IMEN764 **Dynamic Programming**
Fall 2017
Instructor: Prof. Dong Gu Choi

Homework #4: Programming an MDP

Yongkyu Cho (20130712)
jyg1124@postech.ac.kr

0. MDP Formulation

- S: state space

$$S=\{\textbf{S}\text{tanding, }\textbf{M}\text{oving, }\textbf{F}\text{allen}\}$$

- A_s: action space given a state $s \in \mathbb{S}$

$$A_s = \begin{cases} \{\text{Slow, Fast}\} & \text{if } s \text{ is } \textbf{S}\text{tanding or } \textbf{M}\text{oving} \\ \{\text{Slow}\} & \text{if } s \text{ is } \textbf{F}\text{allen,} \end{cases}$$

- D: set of decision rules
- P_d: stochastic matrix with components $p_d(s'|s)$ given a decision rule $d \in D$
- $g(s,a)$: the expected reward of taking an action 'a' given the current state is 's'

$$g(s,a) = \begin{cases} +1 & \text{if } s \text{ is } \textbf{S}\text{tanding and } a \text{ is Slow} \\ +0.8 & \text{if } s \text{ is } \textbf{S}\text{tanding and } a \text{ is Fast} \\ +1 & \text{if } s \text{ is } \textbf{M}\text{oving and } a \text{ is Slow} \\ +1.4 & \text{if } s \text{ is } \textbf{M}\text{oving and } a \text{ is Fast} \\ -0.2 & \text{if } s \text{ is } \textbf{F}\text{allen and } a \text{ is Slow} \end{cases}$$

1. Policy Iteration

(a) Possible Stationary Policies

We have $2 \times 2 \times 1 = 4$ stationary policies as written below:

- $d^{\infty 1} = [d(\mathbf{S}) = \text{Slow}, d(\mathbf{M}) = \text{Slow}, d(\mathbf{F}) = \text{Slow}]$
- $d^{\infty 2} = [d(\mathbf{S}) = \text{Slow}, d(\mathbf{M}) = \text{Fast}, d(\mathbf{F}) = \text{Slow}]$
- $d^{\infty 3} = [d(\mathbf{S}) = \text{Fast}, d(\mathbf{M}) = \text{Slow}, d(\mathbf{F}) = \text{Slow}]$
- $d^{\infty 4} = [d(\mathbf{S}) = \text{Fast}, d(\mathbf{M}) = \text{Fast}, d(\mathbf{F}) = \text{Slow}]$

(b) Effect of Discount Factor

We conduct an experiment to find the effect of discount factor on policy iteration. As can be seen in Figure 1, value function increases exponentially as the discount factor goes to 1. The number of iteration is 2 until the discount factor is less than 0.8x and becomes 1 after that. The optimal policy that is found from policy iteration algorithm is $d^{\infty 2}$ when $\delta < 0.8$x and $d^{\infty 1}$ when $\delta > 0.8$x. Therefore, the optimal policy differs for $\delta = 0.7$ and $\delta = 0.9$.

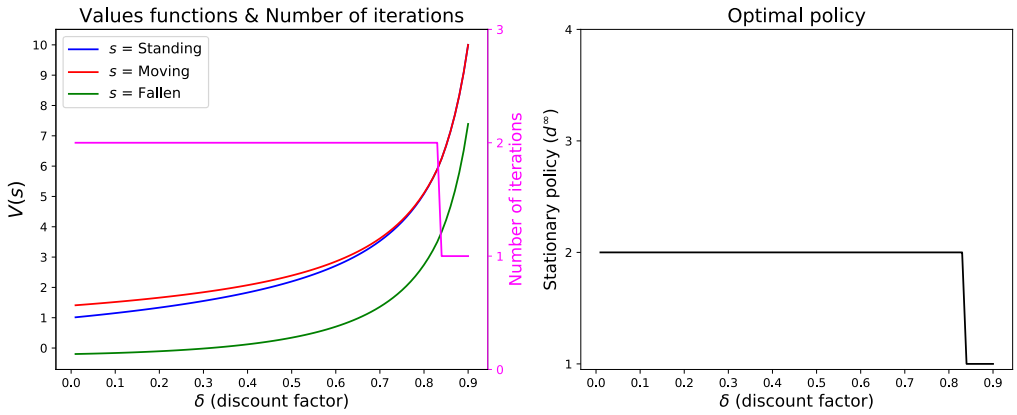


Figure 1: Effect of discount factor on *policy iteration*

(c) Optimal policy and its values for $\delta = 0.7, 0.9$

- $\delta = 0.9$:

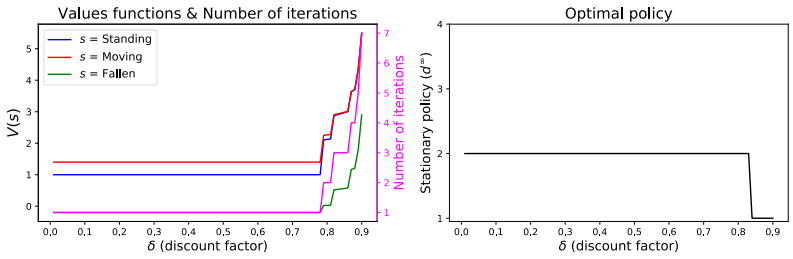
$$d^{\infty *} = d^{\infty 1} = [d(\mathbf{S}) = \text{Slow}, d(\mathbf{M}) = \text{Slow}, d(\mathbf{F}) = \text{Slow}],\ V = \begin{bmatrix} 10.00 \\ 10.00 \\ 7.39 \end{bmatrix}$$

- $\delta = 0.7$:

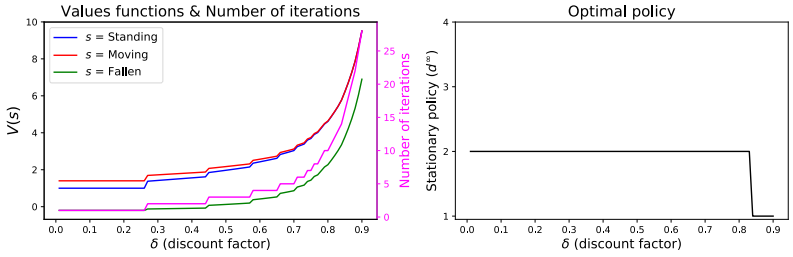
$$d^{\infty *} = d^{\infty 2} = [d(\mathbf{S}) = \text{Slow}, d(\mathbf{M}) = \text{Fast}, d(\mathbf{F}) = \text{Slow}],\ V = \begin{bmatrix} 3.53 \\ 3.61 \\ 1.40 \end{bmatrix}$$

2. Value Iteration

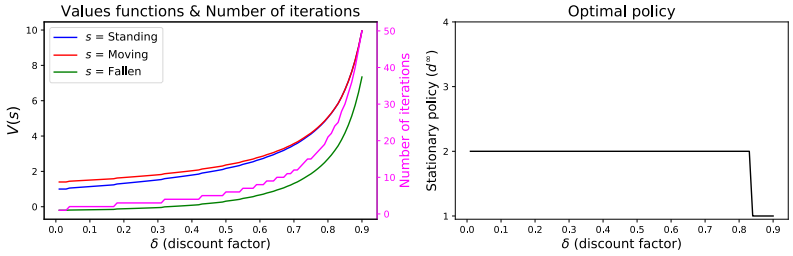
(a) Effect of Error Tolerance on the Number of Iterations
We conduct several experiments to find the effect of error tolerance on the number of iterations for value iteration. Figure 2 shows the results. The smaller the ϵ value is, the higher the number of iterations increases. The optimal policy is the same for all $\epsilon = 10.0, 1.0, 0.1, 0.01$.



