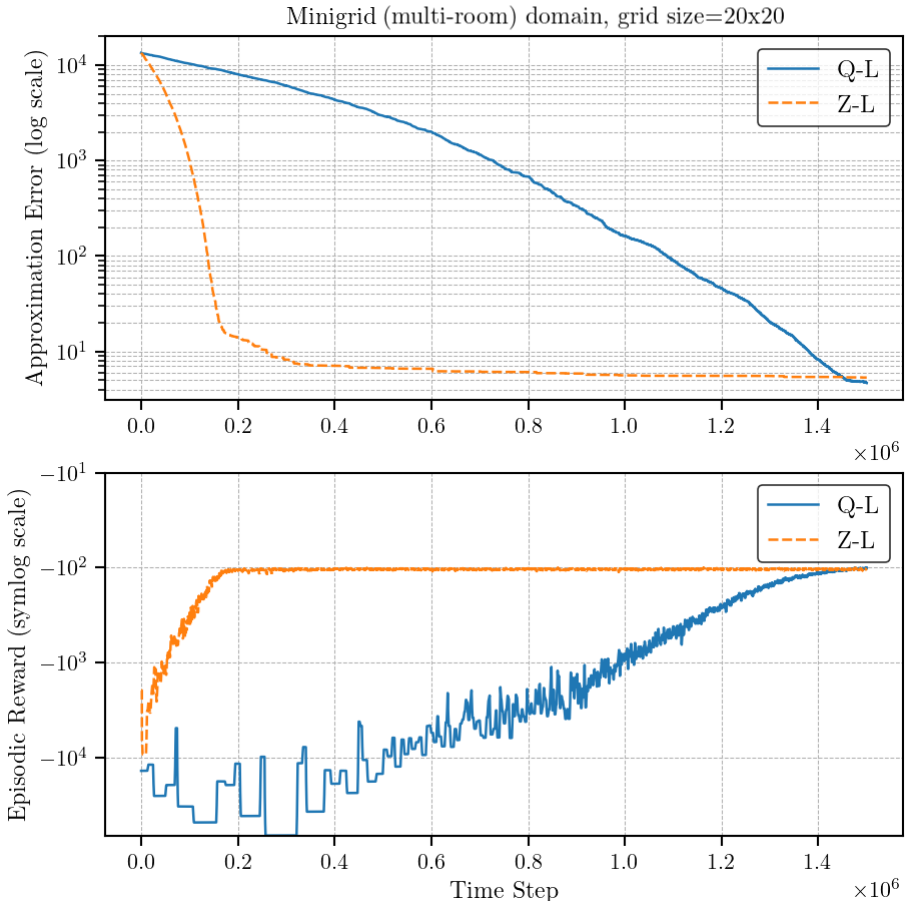

Figure 11: Comparison of Q-learning and Z-learning.

Benchmarking and Evaluation

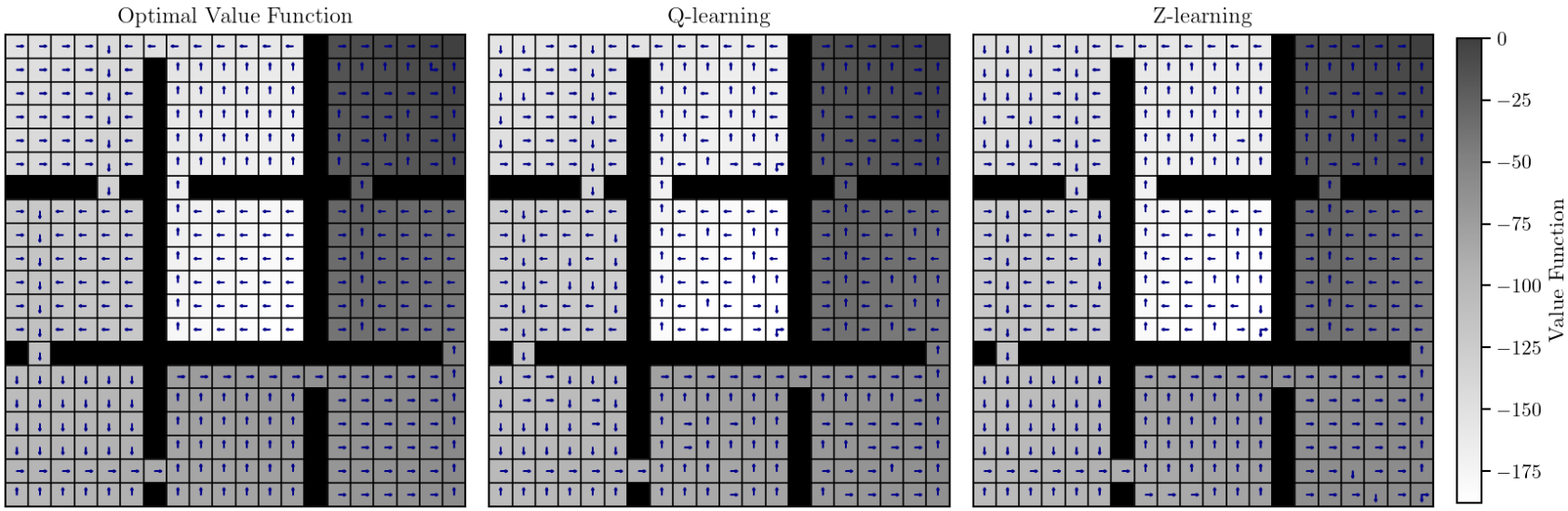

Figure 12: Comparison optimal value function, Q-learning and Z-learning approximations in multi-room domain.

