Reinforcement Learning HW3

Yonatan Ariel Slutzky Eyal Grinberg

5 May 2023

1 Theoretical Questions

1.1 Question 1

We saw in class that the optimal policy π^*_M satifies the following:

$$\forall s \in S.\pi^*_M(s) \in argmax_{a\in A}(Q^*_M(s,a)) = (*)$$

Since $f(s)$ isn't dependant on a, we get that

$$(*) = argmax_{a\in A}(Q^*_M(s,a) + f(s)) = argmax_{a\in A}(Q^*_{M'}(s,a))$$

Thus, π^*_M is also greedy w.r.t $Q^*_{M'}$ and thus it is also optimal for M'. QED

1.2 Question 2

1.2.1 Section a

We'll denote the r.v. $R(s,a)$ to be the reward received when performing a from state s. $R(s,a)$ receives values of the form $r(s,a,s')$ w.p. $p(s'|s,a)$.
We'll denote as we saw in class $r(s,a) := E(R(s,a))$.
In our case, we get that

$$r(s,a) = \frac{r'(s,a) - b}{c}$$

Let $\pi \in SD$. The following holds:

$$\forall s \in S.V^\pi_M(s) = r(s,\pi(s)) + \sum_{t=1}^{\infty}\gamma^t(\sum_{s'\in S}(P^\pi)^t_{s,s'}r(s',\pi(s'))) =$$

$$= \frac{r'(s,\pi(s)) - b}{c} + \sum_{t=1}^{\infty}\gamma^t(\sum_{s'\in S}(P^\pi)^t_{s,s'}\frac{r'(s',\pi(s')) - b}{c}) = (*)$$

$\forall t \in \mathbb{N}.(P^\pi)^t$ is a row-stochastic matrix and thus each of its rows sum to 1. Thus,

$$(*) = \frac{r'(s,\pi(s)) + \sum_{t=1}^{\infty}\gamma^t(\sum_{s'\in S}(P^\pi)^t_{s,s'}r'(s',\pi(s'))) - \sum_{t=0}^{\infty}\gamma^t b}{c} =$$

$$= \frac{V^{\pi}_{M'} - \frac{b}{1-\gamma}}{c}$$

Therefore, since $c > 0$, maximizing V^{π}_{M} w.r.t to π is the same as maximizing $V^{\pi}_{M'}$ w.r.t to π. QED

1.2.2 Section b

By the Bellman equations, we have that for $\forall (s,a) \in S \times A$:

$$Q^*_M(s,a) = E_{s' \sim p(.|s,a)}(r(s,a,s') + \gamma max_{a' \in A}(Q^*_M(s',a'))) \rightarrow$$

$$Q^*_M(s,a) + \phi(s) = E_{s' \sim p(.|s,a)}(r(s,a,s') - \gamma\phi(s') + \phi(s) + \gamma max_{a' \in A}(Q^*_M(s',a') + \phi(s')))$$

We'll denote $\hat{Q}^*_M(s,a) := Q^*_M(s,a) + \phi(s)$ and thus we get

$$\hat{Q}^*_M(s,a) = E_{s' \sim p(.|s,a)}(r'(s,a,s') + \gamma max_{a' \in A}(Q^*_M(s',a')))\ (*)$$

We know that the solution for $(*)$ is given by $Q^*_{M'}$ - the solution to the Bellman equations in M'.
Therefore, the same optimal policy is valid in M and in M'. QED
The above proof is adapted from the paper Policy invariance under reward transformations Theory and application to reward shaping.
p.s. - this proof can also be easily adapted to section a.

1.3 Question 3

1.3.1 Section a

For a given iteration t, we'll denote for $\forall s \in S$.

$$d(s) := min_{s' \in S\ s.t.\ \exists a' \in A\ s.t.\ r'_t(s',a')=1}(d(s',s))$$

In other words, $d(s)$ is the minimal distance (in edges) between s and a state s' for which there exists an action a' such that the pair (s',a') is unknown.
We'll prove that any optimal policy reaches an unknown pair (and thus the iteration terminates).
We'll prove so by showing that $\forall s \in S$. If an optimal policy starts from s then it will eventually reach an unknown pair. We'll do so by induction on $d(s) = d$.
We'll note that since the graph is strongly connected, $\forall s \in S.d(s) \in \mathbb{N}$.
$\underline{d = 0}$: In this case, there exists $a \in A$ such that $r'_t(s,a) = 1$. Thus, any optimal policy π^*_t must choose an unknown action a when it starts from state s (otherwise, its total reward can be increased).
$\underline{d \rightarrow d+1}$: Suppose that for $\forall s' \in S$ the argument is true when $d(s') = d$. Suppose that $d(s) = d + 1$. When starting from s, taking an action that results in state s' for which $d(s') > d$ will hurt the total reward;
- In finite horizon, we will spend one more unnecessary step with immediate reward 0
- In discounted horizon, the total reward will shrink by a factor of γ

Thus, when starting from s any optimal policy π_t^* must choose an action a that results in state s' for which $d(s') = d$. Applying the inductive assumption finishes this section.
According to the induction principal, the argument is true for any distance $d(s)$.
QED

1.3.2 Section b

According to the proof in section a, when starting from a state s, the iteration ends after $d(s)$ actions.
Therefore, the worst case in each iteration is no more than $|S|$ actions - this is the longest simple path in a graph of $|S|$ nodes. QED

1.3.3 Section c

To uncover the entire transition function f, the total amount of iteration needed is $|S||A|$ - the number of state-action pairs (since each iteration uncovers a single pair).
If the algorithm records the actual reward it received when discovering each new unknown pair (s, a), then the total amount of iteration needed is as before.
Otherwise, a pass through all of the edges is required (after the transition function is uncovered). This pass can be done in another $|S||A|$ iterations (the number of edges in the graph).

