Imperial College
London

REINFORCEMENT LEARNING

-

COURSEWORK 1
MAZE ENVIRONMENT

Name: Jules Viard

Email: jcv23@ic.ac.uk

CID: 02461091

Date: 18/10/2023 – 3/11/2023

Question I: Dynamic Programming

I.1. Method and parameters selection.

In this section, given our complete knowledge of the MDP and the size of the problem (small size: 13 by 10 grid), we can employ a Dynamic Programming approach.

For this problem, we selected the Value Iteration method. In deciding between Policy Iteration and Value Iteration, we compared their convergence in terms of the number of epochs for a specific set of parameters (discussed below). Based on our result, Value Iteration (70 epochs) required almost ten times fewer epochs than Policy Iteration (670 epochs).

For this model, we chose the parameter "threshold", which is the stopping condition regarding the convergence of the value function. We selected a threshold of 0.0001, as a trade-off between the precision of the value function and computational time. Consequently, we operate under the assumption that this threshold can be attained, ideally with a reasonable number of iterations.

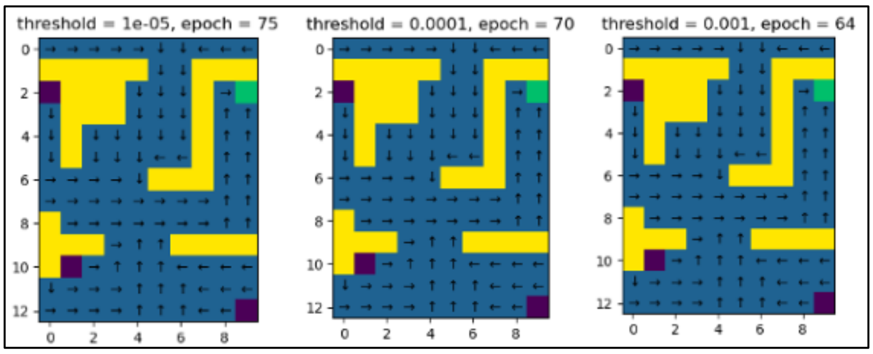


Figure 1: Impact of the threshold range from 1e-5 to 1e-3 on the policy.

Upon testing threshold values ranging from 1e-5 to 1e-3, we found that the policy remains the same, with only minor variations in the number of epochs (in Figure 1).

Moreover, we initialised the policy with action 0: for every state, the policy indicates to go north. As illustrated in Figure 2 this initialisation does not impact the result of our algorithm.

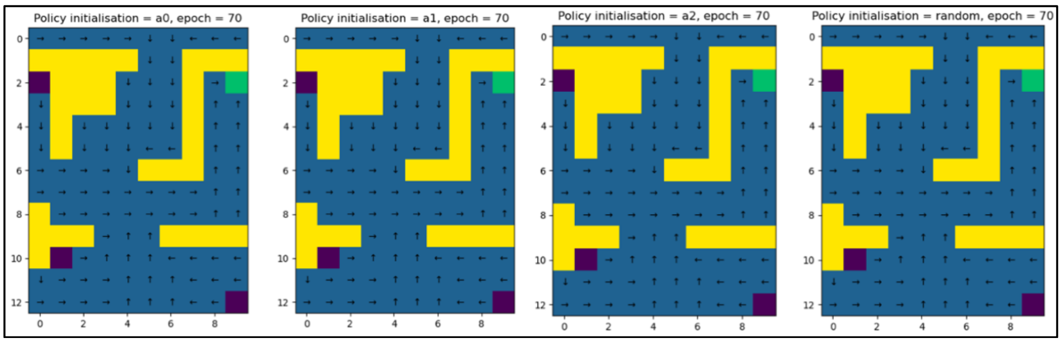


Figure 2: Impact of policy initialization on the final policy.

I.2. Optimal Policy and Value function.

The optimal policy and value function (respectively Figure 3 and Figure 4) are computed for the following parameters: $p = 0.8$ and $\gamma = 0.98$.