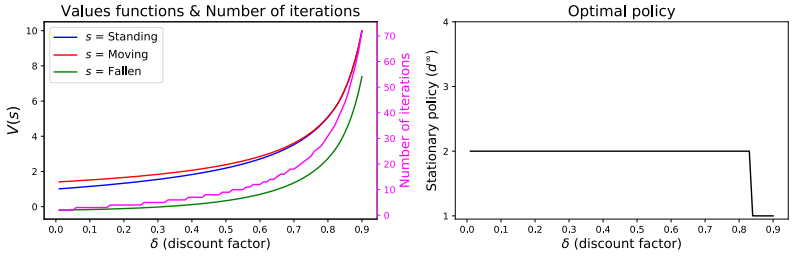
(a) $\epsilon = 10.0$



(b) $\epsilon = 1.0$



(c) $\epsilon = 0.1$



(d) $\epsilon = 0.01$

Figure 2: *Value iteration* with various discount factors under the four error tolerances ($\epsilon = 10.0, 1.0, 0.1, 0.01$)

(b) Optimal Policy and its Values for $\delta = 0.7, 0.9$ ($\epsilon = 0.01$)

- $\delta = 0.9$:

$$d^{\infty *} = d^{\infty 1} = [d(\mathbf{S}) = \text{Slow}, d(\mathbf{M}) = \text{Slow}, d(\mathbf{F}) = \text{Slow}],\ V \approx \begin{bmatrix} 10.00 \\ 10.00 \\ 7.39 \end{bmatrix}$$

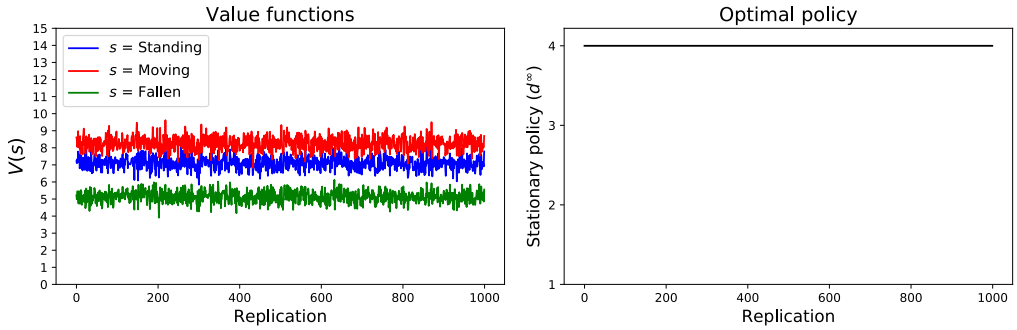
- $\delta = 0.7$:

$$d^{\infty *} = d^{\infty 2} = [d(\mathbf{S}) = \text{Slow}, d(\mathbf{M}) = \text{Fast}, d(\mathbf{F}) = \text{Slow}],\ V \approx \begin{bmatrix} 3.53 \\ 3.61 \\ 1.35 \end{bmatrix}$$

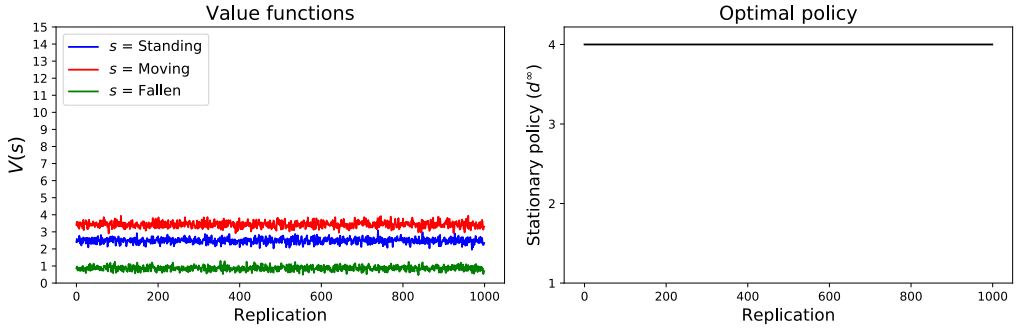
3. Q-Learning without Exploration

(a) Convergence

We run the 10000-iteration algorithm for 1000 replications and record the value funtion and optimal policy (Figure 3). As a result of a number of experiments, Q-learning without exploration does not converge well and the values of V highly fluctuate. Moreover, it always returns $d^{\infty 4}$ as the optimal policy, which is not true.



