AE598 HW1 Dynamic Programming Report

Mikihisa Yuasa
Department of Aerospace Engineering
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801, USA
Email: myuasa2@illinois.edu

1. Introduction

In this homework, five reinforcement learning (RL) algorithms, Policy Iteration, Value Iteration, TD(0), State–action–reward–state–action (SARSA), and Q-Learning, are implemented for two environments, Grid World and Discrete Pendulum. In the following section, the resulted plots of the algorithms are discussed.

2. Methodology

For the policy iteration and value iteration algorithms, the hyperparameters are chosen as $\gamma = 0.95$ and $\theta = 1 \times 10^{-3}$.

For the SARSA and Q-learning algorithms, the hyperparameters are chose as $\alpha = 0.5, \gamma = 0.95$, and $\epsilon = 0.1$ for the preliminary learning curve plot and example trajectory, policy, and state-value function plots. Additional hyperparameter values $\alpha = \{0.7, 0.5, 0.3\}$ and $\epsilon = \{0.15, 0.1, 0.05\}$ are chosen to observe the difference in learning curves due to the choice of the hyperparameters. The numbers of episodes for the grid world and pendulum environments are chosen as 5000 and 200, respectively.

3. Results

3.1. Plot of the learning curve for each algorithm

Figs. 1 to 4 represent the learning curve of each algorithm for the grid world environment. For the model-based algorithms, the mean value of the returns of each iteration of evaluation and improvement is plotted. Fig. 1 shows a convergence after the third iteration while Fig. 2 shows a clear convergence after the 15th iteration.

For the model-free algorithms, the episodic return is plotted as the learning curve in Figs. 3 and 4. The plots are smoothed by moving average whose window size is 100. Fig. 3 shows convergence after approximately 1,900 episodes while Fig. 4 converges after 2,100 episodes.

Figs. 5 and 6 represent the learning curves of SARSA and Q-learning algorithms for the pendulum environment. Both of the plots in Figs. 5 and 6 are smoothed by moving average with the window size of 5. Fig. 5 shows convergence after 25 episodes while Fig. 6 shows convergence after 40 episodes.

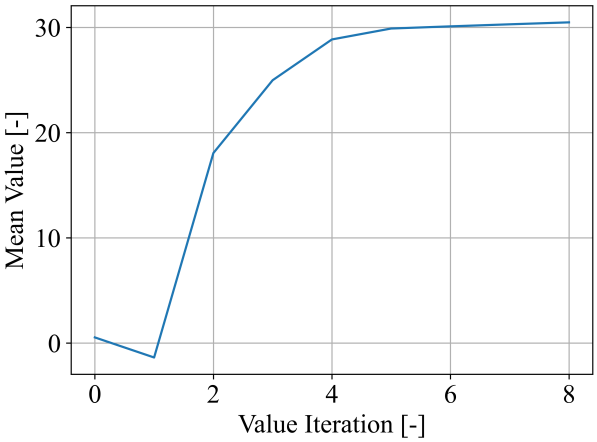


Figure 1: Learning curve of the policy iteration algorithm.

3.2. Plot of learning curves for several different α and ϵ values

Fig. 7 and Fig. 8 represent the learning curves of the SARSA and Q-learning algorithms for the grid world environment for $\alpha = \{0.7, 0.5, 0.3\}$ and $\epsilon = \{0.15, 0.1, 0.05\}$.

Fig. 9 and Fig. 10 represent the learning curves of the SARSA and Q-learning algorithms for the pendulum environment for $\alpha = \{0.7, 0.5, 0.3\}$ and $\epsilon = \{0.15, 0.1, 0.05\}$.

3.3. Example trajectories of the trained agents

Figs. 11 to 14 represent the example trajectories of the learned agents of each algorithm for the grid world environment.

Figs. 15 and 16 represent the example trajectories of the learned agents of each algorithm for the pendulum environment.