Benchmarking and Evaluation

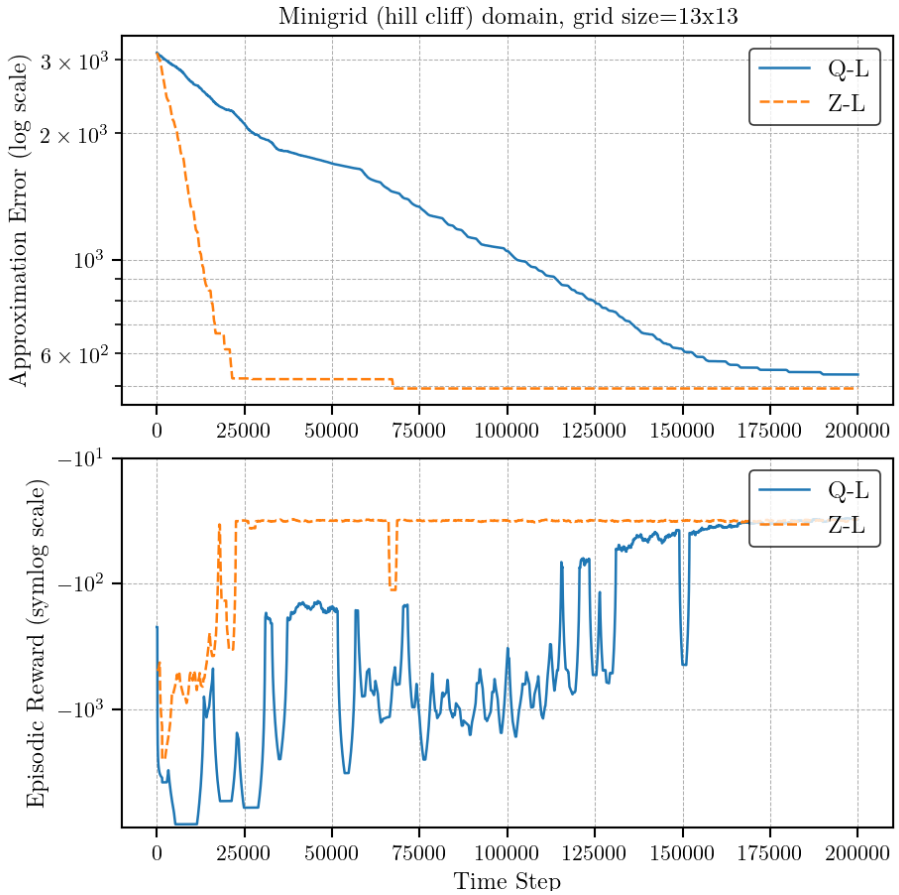

Figure 13: Comparison of Q-learning and Z-learning.

Benchmarking and Evaluation

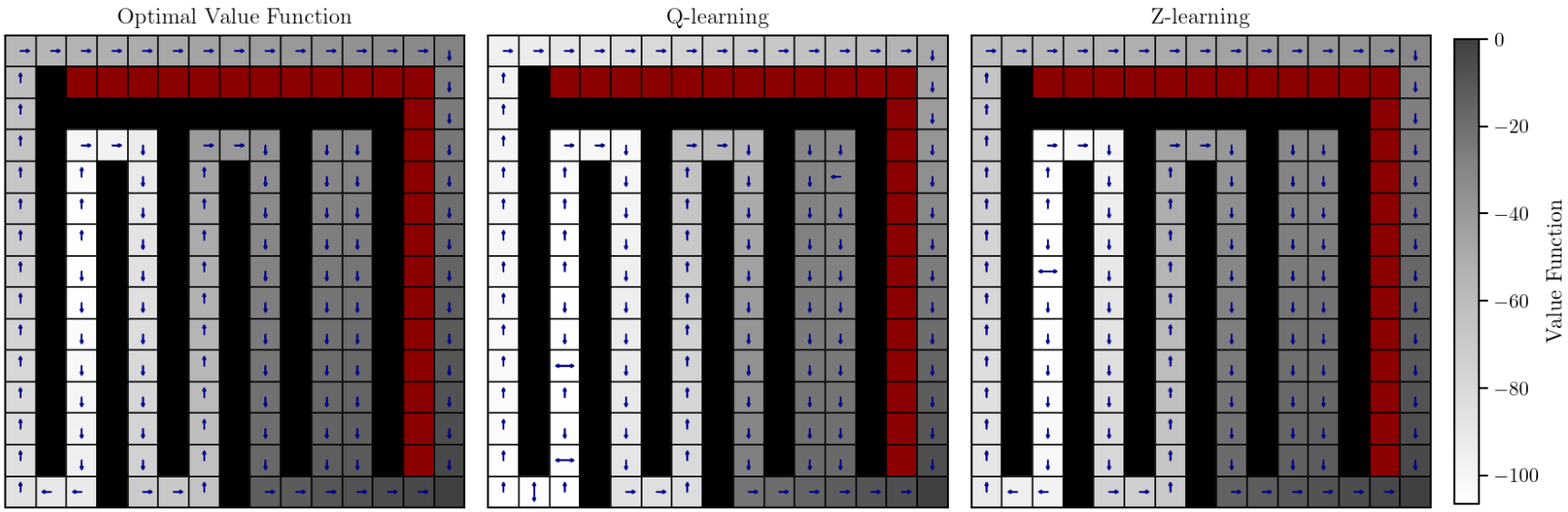

Figure 14: Comparison optimal value function, Q-learning and Z-learning approximations in multi-room domain.

Conclusions

01

LMDP's improvements in scalability and efficiency.

Conclusions

01

LMDP's improvements in scalability and efficiency.

02

Development of robust embedding techniques.

Conclusions

01

LMDP's improvements in scalability and efficiency.

02

Development of robust embedding techniques.

03

Exploration strategy and algorithms design optimization.

Future Work

Experimentation in larger domains.

Future Work

Experimentation in larger domains.

Framework extension to dynamic environments.

Future Work

Experimentation in larger domains.

Framework extension to dynamic environments.

Deep learning techniques integration.

Bellman, R. (1957). A Markovian Decision Process. *Indiana University Mathematics Journal*, 6(4):679–684.

Bertsekas, D. P. (1987). *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall, Englewood Cliffs, NJ.

Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming*, volume 27. Athena Scientific.

Gómez, V., Kappen, H. J., Peters, J., and Neumann, G. (2014). Policy search for path integral control. In Calders, T., Esposito, F., Hüllermeier, E., and Meo, R., editors, *Machine Learning and Knowledge Discovery in Databases*, pages 482–497, Berlin, Heidelberg. Springer Berlin Heidelberg.

Gómez, V., Thijssen, S., Symington, A., Hailes, S., and Kappen, H. J. (2020). Real-time stochastic optimal control for multi-agent quadrotor systems.