1.3.4 Section d

We saw in section b that the time spent performing each iteration is the distance $d(s_0)$ where s_0 is the state we start the iteration from.
If the first state is shared between all iterations, then it holds that

$$T_{explotation} = \sum_{s \in S} d(s, s_0)|A| \leq |S|^2|A|$$

In other words, the total time is the sum of distances between s_0 and all states, times the amount of actions (the amount of times we need to reach any state).
The distance is obviously bound by $|S|$.
QED
p.s. - this bound can also be reached by combining sections b and c

1.4 Question 4

1.4.1 Section a

We'll estimate the reward function by the empirical mean of the rewards in each type of transition.
We'll estimate the transition probability by the empirical probability to make

each transition given its original state.

$$\hat{r}(S_A, S_B) = \frac{0+0+0+0+0+0+0}{7} = 0, \hat{r}(S_A, terminate) = \frac{2+2+2+2+2}{5} = 2$$
$$\hat{r}(S_B, S_A) = \frac{0+0+0+0+0+0+0}{7} = 0, \hat{r}(S_B, terminate) = \frac{1+1+1+1+1+1+1}{7} = 1$$
$$\hat{p}(S_A, S_B) = \frac{7}{12}, \hat{p}(S_A, terminate) = \frac{5}{12}$$
$$\hat{p}(S_B, S_A) = \frac{7}{14}, \hat{p}(S_B, terminate) = \frac{7}{14}$$

The expected empirical rewards are

$$\hat{E}(\hat{r}(S_A, .)) = \frac{10}{12}, \hat{E}(\hat{r}(S_B, .)) = \frac{1}{2}$$

1.4.2 Section b

Using the algorithm seen in class, we get the following estimations:

$$\hat{V}(S_A) = \frac{1+2+2+1+2+1+2+1+2}{9} = \frac{14}{9}, \hat{V}(S_B) = \frac{1+2+1+2+1+2+1+1+1+1+2}{11} = \frac{15}{11}$$

1.5 Question 5

1.5.1 Section a

Plugging in the definition of Δ_t and assuming it uses V^π, we get the following equality

$$E(\Delta_t | S_t = s) = E(r_t + \gamma V^\pi(S_{t+1}) - V^\pi(S_t) | S_t = s) =$$
$$= E(r(s, \pi(s)) + \gamma E(V^\pi(S_{t+1}) | S_t = s)) - E(V^\pi(s)) =$$
$$= V^\pi(s) - V^\pi(s) = 0$$

The meaning of this result is that if Δ_t uses V^π, then the expected error it quantifies is 0 as we expect it to be.
In other words, the estimator used in the algorithm will be an unbiased estimator for V^π. QED

1.5.2 Section b

Plugging in the definition of Δ_t and assuming it uses V^π, we get the following equality

$$E(\Delta_t | S_t = s, A_t = a) = E(r_t + \gamma V^\pi(S_{t+1}) - V^\pi(S_t) | S_t = s, A_t = a) =$$
$$= E(r(s, a) + \gamma V^\pi(S_{t+1}) | S_t = s) - E(V^\pi(s)) =$$
$$= Q^\pi(s, a) - V^\pi(s)$$

The meaning of this result is that the expected direction the algorithm advances to is $Q^\pi(s, a)$.