3.4. State-value functions and policies of the trained agents

Figs. 17 to 20 represent the state-value function and policy of the learned agents of each algorithm for the grid

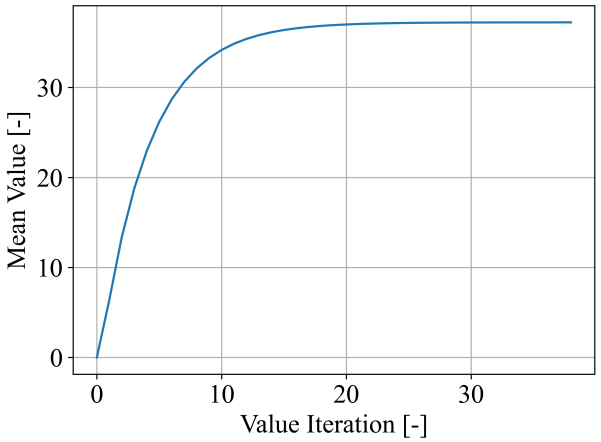


Figure 2: Learning curve of the value iteration algorithm

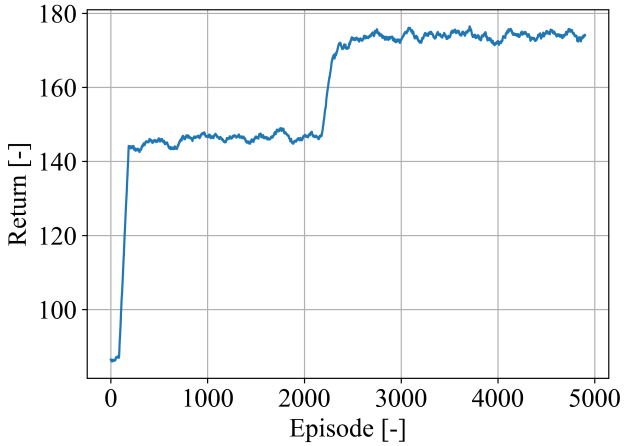


Figure 4: Learning curve of the Q-learning algorithm

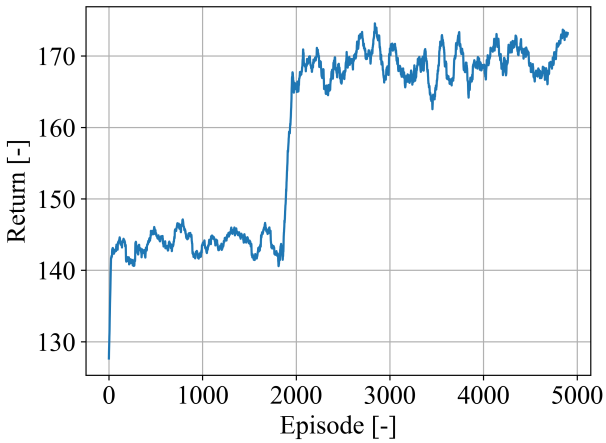


Figure 3: Learning curve of the SARSA algorithm

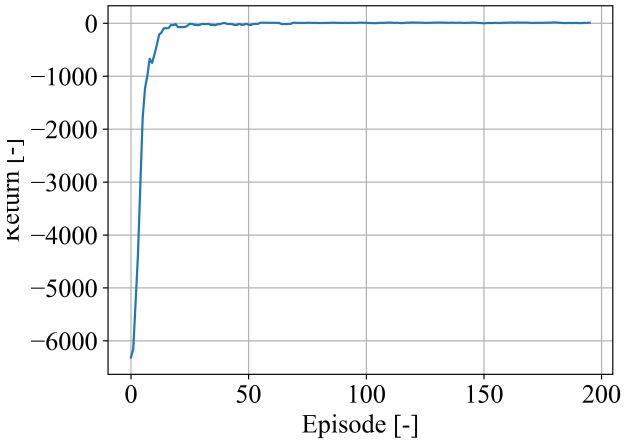


Figure 5: Learning curve of the SARSA algorithm

world environment.

Figs. 21 and 22 represent the state-value function and policy of the learned agents of each algorithm for the pendulum environment.

4. Discussion

Figs. 1 to 6 show that the learning curves of the implemented algorithms converge and thus imply that the algorithms are capable of learning both of the environments as intended.

For the grid world environment, both of the model-based methods learned similar policies according to Figs. 17 and 18, though, the state-value functions have different peaks while those of the model-free methods in Figs. 19 and 20 are consistent with that of value iteration. This implies that the policy iteration agent stops learning before it fully learns the value function. Nonetheless, from Figs. 11 to 14, it is observed that the agents learn the optimal policy to enter and exit the area A back and forth to maximize the reward. For the pendulum environment, both of the agents learned policies that result in similar state-value function based on Figs. 21 and 22, and the sample trajectories in Figs. 15 and 16 show that the agents successfully avoid the large penalty due to exceeding the maximum $\dot{\theta}$.