Infante, G., Jonsson, A., and Gómez, V. (2022). Globally optimal hierarchical reinforcement learning for linearly-solvable markov decision processes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(6):6970–6977.

Jaakkola, T., Jordan, M. I., and Singh, S. P. (1994). On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6:1185–1201.

Jonsson, A. and Gómez, V. (2016). Hierarchical linearly-solvable markov decision problems. In *Proceedings of the 26th International Conference on Automated Planning and Scheduling (ICAPS)*.

Kappen, H. J. (2005a). Linear theory for control of nonlinear stochastic systems. *Phys. Rev. Lett.*, 95:200201.

Kappen, H. J. (2005b). Linear theory for control of nonlinear stochastic systems. *Phys. Rev. Lett.*, 95:200201.

Kappen, H. J., Gómez, V., and Opper, M. (2012). Optimal control as a graphical model inference problem. *Machine Learning*, 87(2):159–182.

Levine, S. (2018). Reinforcement learning and control as probabilistic inference: Tutorial and review.

Molina, G. I., Jonsson, A., and Gómez, V. (2023). Optimal hierarchical average-reward linearly-solvable markov decision processes. In *Sixteenth European Workshop on Reinforcement Learning*.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, USA, 1st edition.

Rawlik, K., Toussaint, M., and Vijayakumar, S. (2012). On stochastic optimal control and reinforcement learning by approximate inference. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia.

Rummery, G., Niranjan, M., and of Cambridge. Engineering Department, U. (1994). *On-line Q-learning Using Connectionist Systems*. CUED/F-INFENG/TR. University of Cambridge, Department of Engineering.

Sutton, R. S. (1991). Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: an introduction*. Adaptive computation and machine learning series. The MIT Press, Cambridge, Massachusetts, second edition edition.

Todorov, E. (2006). Linearly-solvable markov decision problems. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

Todorov, E. (2009). Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences of the United States of America*, 106:11478–83.

Tsitsiklis, J. N. (1994). Asynchronous stochastic approximation and q-learning. *Machine Learning*, 16(3):185–202.

Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards*. PhD thesis, University of Cambridge.

Watkins, C. J. C. H. and Dayan, P. (1992). Q-learning. *Machine Learning*, 8(3-4):279–292.

Williams, G., Aldrich, A., and Theodorou, E. (2017). Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40:1–14.

Efficient Algorithms for Linearly Solvable Markov Decision Processes

Bachelor's Thesis

David Pérez Carrasco

Supervisor : Anders Jonsson

Universitat Pompeu Fabra

July 1st, 2024

Thank You!

Q/A

Questions and Answers

Any Questions?

Learning Rate

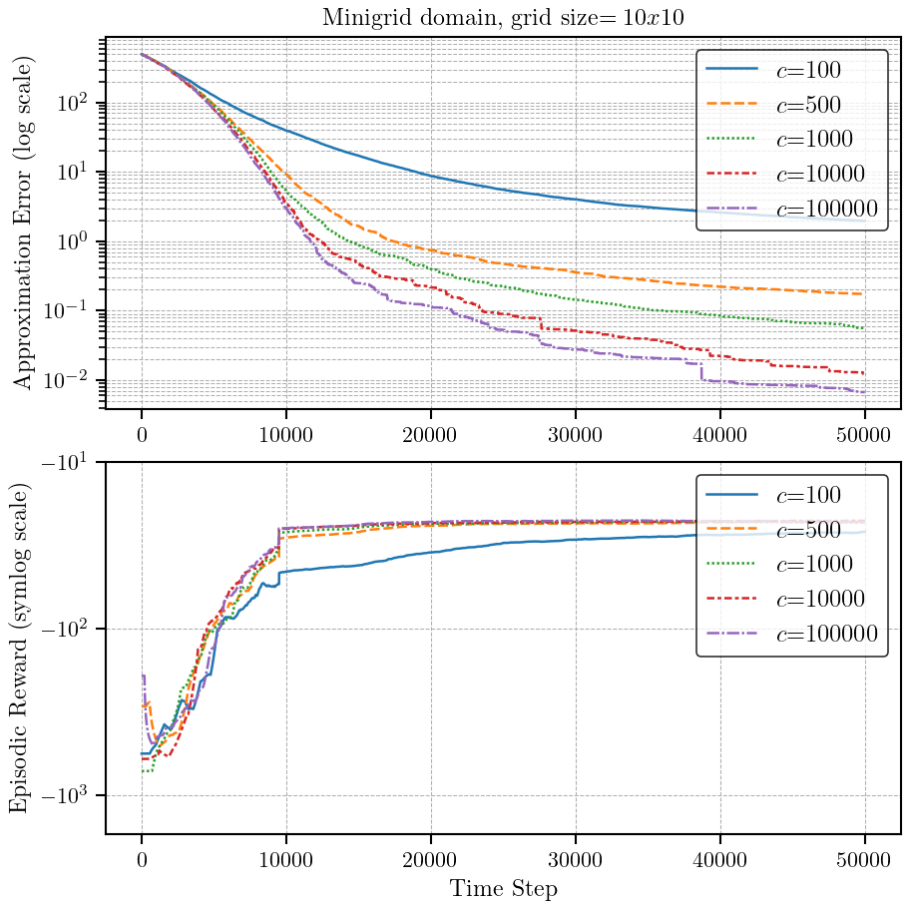

Figure 15: Comparison of different c values.