(a) $\delta = 0.9$

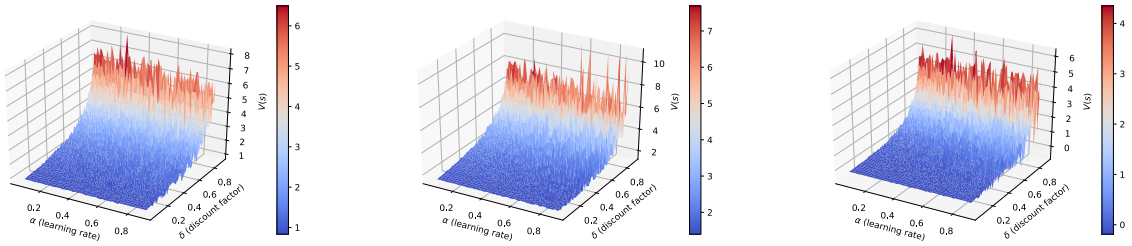


(b) $\delta = 0.7$

Figure 3: 1000 replications of *Q-learning without exploration* under two discount factors ($\alpha = 0.05$)

(b) Effects of α and δ on the Estimated Value Function

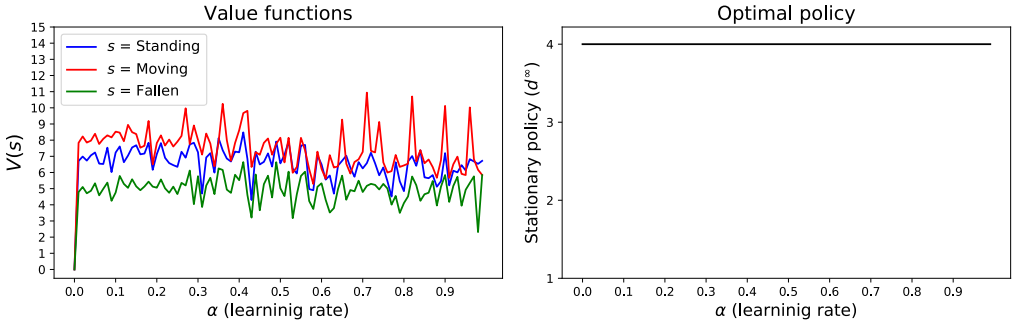
We conduct experiments with various α and δ values under the maximum iteration 10000. Figure 4 shows the results. Overall, high δ value increases the values of the estimated value V. On the other hand, the learning rate α does not seem to significantly affect the result of Q-learning without exploration.



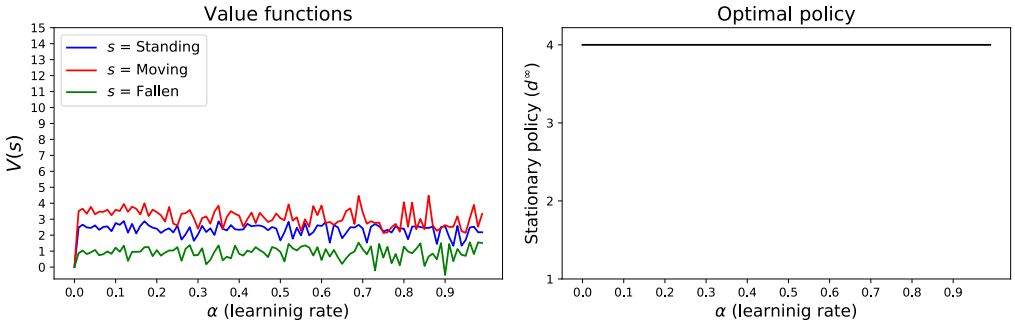
(a) $s =$ Standing (b) $s =$ Moving (c) $s =$ Fallen

Figure 4: $V(s)$ from *Q-learning without exploration* with various α & δ (iteration = 10000)

(c) Effect of Learning Rate on the Estimated Value Function and the Optimal Policy
Figure 5 describes the effects of learning rate α on value function and optimal policy. We conduct the experiments with two discount factors $\delta = 0.9, 0.7$. We find that the value of the learning rate has no critical effect on the result of Q-learning without exploration. We may conclude that only using exploitation lets us fall into the local optimum.