Figs. 7 to 10 demonstrate the influence of the hyper-parameter choice on the agents' convergence speed and final convergence value. For the grid world environment, $(\alpha, \epsilon) = (0.5, 0.05)$ results in the highest return above 180 for SARSA while $(\alpha, \epsilon) = (0.7, 0.1)$ results in the highest return approximately 170 and the fastest convergence after around 1000 episodes based on Figs. 7 and 8. On the other hand, for the pendulum environment, Figs. 9 and 10 show that $(\alpha, \epsilon) = (0.5, 0.1)$ has the fastest convergence around episode 25 for the SARSA agent while $(\alpha, \epsilon) = (0.7, 0.05)$ has the fastest convergence around episode 20 for the Q-

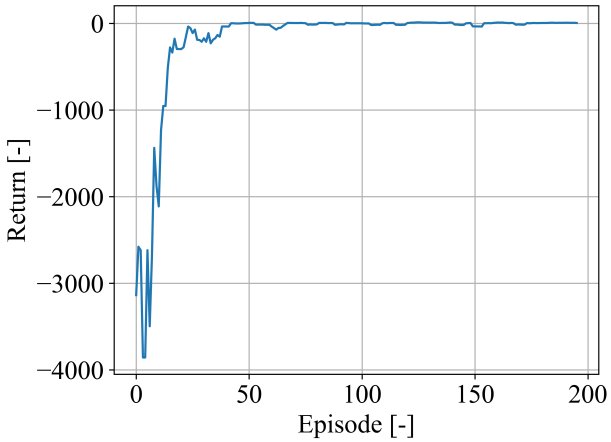
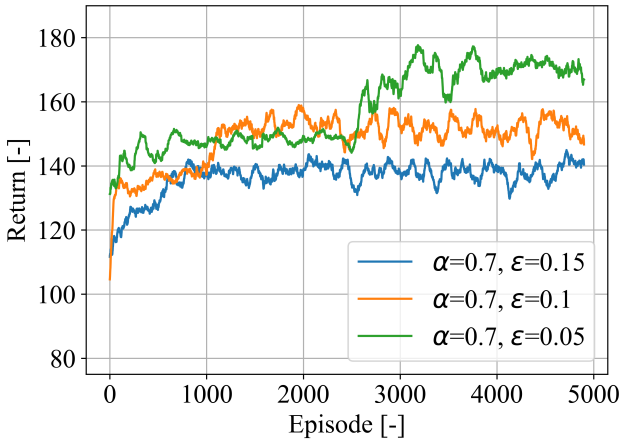
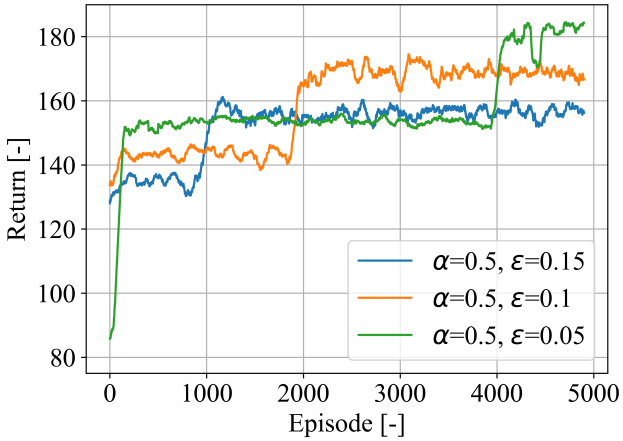


Figure 6: Learning curve of the Q-learning algorithm

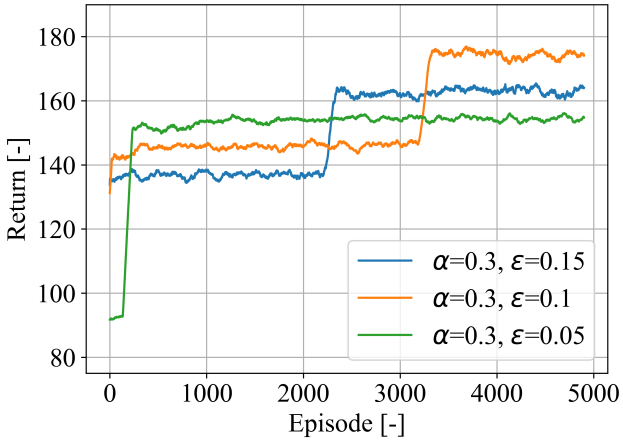
learning agent. Therefore, the choice of the hyperparameters both depend on algorithms and problems or environment that an RL agent learns.