Learning Rate

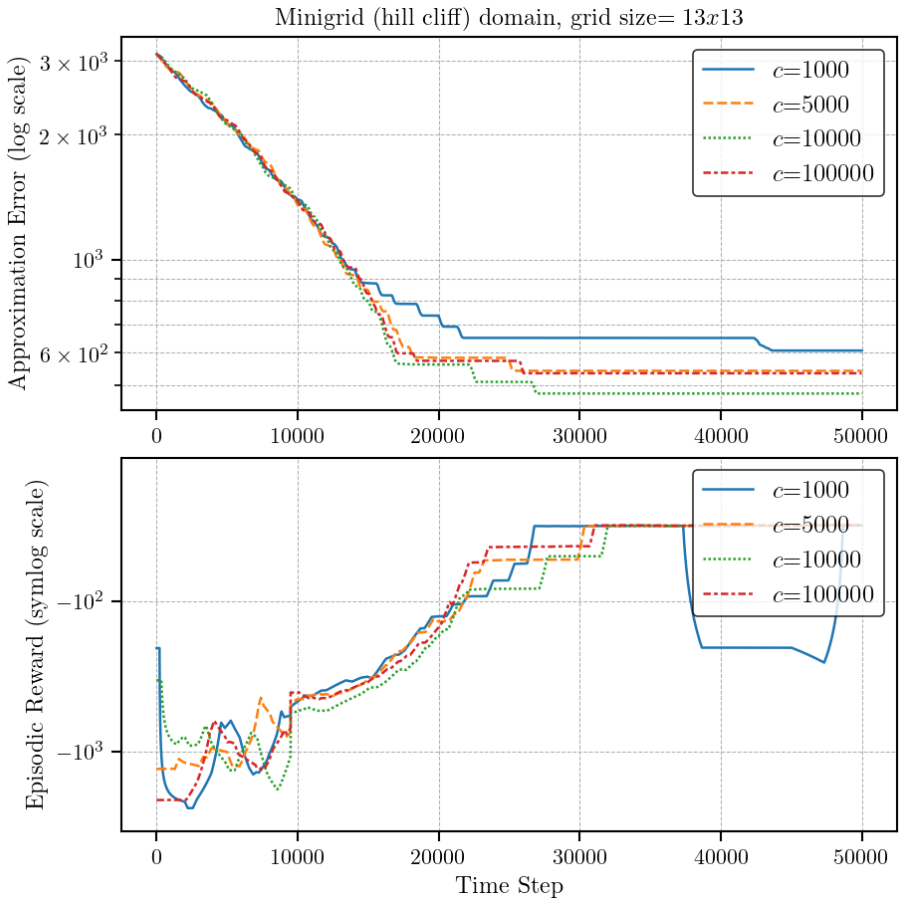

Figure 16: Comparison of different c values.

Temperature Parameter

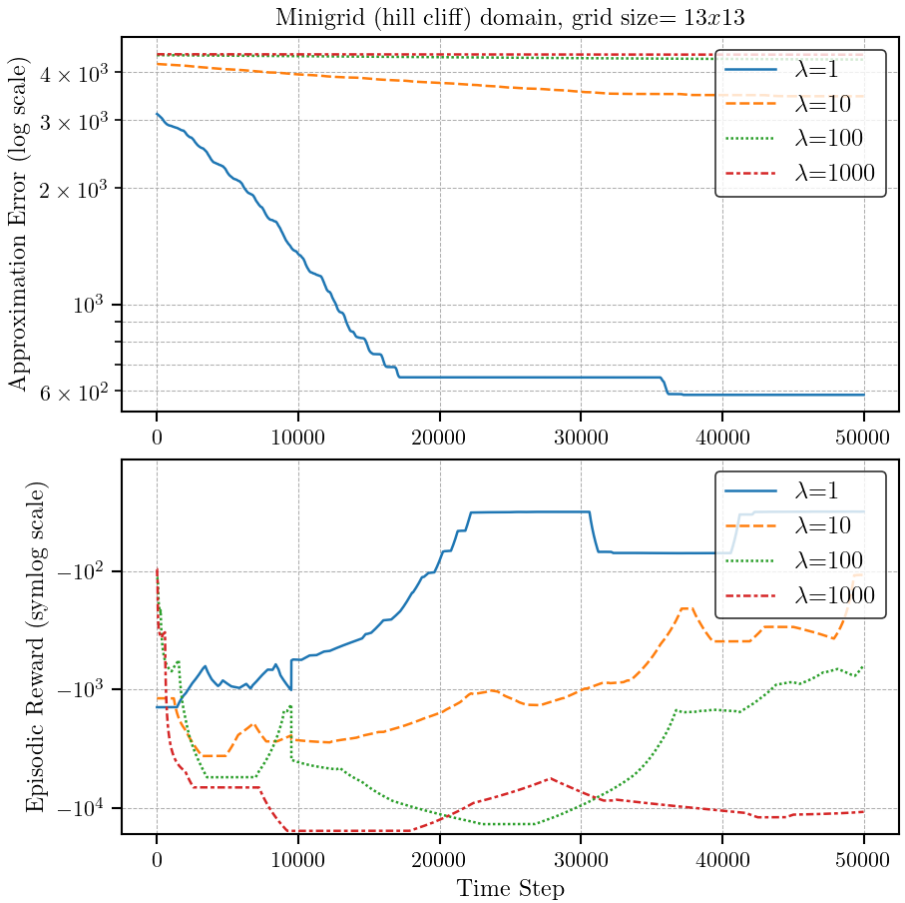

Figure 17: Comparison of different λ values.