(a) $\delta = 0.9$



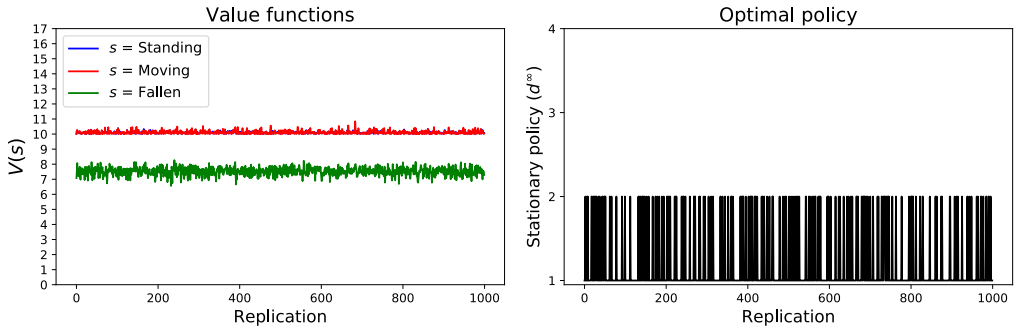
(b) $\delta = 0.7$

Figure 5: *Q-learning without exploration* with various learning rates under two discount factors ($\delta = 0.9, 0.7$)

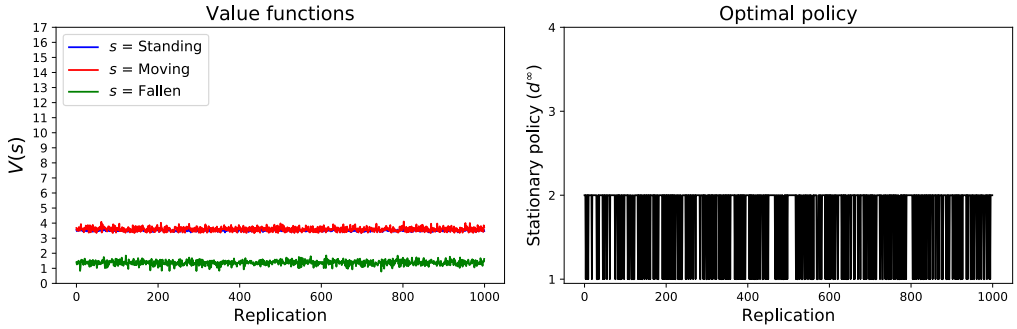
4. Q-Learning with ϵ-greedy Policy ($\epsilon = 0.5$)

   (a) Convergence

   We run the 10000-iteration Q-learning with ϵ-greedy policy for 1000 replications and record the value funtion and optimal policy. We set the learning rate $\alpha = 0.05$ again. As we can notice from Figure 6, the result is more convergent than the previous experiments. The estimated value function is quite close to the true optimal. However, it is not guaranteed to return the true optimal policy as well. The optimal policy derived from this alogorithm wanders between $d^{\infty 1}$ and $d^{\infty 2}$. This is because the optimal value function (V) of the policies $d^{\infty 1}$ and $d^{\infty 2}$ has no big difference ($V_{0.9}^{d^{\infty 1}} = [10, 10, 7.39], V_{0.9}^{d^{\infty 2}} = [9.59, 9.55, 7.07], V_{0.7}^{d^{\infty 1}} = [3.53, 3.61, 1.36], V_{0.7}^{d^{\infty 2}} = [3.33, 3.33, 1.26]$) so even a small fluctuation in estimated values can result in a different optimal policy.