(a) $\alpha = 0.7, \epsilon = \{0.15, 0.1, 0.05\}$.

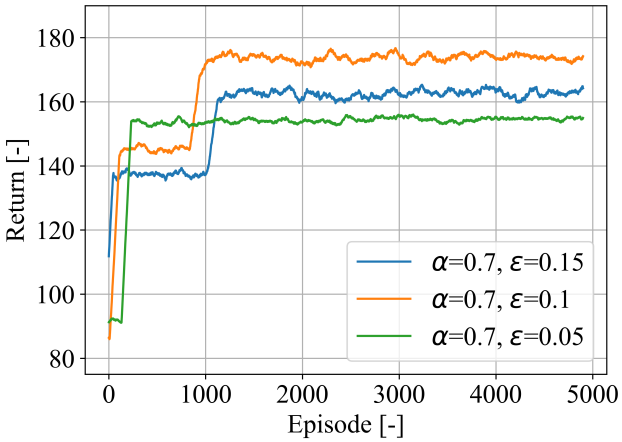


(b) $\alpha = 0.5, \epsilon = \{0.15, 0.1, 0.05\}$.

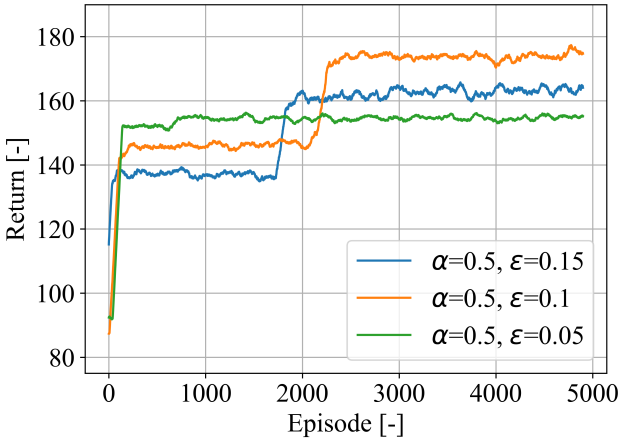


(c) $\alpha = 0.3, \epsilon = \{0.15, 0.1, 0.05\}$.

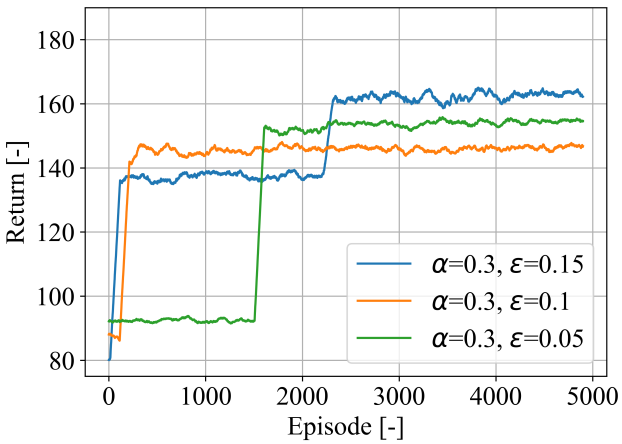
Figure 7: Learning curves of the SARSA algorithm for $\alpha = \{0.7, 0.5, 0.3\}$ and $\epsilon = \{0.15, 0.1, 0.05\}$ for the grid world environment.



(a) $\alpha = 0.7, \epsilon = \{0.15, 0.1, 0.05\}$.

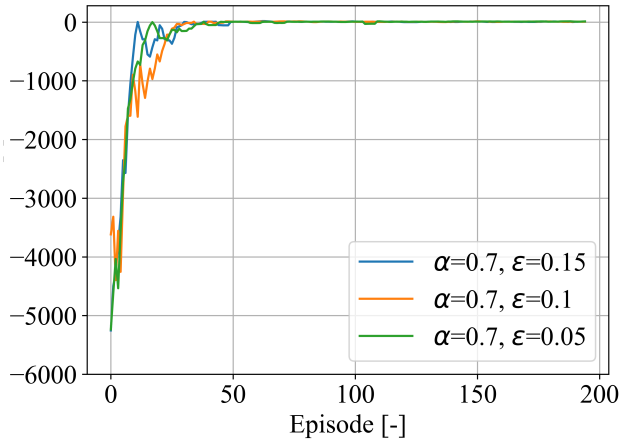


(b) $\alpha = 0.5, \epsilon = \{0.15, 0.1, 0.05\}$.