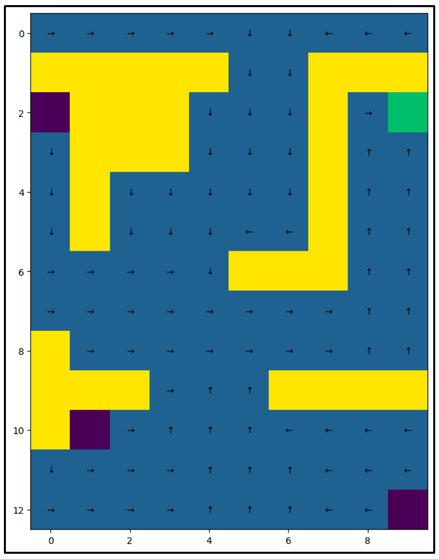


Figure 3: Optimal policy for value iteration in Maze.

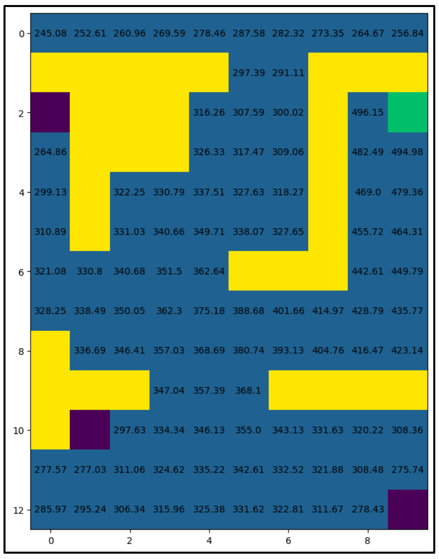


Figure 4: Optimal value function.

I.3. Investigation of discount factor and success probability.

To investigate the impact of setting $p < 0.25$, we selected $p = 0.2$. In this case, choosing the best action is less likely than selecting an action that minimizes the value function. The policy, displayed in Figure 5, is the "opposite" of the optimal policy, shown in Figure 3.
In the case of $p = 0.25$, each action has the same probability of being taken. The transition matrix becomes independent of the actions. As a result, since the best action index (argmax) will select the first action when all actions yield the same values (for p=0.25), the policy indicates "go north" in all states (Figure 6).
When $p > 0.25$ (Figure 7), the best action has the highest probability of being taken. Consequently, we obtain the right policy with a decreasing number of epochs.

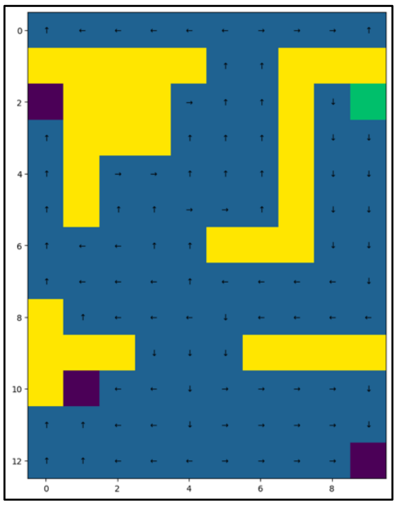


Figure 5: Policy for p = 0.2.

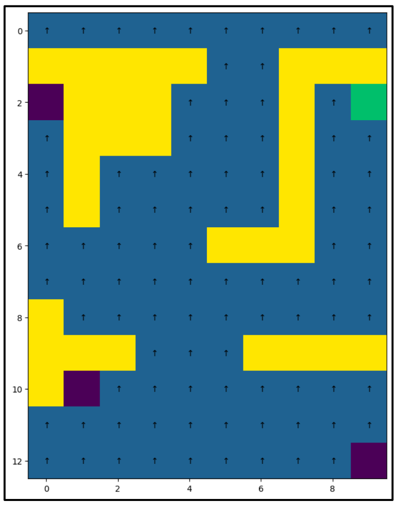


Figure 6: Policy for p = 0.25.

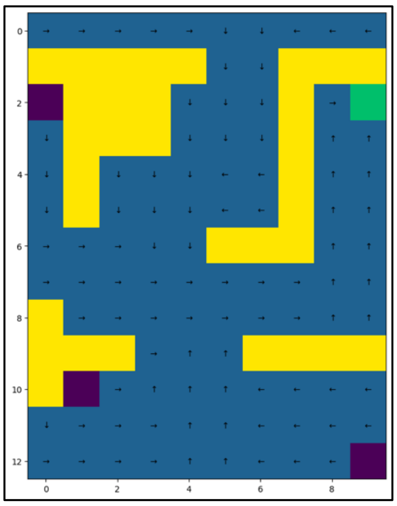


Figure 7: Policy for p = 0.4.