2 Programming Questions

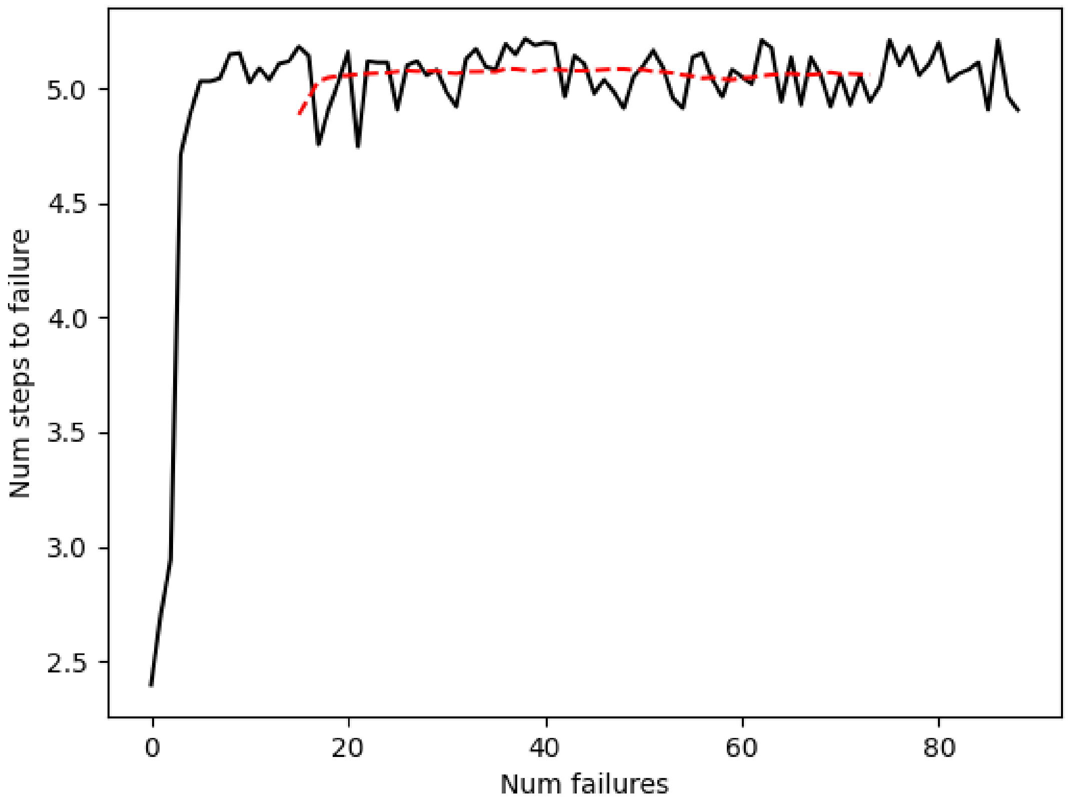
2.1 Question 1

2.1.1 Section 1

The total amount of trials until convergence was 89 iterations.

2.1.2 Section 2

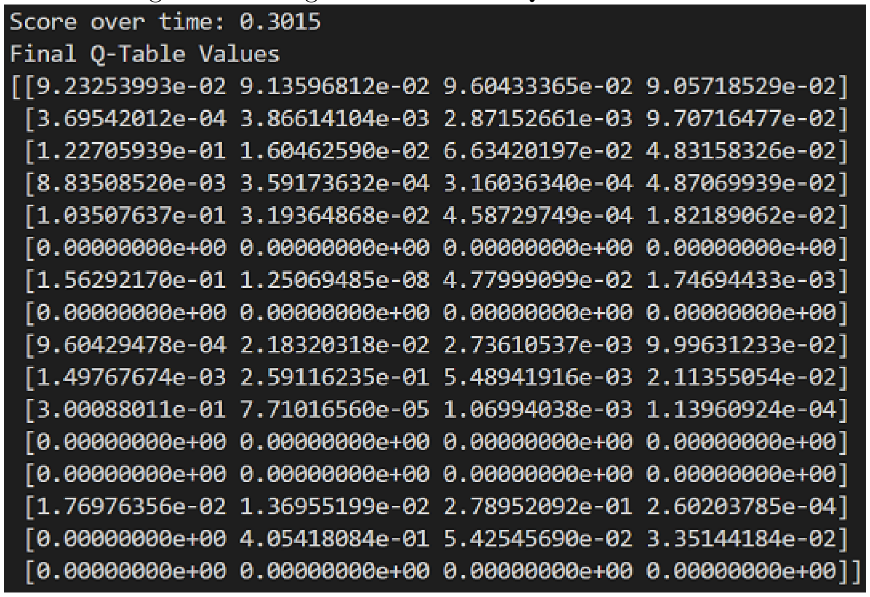
The following is the required plot:



2.2 Question 2

2.2.1 Section 1

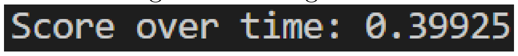
The following is the average score and final Q-table:



```
Score over time: 0.3015
Final Q-Table Values
[[9.23253993e-02 9.13596812e-02 9.60433365e-02 9.05718529e-02]
 [3.69542012e-04 3.86614104e-03 2.87152661e-03 9.70716477e-02]
 [1.22705939e-01 1.60462590e-02 6.63420197e-02 4.83158326e-02]
 [8.83508520e-03 3.59173632e-04 3.16036340e-04 4.87069939e-02]
 [1.03507637e-01 3.19364868e-02 4.58729749e-04 1.82189062e-02]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [1.56292170e-01 1.25069485e-08 4.77999099e-02 1.74694433e-03]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [9.60429478e-04 2.18320318e-02 2.73610537e-03 9.99631233e-02]
 [1.49767674e-03 2.59116235e-01 5.48941916e-03 2.11355054e-02]
 [3.00088011e-01 7.71016560e-05 1.06994038e-03 1.13960924e-04]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]
 [1.76976356e-02 1.36955199e-02 2.78952092e-01 2.60203785e-04]
 [0.00000000e+00 4.05418084e-01 5.42545690e-02 3.35144184e-02]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00]]
```

2.2.2 Section 2

The following is the average score over 4000 episodes:



```
Score over time: 0.39925
```

As we can see, using a NN improved the results over the tabular method.