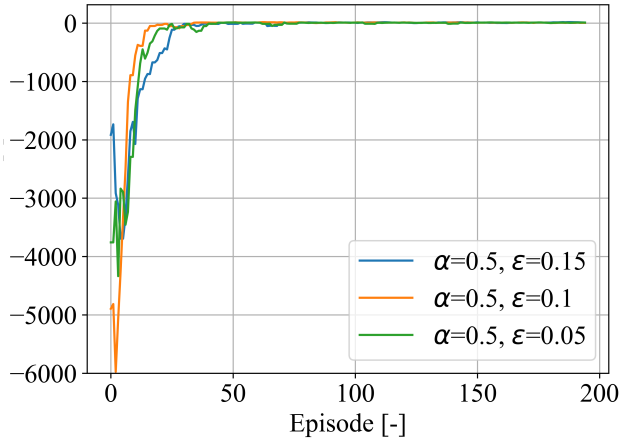


(c) $\alpha = 0.3, \epsilon = \{0.15, 0.1, 0.05\}$.

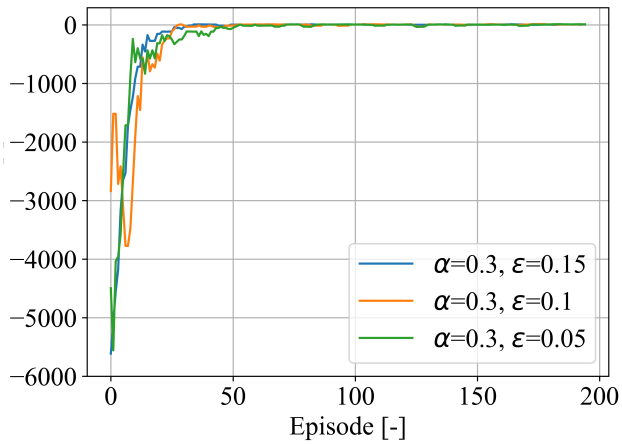
Figure 8: Learning curves of the Q-learning algorithm for $\alpha = \{0.7, 0.5, 0.3\}$ and $\epsilon = \{0.15, 0.1, 0.05\}$ for the grid world environment.



(a) $\alpha = 0.7, \epsilon = \{0.15, 0.1, 0.05\}$.

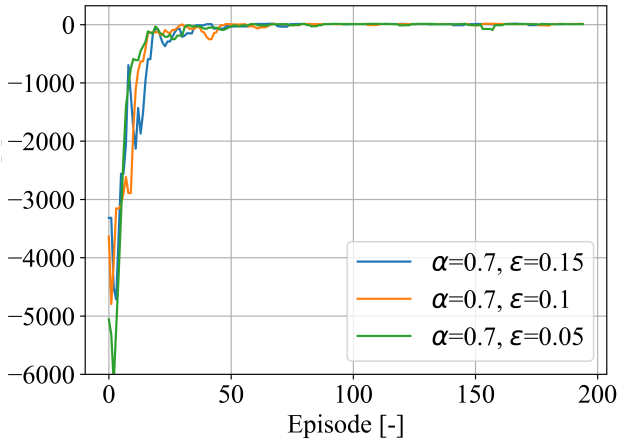


(b) $\alpha = 0.5, \epsilon = \{0.15, 0.1, 0.05\}$.

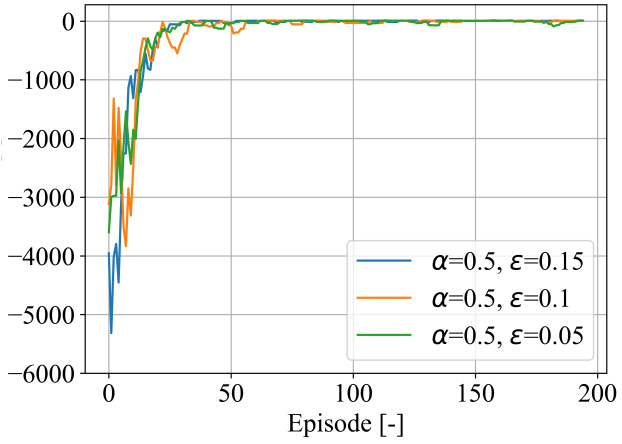


(c) $\alpha = 0.3, \epsilon = \{0.15, 0.1, 0.05\}$.