Temperature Parameter

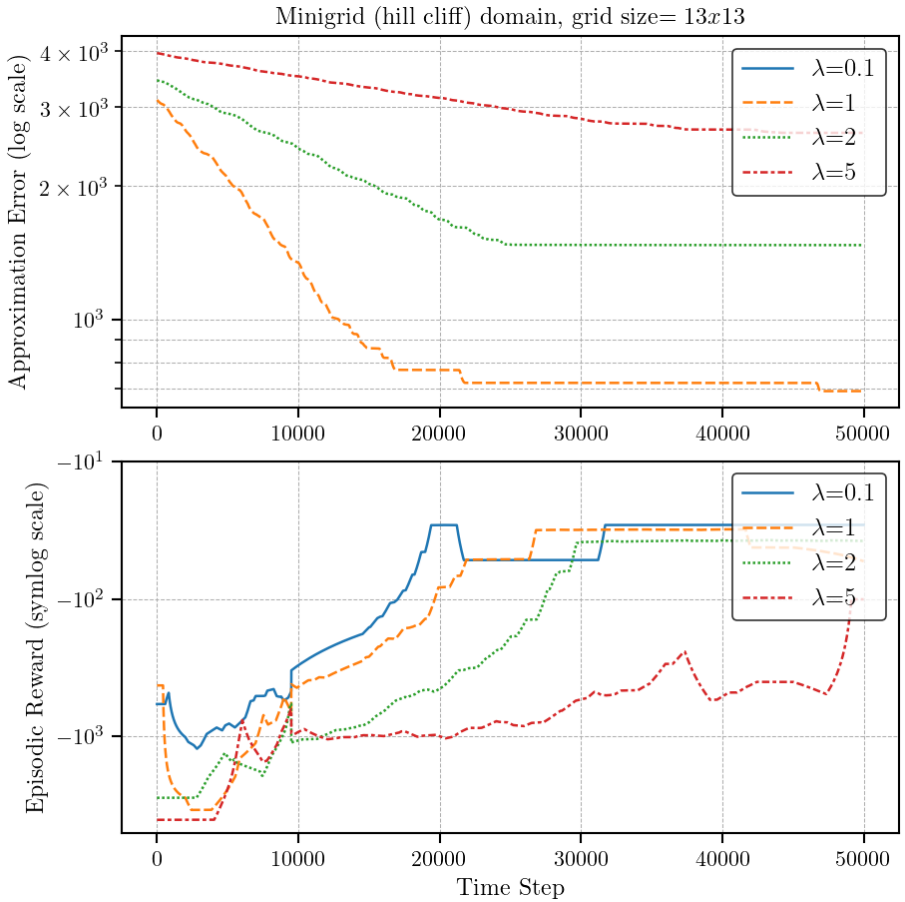

Figure 18: Comparison of different λ values.

Core Components of RL

- Policy
  - Deterministic Policy: $\pi : \mathcal{S}^- \to \mathcal{A}$
  - Stochastic Policy: $\pi : \mathcal{S}^- \to \Delta(\mathcal{A})$

Core Components of RL

- Policy
- Reward Function

$$\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$$

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}. \tag{26}$$

Core Components of RL

- Policy
- Reward Function
- Value Function

$$v_\pi(s) \doteq \mathbf{E}_\pi\left[G_t | S_t = s\right] = \mathbf{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right] \forall s \in \mathcal{S}. \tag{26}$$

$$\begin{aligned} q_\pi(s,a) &\doteq \mathbf{E}_\pi\left[G_t | S_t = s, A_t = a\right] \\ &= \mathbf{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a\right] \forall s \in \mathcal{S}, \forall a \in \mathcal{A}. \end{aligned} \tag{27}$$

Sutton and Barto (2018).

Core Components of RL

- Policy
- Reward Function
- Value Function
- Model

Embedding of Deterministic MDP

Algorithm 4 Deterministic MDP Embedding through iterative KL update

input: MDP $\mathcal{M}$ with $\mathcal{A}$, $\mathcal{S}$, $\mathcal{T}$, $\mathcal{S}^-$, $\tilde{\mathcal{R}}$ and deterministic $\tilde{\mathcal{P}}$, temperature parameter λ, optimal value function V^* from $\mathcal{M}$, convergence threshold ϵ, stopping criteria parameter N
output: LMDP $\mathcal{L}$ with $\hat{\mathcal{R}}$ and $\mathcal{P}$
$\hat{\mathcal{R}}(\cdot) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{R}}(\cdot, a)$
$\mathcal{P}(\cdot \mid \cdot) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{P}}(\cdot|\cdot, a)$
Obtain Z from power iteration for LMDP $\mathcal{L}$ with $\mathcal{P}$ and $\hat{\mathcal{R}}$
$V = \lambda \log Z$
$n = 0$
while $MSE(V^*, V) > \epsilon$ and $n < N$ **do**
    $\mathcal{P}_{\mathbf{u}^*} \leftarrow \mathcal{P}Z / \sum_{s' \in \mathcal{S}} \mathcal{P}(s' \mid \cdot) Z(s')$
    $\hat{\mathcal{R}} \leftarrow \hat{\mathcal{R}} + \lambda KL(\mathcal{P}_{\mathbf{u}^*} \| \mathcal{P})$
    $n \leftarrow n + 1$
end while
return $\mathcal{L}$ with $\hat{\mathcal{R}}$ and $\mathcal{P}$

Embedding of Deterministic MDP

Algorithm 5 Deterministic MDP Embedding through binary search

input: MDP $\mathcal{M}$ with $\mathcal{A}$, $\mathcal{S}$, $\mathcal{T}$, $\mathcal{S}^-$, $\tilde{\mathcal{R}}$ and deterministic $\tilde{\mathcal{P}}$, temperature parameter λ, small ϵ, optimal value function V^* from $\mathcal{M}$
output: LMDP $\mathcal{L}$ with $\mathcal{R}$
$\hat{\mathcal{R}}(\cdot) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{R}}(\cdot, a)$
$\mathcal{P}(\cdot|\cdot) \leftarrow \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \tilde{\mathcal{P}}(\cdot|\cdot, a)$
Obtain Z from power iteration method for LMDP $\mathcal{L}$ with $\mathcal{P}$ and $\hat{\mathcal{R}}$
$\mathcal{P}_{\mathbf{u}^*} \leftarrow \mathcal{P}Z / \sum_{s' \in \mathcal{S}} \mathcal{P}(s'|\cdot)Z(s')$
$\hat{\mathcal{R}} \leftarrow \hat{\mathcal{R}} + \lambda KL(\mathcal{P} \| \mathcal{P}_{\mathbf{u}^*})$
Set $K_{min} = 0$ and $K_{max} = 1$
$V^*_{MS} = \frac{1}{|\mathcal{S}|} \left(\sum_{s \in \mathcal{S}} V^* \right)^2$
while $K_{max} - K_{min} \geq \epsilon$ **do**
    $K = (K_{min} + K_{min})/2$
    $\hat{\mathcal{R}}' = K \cdot \hat{\mathcal{R}}$
    Obtain Z' from power iteration method for LMDP $\mathcal{L}$ with $\mathcal{P}$ and $\hat{\mathcal{R}}'$
    $V = \lambda \log Z'$
    $V_{MS} = \frac{1}{|\mathcal{S}|} \left(\sum_{s \in \mathcal{S}} V \right)^2$
    if $V^*_{MS} > V_{MS}$ **then**
        $K_{min} = K$
    else
        $K_{max} = K$
    end if
end while
$\mathcal{R} = K \cdot \hat{\mathcal{R}}$
return $\mathcal{L}$ with $\mathcal{P}$ and $\mathcal{R}$