When $\gamma < 0.5$, immediate rewards are privileged, leading to the value function of two neighbouring states appearing nearly identical (in Figure 8). On the other hand, for $\gamma > 0.5$, the subsequent rewards start to hold a significant weight. With an increasing discounted factor, further rewards gradually equalize in significance, and the value function of two neighbouring states become more different.

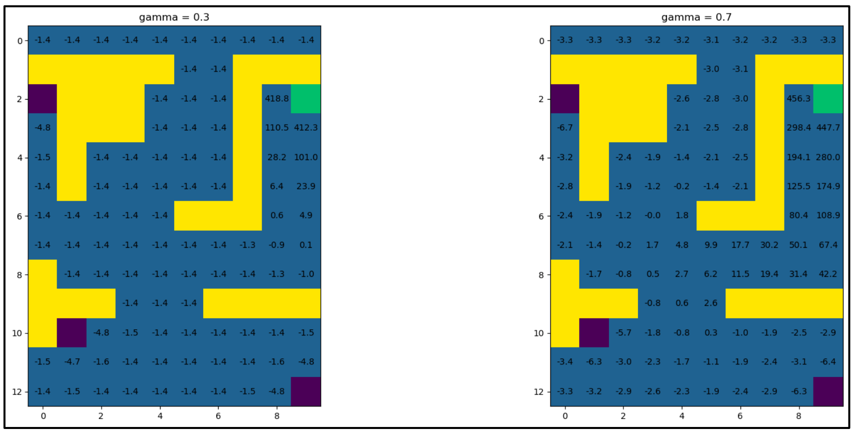


Figure 8: Impact of ɣ on value function for ɣ = 0.3 and ɣ = 0.7.

We tend towards an optimal policy as the discounted factor increases (Figure 9). However, it should be noted that the number of epochs is also increasing.

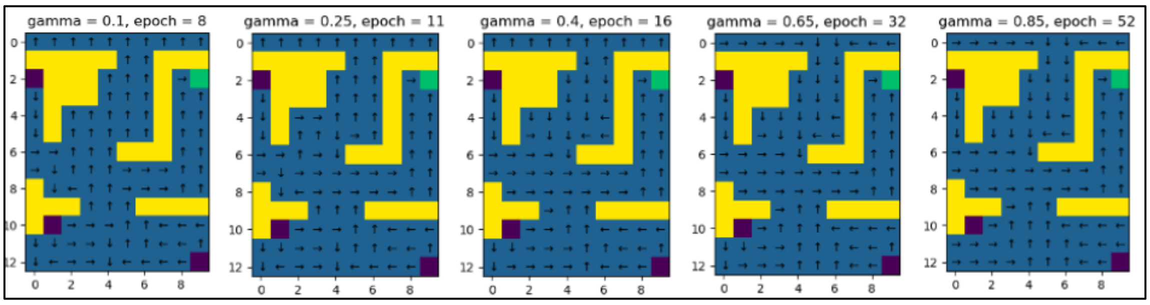


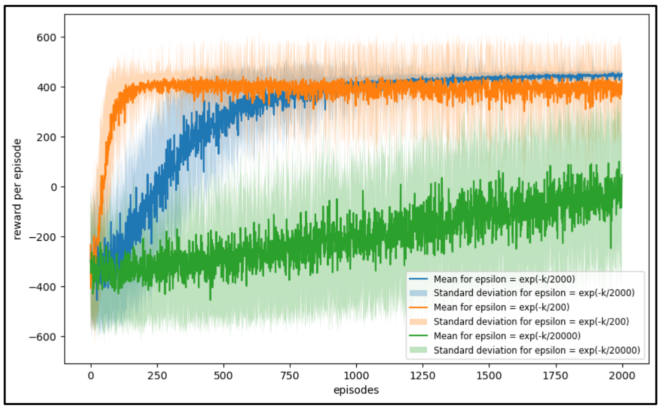
Figure 9: Impact of gamma on the policy.

<u>Question II: Monte-Carlo.</u>

II.1. Method and parameters selection.

We have chosen to implement the online Monte-Carlo First Visit with an ε-greedy policy. Our focus on the first visit aims to have a more straightforward update process. Given that the world is stationary, each episode provides consistent information, allowing for faster convergence through online learning.