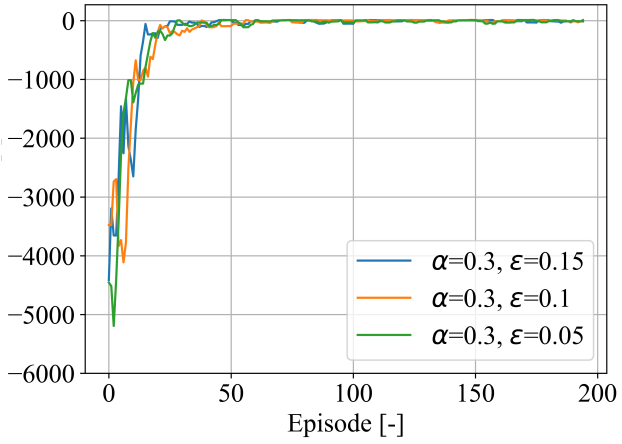
Figure 9: Learning curves of the SARSA algorithm for $\alpha = \{0.7, 0.5, 0.3\}$ and $\epsilon = \{0.15, 0.1, 0.05\}$ for the pendulum environment.



(a) $\alpha = 0.7, \epsilon = \{0.15, 0.1, 0.05\}$.



(b) $\alpha = 0.5, \epsilon = \{0.15, 0.1, 0.05\}$.



(c) $\alpha = 0.3, \epsilon = \{0.15, 0.1, 0.05\}$.

Figure 10: Learning curves of the Q-learning algorithm for $\alpha = \{0.7, 0.5, 0.3\}$ and $\epsilon = \{0.15, 0.1, 0.05\}$ for the pendulum environment.

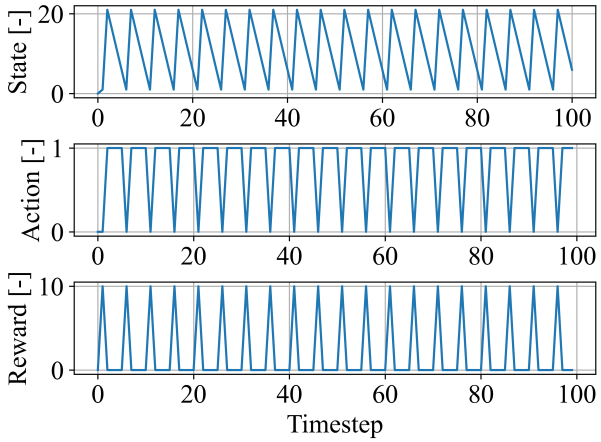


Figure 11: Example trajectory for a learned policy iteration agent for the grid world environment.

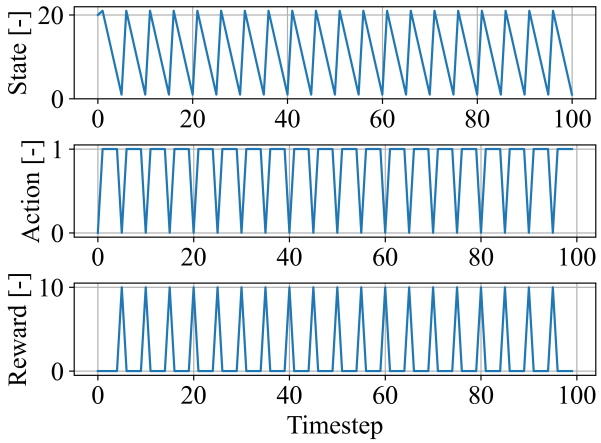


Figure 12: Example trajectory for a learned value iteration agent for the grid world environment.

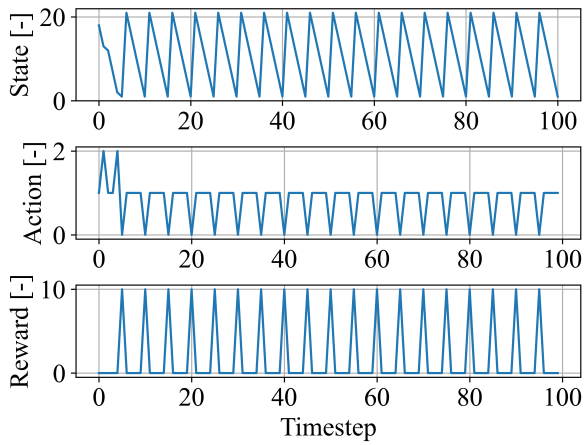


Figure 13: Example trajectory for a learned SARSA agent for the grid world environment.