Embedding of LMDP

Algorithm 6 Vectorized Embedding of LMDP into MDP

input: LMDP $\mathcal{L}$ with $\mathcal{R}$ and $\mathcal{P}$, temperature parameter λ
output: MDP $\mathcal{M}$ with $\tilde{\mathcal{P}}$ and $\tilde{\mathcal{R}}$
Obtain Z from power iteration method for LMDP $\mathcal{L}$ with $\mathcal{R}$ and $\mathcal{P}$
$\mathcal{P}_{\mathbf{u}^*} \leftarrow \mathcal{P}Z / \sum_{s' \in \mathcal{S}} \mathcal{P}(s''|\cdot)Z(s')$
$\max(a) \leftarrow \max N(s)$ for all $s \in \mathcal{S}^-$
$\tilde{\mathcal{R}}(\cdot, a) \leftarrow \mathcal{R}(\cdot) - \lambda KL\left(\mathcal{P}_{\mathbf{u}^*}(\cdot \mid \cdot) \| \mathcal{P}(\cdot \mid \cdot)\right)$ for all actions a
Identify the number of potential next states $N(s)$ from each state $s \in \mathcal{S}^-$
Obtain the set $\mathcal{N}$ by collecting the unique values of $N(s)$ for all states $s \in \mathcal{S}^-$
for $n \in \mathcal{N}$ **do**
    Identify state space $\mathcal{S}^n$ of states s with $N(s) = n$
    Identify state space $\mathcal{S}'^n$ of potential next states s' of states s in $\mathcal{S}^n$
    for action a in $\max(a)$ **do**
        $\tilde{\mathcal{P}}(\mathcal{S}'^n \mid \mathcal{S}^n, a) \leftarrow \mathcal{P}_{\mathbf{u}^*}(\mathcal{S}'^n \mid \mathcal{S}^n)$ shifted a times.
    end for
end for
return $\mathcal{M}$ with $\tilde{\mathcal{R}}$ and $\tilde{\mathcal{P}}$

Embedding of Deterministic MDP

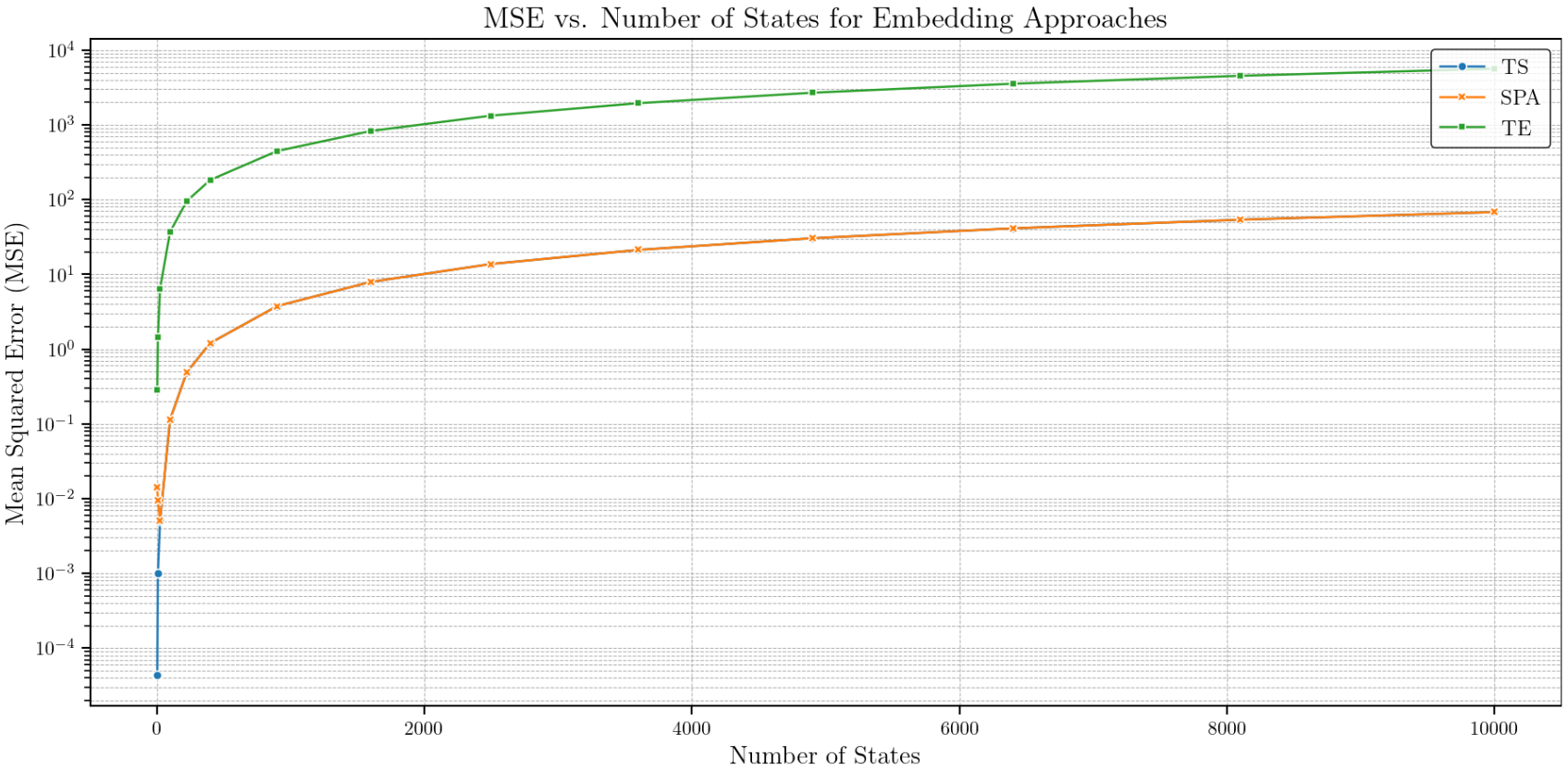

Figure 19: Embedding MSE Comparison in Simple Grid

Embedding of Stochastic MDP

Grid Size	Number of States	Loop Time (s)	Vectorized Time (s)
2	4	0.00042	0.00007
3	9	0.00120	0.00021
5	25	0.00311	0.00053
10	100	0.01111	0.00153
15	225	0.02439	0.00315
20	400	0.04360	0.00596
30	900	0.11308	0.02454
40	1600	0.20145	0.06201
50	2500	0.34319	0.13674
60	3600	0.55813	0.27363
70	4900	0.84053	0.49956
80	6400	1.34381	0.94295
90	8100	1.78063	1.38341
100	10000	2.36575	1.95391

Table 1: Embedding execution times for different grid sizes using vectorized and loop approaches.