The parameter epsilon is set to decay exponentially (Figure 11) in relation to the episode index k: $\varepsilon_k = e^{-k/2000}$. This setting is a trade-off between the number of episodes required for converge and the performance of the method, as compared in Figure 10.
Thus, we obtain an exploratory behaviour initially, followed by convergence with a reasonable variance (Figure 11). The GLIE condition is also satisfied.
We assume that 3000 episodes are sufficient to achieve convergence. Empirically, we observed stable behaviour with 2000 episodes. For our experiments, we used 25 training runs.



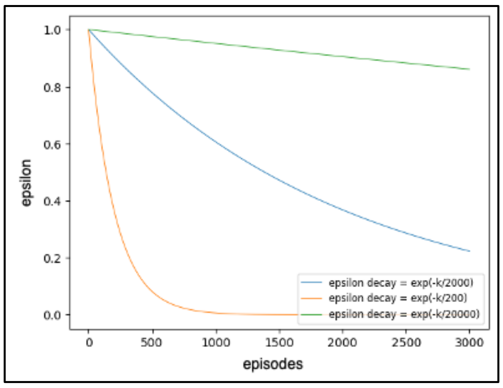
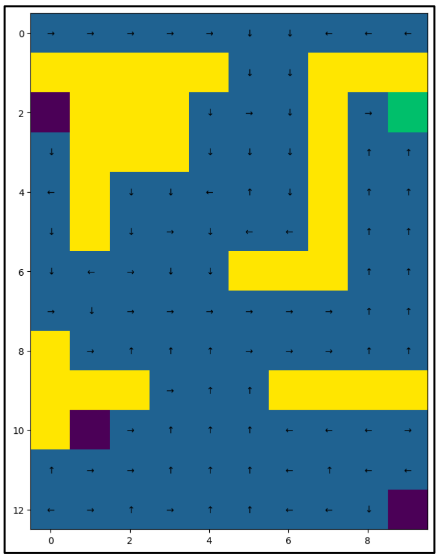


Figure 10: Impact of epsilon on the model's convergence.

Figure 11: Exponential decaying of epsilon.

II.2. Optimal Policy and Value function.

The optimal policy and value function are computed, using the parameters of section II.1.



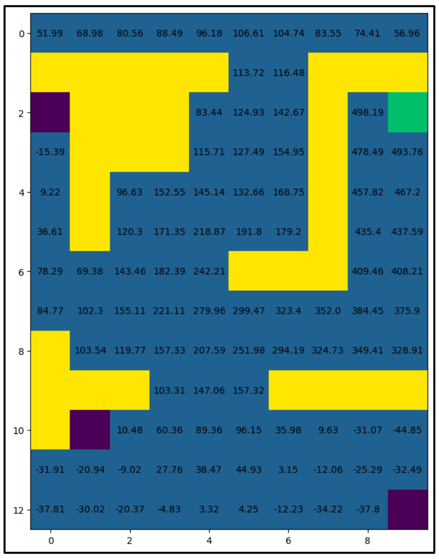


Figure 12: Optimal policy for Monte-Carlo.

Figure 13: Optimal Value function for Monte-Carlo.

II.3. Learning curve of the agent.

Figure 14 displays the learning curve of the agent with $\varepsilon_k = e^{-k/2000}$ across 25 training runs.

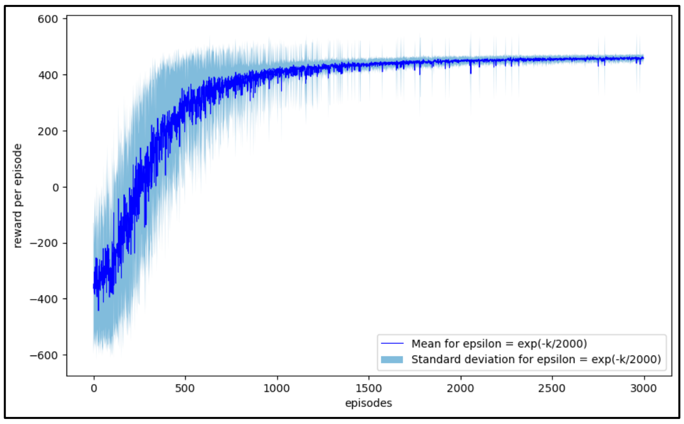


Figure 14: Mean and standard deviation of the learning curve.