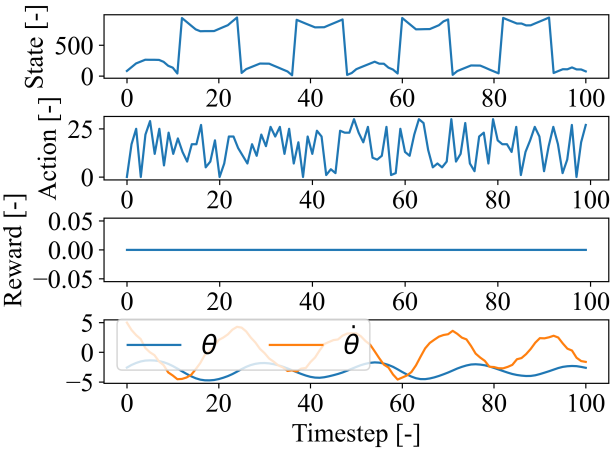


Figure 15: Example trajectory for a learned SARSA agent for the pendulum environment.

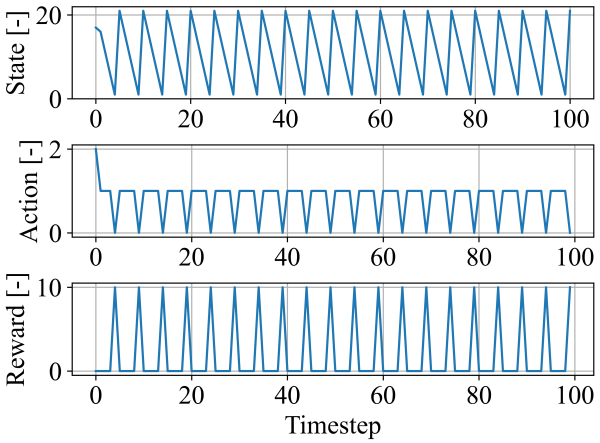


Figure 14: Example trajectory for a learned Q-learning agent for the grid world environment.

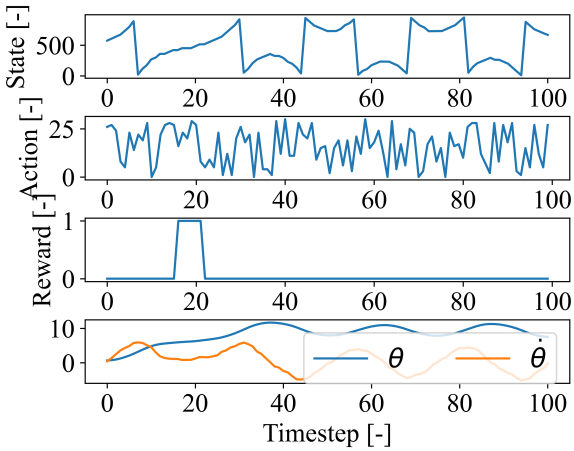


Figure 16: Example trajectory for a learned Q-learning agent for the pendulum environment.

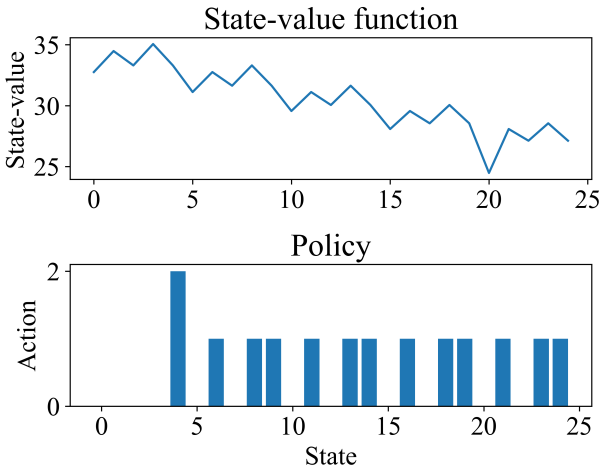


Figure 17: State-value function and policy of a learned policy iteration agent for the grid world environment.

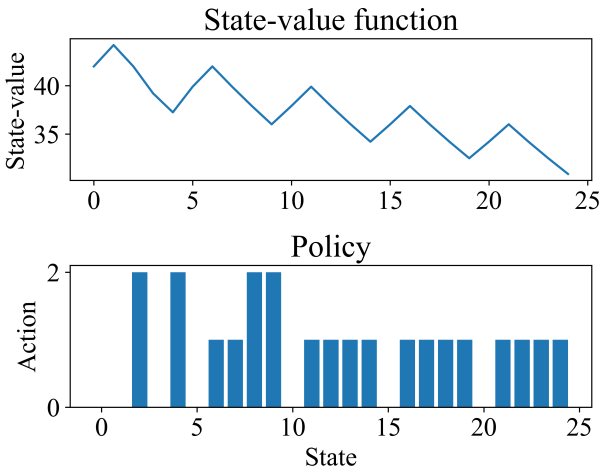


Figure 18: State-value function and policy of a learned value iteration agent for the grid world environment.