(a) $\delta = 0.9$

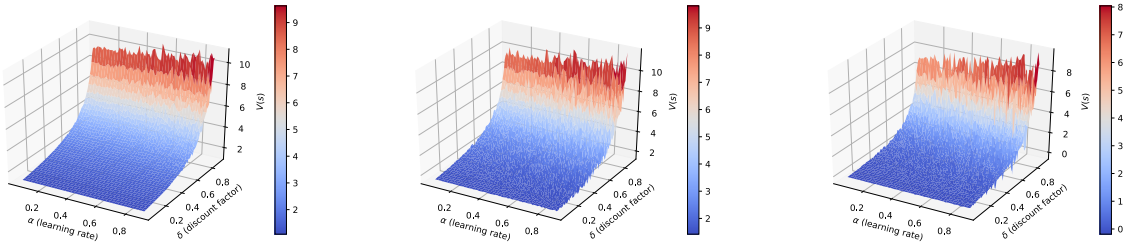


(b) $\delta = 0.7$

Figure 6: 1000 replications of *Q-learning with ϵ-greedy policy* under two discount factors ($\alpha = 0.05$)

   (b) Effects of α and δ on the Estimated Value Function

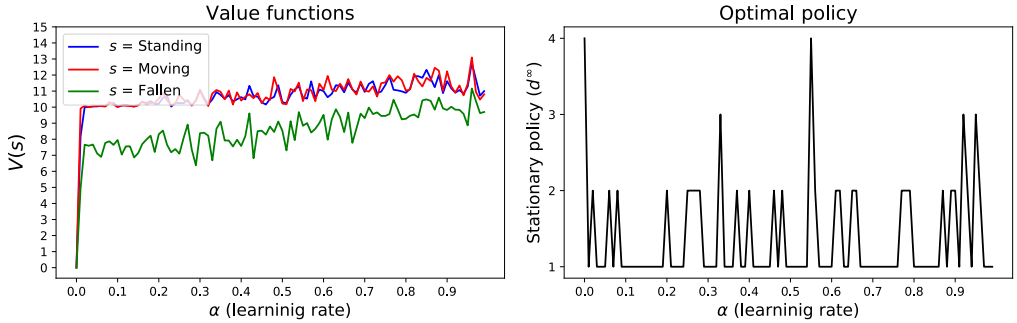
   We conduct similar experiment to the Q-learning without exploration to compare. Figure 7 shows the results. Overall, high δ value increases the values of the estimated value V again. However, we can see more stable result than Q-learning without exploration. We also find that high learning rate gives us less convergent result.



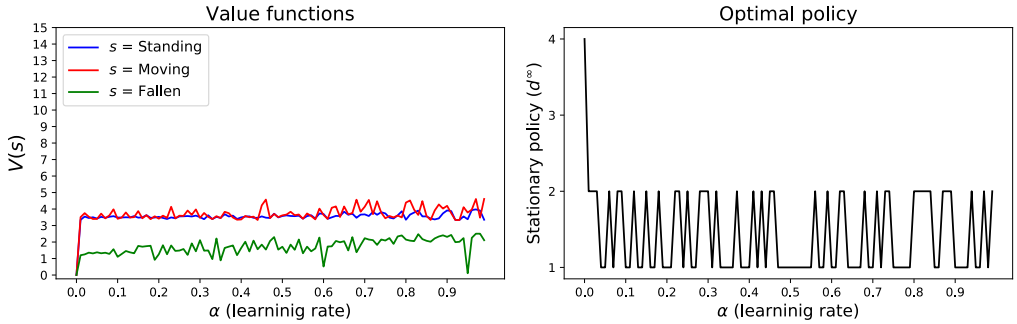
(a) $s =$ Standing (b) $s =$ Moving (c) $s =$ Fallen

Figure 7: $V(s)$ from *Q-learning with ϵ-greedy policy* with various α & δ (iteration = 10000)

(c) Effect of Learning Rate on the Estimated Value Function and the Optimal Policy
In this experiment, we also find that the estimated value function is more convergent than Q-learning without exploration. Moreover, the optimal policy derived from the algorithm is closer to the true optimal. We may conclude that the ϵ-greedy policy helps us to avoid falling into the local optimum.



(a) $\delta = 0.9$



(b) $\delta = 0.7$

Figure 8: *Q-learning with ϵ-greedy policy* under two discount factors ($\delta = 0.9, 0.7$)