Question III: Temporal Difference

III.1. Method and parameters selection.

For this part, we selected the Q-Learning method. Indeed, Q-Learning is inclined to choose the optimal path, as the model may take risks. In most experiments, Q-learning converges slightly more quickly than SARSA method, as illustrated in Figure 15.

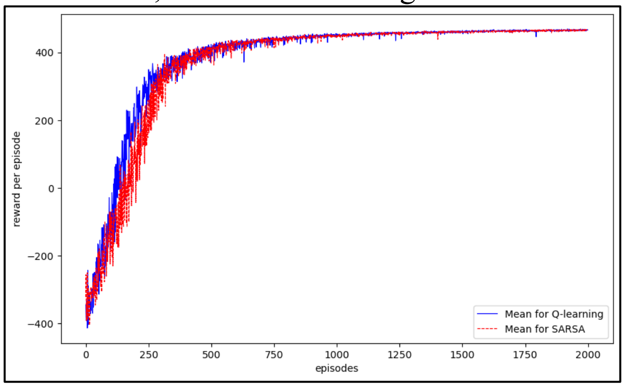


Figure 15: Comparison of convergence between Q-learning and SARSA methods across 30 training runs.

In this model, we have three parameters: the number of episodes, learning rate (α), and epsilon (ε-greedy policy).
Epsilon is set to decay exponentially, facilitating an exploratory phase and convergence with a low variance. The exponential decay formula is designed to obtain an ε value close to zero by the end of 2000 episode: $\varepsilon_k = e^{-k/1000}$ (where k is the index of the episode). The GLIE condition is satisfied.
Given that the world is stationary (without noise), we opted for a low learning rate. Empirically, we set the learning rate at 0.25. Thus, we assign a moderate weight to the new Q-value estimates.
We assume that 2000 episodes are sufficient for the agent to converge. Since we do not know the terminal states, we initialise the Q matrix with zero values. For comparative purpose, 30 training runs are done.

III.2. Optimal Policy and Value function.

The optimal policy and value function are computed, using the parameters of section III.1.

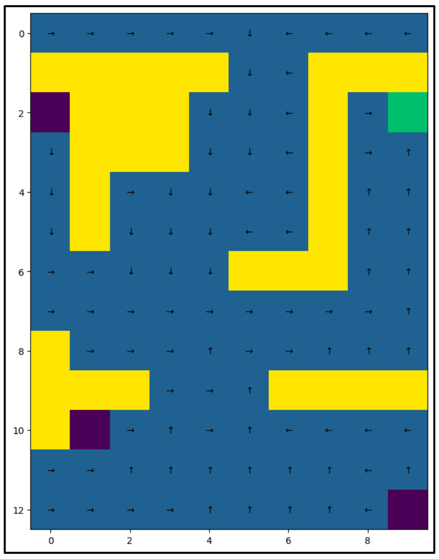


Figure 16: Optimal Policy for Q-Learning.

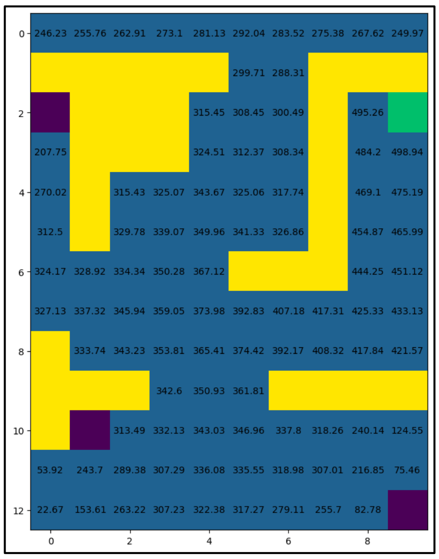


Figure 17: Optimal Value function for Q-Learning.

III.3. Optimal Policy and Value function.

For a low epsilon (for instance, ε=0.1), the agent will use its knowledge and avoid taking random action. Consequently, the model converges quickly but this may lead to a sub-optimal solution. For epsilon around 0.5, the model is balanced between using its knowledge and exploring the environment. When epsilon is high, the agent is more likely to take random actions. Thus, the convergence is very slow, and the associated variance is high. The comparison is displayed in Figure 18.