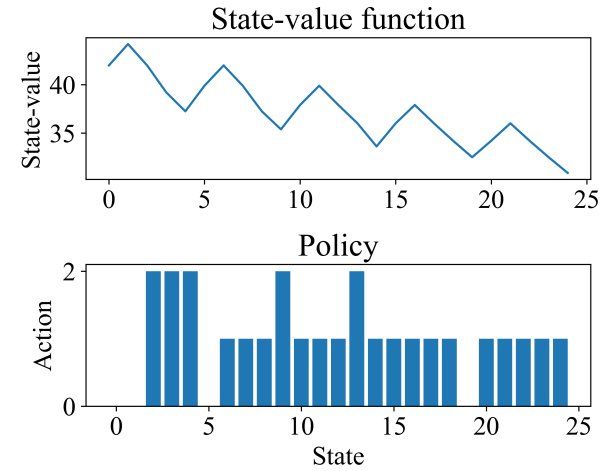


Figure 19: State-value function and policy of a learned SARSA agent for the grid world environment.

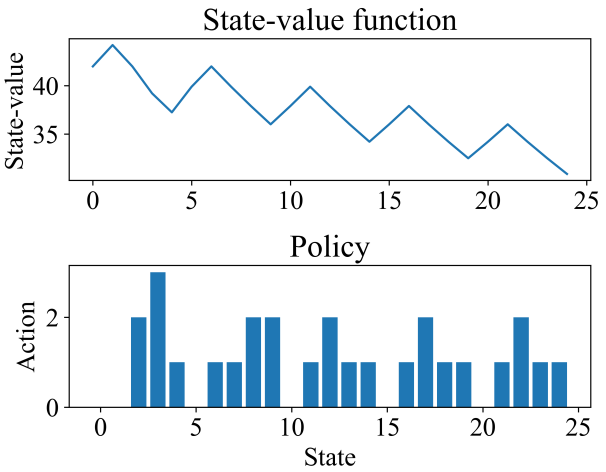


Figure 20: State-value function and policy of a learned Q-learning agent for the grid world environment.

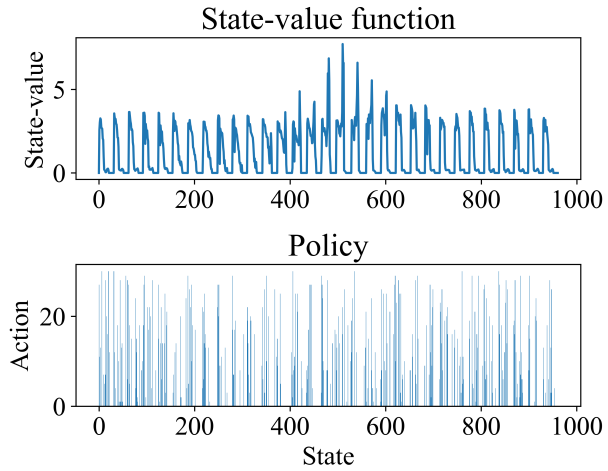


Figure 21: State-value function and policy of a learned SARSA agent for the pendulum environment.

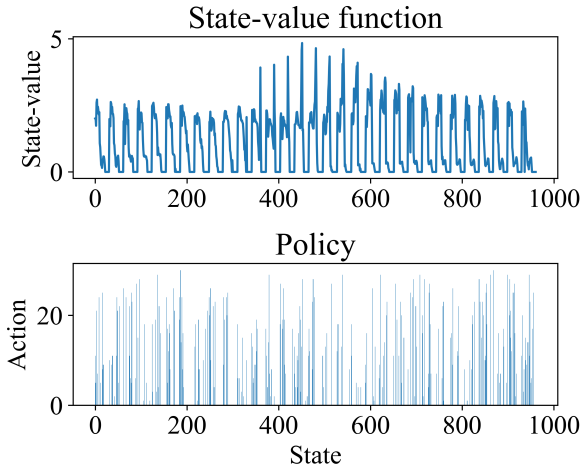


Figure 22: State-value function and policy of a learned Q-learning agent for the pendulum environment.