Embedding of LMDP

Grid Size	Number of States	Loop Time (s)	Vectorized Time (s)
2	4	0.00844	0.00590
3	9	0.01250	0.00845
5	25	0.02302	0.00899
10	100	0.08089	0.01512
15	225	0.18019	0.02333
20	400	0.33202	0.05006
30	900	0.83924	0.21857
40	1600	1.92297	0.83924
50	2500	3.58723	2.03189
60	3600	7.30093	4.79076
70	4900	14.47642	9.59661
80	6400	20.80158	16.69498
90	8100	33.92493	29.42800
100	10000	54.42098	51.65870

Table 2: Embedding execution times for different grid sizes using vectorized and loop approaches.

Value Iteration

Algorithm 7 Value Iteration

input: discount rate $\gamma \in [0, 1)$, a small positive constant θ, MDP with $\mathcal{R}$, $\mathcal{P}$, $\mathcal{S}$, $\mathcal{A}$, $\mathcal{S}^-$, $\mathcal{T}$
output: $V : \mathcal{S} \to \mathbb{R}$
initialize $V(s) \leftarrow 0 \quad \forall s \in \mathcal{S}^-$, $V(s) \leftarrow \mathcal{R}(s, \cdot) \quad \forall s \in \mathcal{T}$
repeat
    $\Delta \leftarrow 0$
    for each state $s \in \mathcal{S}^-$ **do**
        $v \leftarrow V(s)$
        $V(s) \leftarrow \max_{a \in \mathcal{A}}(\mathcal{R}(s, a) + \gamma \sum_{s'} \mathcal{P}(s'|s, a) V(s'))$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
    end for
until $\Delta < \theta$

Vectorized Value Iteration

Algorithm 8 Vectorized Value Iteration Algorithm

input: MDP with state space $\mathcal{S}$, terminal state space $\mathcal{T}$, non-terminal state space $\mathcal{S}^-$, action space $\mathcal{A}$, reward matrix $\mathcal{R}$, transition matrix $\mathcal{P}$, discount factor γ, and convergence threshold ϵ
output: Action-value function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, policy $\pi : \mathcal{S} \to \mathcal{A}$, and number of iterations N
initialize: $Q(s,a) \leftarrow 0 \ \forall s \in \mathcal{S}^- \ \forall a \in \mathcal{A}$, $Q(s,a) \leftarrow \mathcal{R}(s,a) \ \forall s \in \mathcal{T} \ \forall a \in \mathcal{A}$, $N \leftarrow 0$, V_{diff} with the state indices
$R \leftarrow \mathcal{R}(\mathcal{S}^-)$
$P \leftarrow \gamma\mathcal{P}$
$QT \leftarrow \mathcal{R}(\mathcal{T})$
while $\max(V_{diff}) - \min(V_{diff}) > \epsilon$ **do**
    $TQ \leftarrow R + P\max_a(Q)$
    $TQ \leftarrow (TQ, QT)$
    $V_{diff} \leftarrow \max_a(TQ) - \max_a(Q)$
    $Q \leftarrow TQ$
    $N \leftarrow N + 1$
end while
$\pi \leftarrow \arg_a \max(Q)$
return Q, π, N

Q-learning

Algorithm 9 Q-learning

input: discount rate $\gamma \in (0, 1]$, exploration factor $\epsilon \in (0, 1]$, MDP with $\mathcal{R}$, $\mathcal{P}$, $\mathcal{S}$, $\mathcal{A}$, $\mathcal{S}^-$, $\mathcal{T}$
output: $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$
initialize $Q(s, a) \leftarrow 0 \; \forall s \in \mathcal{S}^- \; \forall a \in \mathcal{A}$, $Q(s, a) \leftarrow \mathcal{R}(s, a) \; \forall s \in \mathcal{T} \; \forall a \in \mathcal{A}$
repeat
    $s_t \leftarrow s_0$ (sample state from initial state distribution)
    while $s \notin \mathcal{T}$ **do**
        Select action a_t using policy derived from Q (e.g., ϵ-greedy)
        Take action a_t, observe r_{t+1}, s_{t+1}
        $Q(s_t, a_t) \leftarrow (1 - \alpha) Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \right]$
        $s_t \leftarrow s_{t+1}$
    end while
until convergence
return Q

ϵ-Greedy Q-learning

Algorithm 10 Q-learning with ϵ decay

input: discount rate $\gamma \in [0,1]$, exploration factor $\epsilon \in (0,1]$, ϵ decay parameter $d_\epsilon \in (0,1]$, $min_\epsilon \in [0,1)$, $c \in \mathbb{R}$, restart randomness parameter r_{s_0}, MDP with $\mathcal{R}$, $\mathcal{P}$, $\mathcal{S}$, $\mathcal{A}$, $\mathcal{S}^-$, $\mathcal{T}$, s_0
output: $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$
initialize $Q(s,a) \leftarrow 0$ for all $s \in \mathcal{S}^-$, $a \in \mathcal{A}$, $Q(s,a) \leftarrow \mathcal{R}(s,a)$ for all $s \in \mathcal{T}$, $a \in \mathcal{A}$, $N = 0$, $\alpha = 1$
repeat
    if $r_{s_0} = 0$ **then**
        $s = s_0$
    else
        Sample a random state s from $\mathcal{S}^-$
    end if
    while $s \notin \mathcal{T}$ **do**
        Choose a from s using ϵ-greedy policy derived from Q
        Take action a, observe r, s'
        $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$
        $s \leftarrow s'$
    end while
    $N \leftarrow N + 1$
    $\epsilon \leftarrow \max\left(min_\epsilon, \epsilon \cdot d_\epsilon\right)$
    $\alpha \leftarrow c/\left(c + N\right)$
until convergence
output: Q