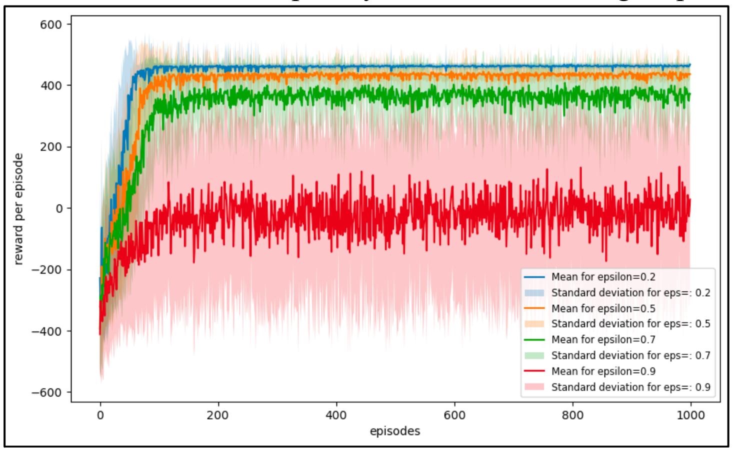


Figure 18: Impact of the epsilon value on the learning curve.

The learning rate (α) is associated with the update of the Q-value and influences the weighting of the next action. As α approaches 0, the significance of the next action is minimised, resulting in a stable agent, with low variance. When α approaches 1, convergence is slightly faster, but with a high variance, as illustrated in Figure 20.

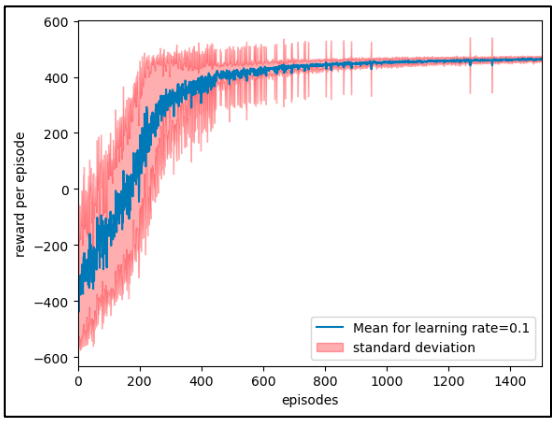


Figure 19: Learning curve for α = 0.1.

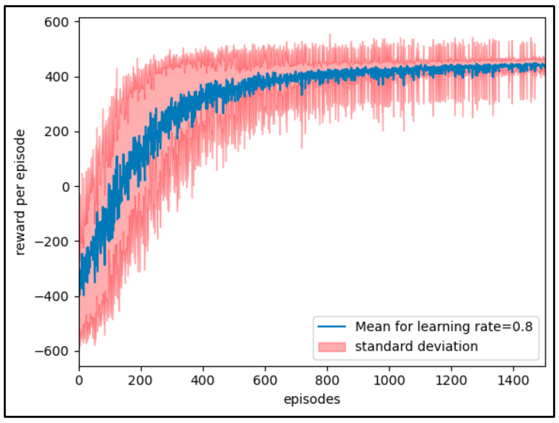


Figure 20: Learning curve for α = 0.8.

Using a decaying learning rate can be an interesting setting. This strategy allows a rapid initial learning phase, followed by the emergence of a stable agent in later stages.

In Figure 21, the learning rate decays according to the formula: $\alpha_k = e^{-k/1000}$ (where k is the episode index). This setting enables faster convergence compared to a low learning rate (for instance α=0.1). As the number of episodes increases, the variance decreases.

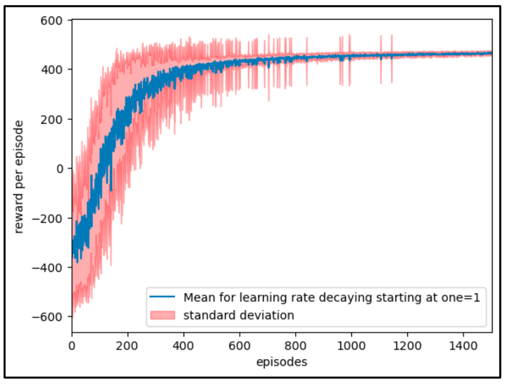


Figure 21: Impact of a decaying α on the learning curve.