Power Iteration

Algorithm 11 Power Iteration Algorithm

input: LMDP with initial state distribution $\mathcal{P}$, state space $\mathcal{S}$, terminal state space $\mathcal{T}$, non-terminal state space $\mathcal{S}^-$, reward matrix $\mathcal{R}$, temperature parameter λ, and convergence threshold ϵ
output: Optimal Z function and number of iterations N
initialize: $Z \leftarrow \mathbf{1}_{|\mathcal{S}|}$, $N \leftarrow 0$, V_{diff} with the state indices
$ZT \leftarrow e^{\mathcal{R}(\mathcal{T})/\lambda}$
$G \leftarrow e^{\mathcal{R}(\mathcal{S}^-)/\lambda}$ as a diagonal matrix
while $\max(V_{diff}) - \min(V_{diff}) > \epsilon$ **do**
    $TZ \leftarrow G\mathcal{P}Z$
    $TZ \leftarrow (TZ, ZT)$
    $TV \leftarrow \lambda \log TZ$
    $V \leftarrow \lambda \log Z$
    $V_{diff} \leftarrow (TZ) - (Z)$
    $Z \leftarrow TZ$
    $N \leftarrow N + 1$
end while
return Z, N

Z-learning

Algorithm 12 Z-learning

input: learning rate $\alpha \in (0, 1]$, temperature parameter $\lambda > 0$, LMDP with $\mathcal{R}$, $\mathcal{P}$, $\mathcal{S}$, $\mathcal{S}^-$, $\mathcal{T}$
output: $\hat{Z} : S \to \mathbb{R}$
initialize $\hat{Z}(s) \leftarrow 1 \ \forall s \in \mathcal{S}^-$, $\hat{Z}(s) \leftarrow e^{\mathcal{R}(s)/\lambda} \ \forall s \in \mathcal{T}$, $\hat{\mathcal{P}}_{\mathbf{u}} \leftarrow \mathcal{P}$
repeat
    $s_t \leftarrow s_0$ (sample state from initial state distribution)
    while $s_t \notin \mathcal{T}$ **do**
        Take reward r_t from the current state s_t.
        $\mathcal{G}\left[\hat{z}\right](s_t) \leftarrow \sum_{s' \in \mathcal{S}} \mathcal{P}\left(s' \mid s\right) \hat{Z}(s')$
        $\hat{Z}(s_t) \leftarrow \hat{Z}(s_t) + \alpha \left[e^{r_t/\lambda} \mathcal{G}[\hat{z}](s_t) - \hat{Z}(s_t) \right]$
        Update $\hat{\mathcal{P}}_{\mathbf{u}}$ derived from $\hat{Z}$
        Sample a next state s_{t+1} according to $\hat{\mathcal{P}}_{\mathbf{u}}$
        $s_t \leftarrow s_{t+1}$
    end while
until convergence

Z-learning

Algorithm 13 Z-learning

input: temperature parameter $\lambda \in \mathbb{R}$, $c \in \mathbb{R}$, restart randomness parameter r_{s_0}, LMDP $\mathcal{L}$ with $\mathcal{R}$, $\mathcal{P}$, $\mathcal{S}$, $\mathcal{A}$, $\mathcal{S}^-$, $\mathcal{T}$, s_0
output: $Z : \mathcal{S} \to \mathbb{R}$
initialize $Z(s) \leftarrow 1$, for all $s \in \mathcal{S}^-$ and $Z(s) \leftarrow e^{\mathcal{R}(s)/\lambda}$ for all $s \in \mathcal{T}$, $\mathcal{P}_{\mathbf{u}^*} = \mathcal{P}$, $N = 0$
repeat
    if $r_{s_0} = 0$ **then**
        $s_t = s_0$
    else
        Sample a random state s_t from $\mathcal{S}^-$
    end if
    while $s_t \notin \mathcal{T}$ **do**
        Take reward r_t from current state s_t
        Sample a next state s_{t+1} according to $\mathcal{P}_{\mathbf{u}}$.
        $\mathcal{G}[z](s_t) = \sum_{s' \in \mathcal{S}} \mathcal{P}(s' \mid s_t) z(s')$
        $Z(s_t) \leftarrow Z(s_t) + \alpha \left[e^{r_t/\lambda} \mathcal{G}[z](s_t) - Z(s_t) \right]$
        Update $\mathcal{P}_{\mathbf{u}}$
        $s_t \leftarrow s_{t+1}$
    end while
    $N \leftarrow N + 1$
    $\alpha \leftarrow c/(c + N)$
until convergence
return Z

Value Iteration Optimization

Grid Size	Number of States	Loop Time (s)	Vectorized Time (s)
2	4	0.00042	0.00007
3	9	0.00303	0.00194
5	25	0.01124	0.00034
10	100	0.11045	0.00123
15	225	0.49348	0.00433
20	400	1.56553	0.01392
30	900	9.18308	0.09197
40	1600	35.22223	0.37331
50	2500	99.86756	1.07719
60	3600	262.92955	2.99847
70	4900	505.89836	5.49093
80	6400	985.84623	10.65779
90	8100	1842.35975	19.35463
100	10000	3058.16183	33.91510

Table 3: Value iteration execution times for different grid sizes.

Power Iteration Optimization

Grid Size	Number of States	Non-Sparse Time (s)	Sparse Time (s)
2	4	0.00439	0.00661
3	9	0.00079	0.00727
5	25	0.00206	0.00561
10	100	0.00660	0.01041
15	225	0.02661	0.01544
20	400	0.21025	0.02998
30	900	2.03092	0.05614
40	1600	13.63155	0.11217
50	2500	63.88911	0.22688
60	3600	218.97058	0.36187
70	4900	660.02274	0.62532
80	6400	1514.062	1.00146
90	8100	3533.87411	1.45586
100	10000	8723.56723	2.14425

Table 4: Power iteration execution times for different grid sizes.