ECE - 5984 : Homework 4
Instructor: Thinh T. Doan, TA: Amit Dutta
Due Date: 10/28/2021

Problem 1 [20 points]

In this problem, we study the convergence of policy iteration for state-action value function Q. In particular, consider a discounted cost MDP with finite state and finite action spaces. Given a fixed stationary policy μ let Q_μ satisfies

$$Q_\mu = T_\mu(Q_\mu),$$

where

$$T_\mu(Q)(s,a) = \mathbb{E}\left[r(s,a)\right] + \gamma \sum_{s' \in \mathcal{S}} P_{ss'}(a) Q(s', \mu(s'))$$

Consider the following policy iteration algorithm

1. Start with a stationary policy μ_0. Set $k = 0$
2. Compute Q_{μ_k} by solving

$$Q_{\mu_k} = T_{\mu_k}(Q_{\mu_k})$$

3. Find a new policy μ_{k+1} s.t.

$$\mu_{k+1}(s) = \arg \overset{\max_a}{\cancel{\min_a}} Q(s,a)$$

4. Increase k by 1 and go to step (2)

Show that μ_k converges to an optimal stationary policy.

· Consider a policy μ', which is identical to μ except for one state $\mu'(s) = a \neq \mu(s)$, and that this difference results in an improvement of the value function

$\Rightarrow \quad V_\mu(s) \leq Q_\mu(s, \mu'(s))$ | as equal for all states, except the difference which results in improvement (increased value)

$= E\left[r(s,\mu'(s)) + \gamma V_\mu(s')\right]$ | with s' = next state.

$\leq E\left[r(s,\mu'(s)) + \gamma Q_\mu(s', \mu'(s'))\right]$ | as $V_\mu(s) \leq Q_\mu(s, \mu'(s))$

$= E\left[r(s,\mu'(s)) + \gamma E\left[r(s',\mu'(s')) + \gamma V_\mu(s'')\right]\right]$

$= E\left[r(s,\mu'(s)) + \gamma r(s',\mu'(s')) + \gamma^2 V_\mu(s'')\right]$

$\leq E\left[r(s,\mu'(s)) + \gamma r(s',\mu'(s')) + \gamma^2 r(s'',\mu'(s')) + \gamma^3 V_\mu(s''')\right]$

$\vdots$

$\leq E\left[r(s,\mu'(s)) + \gamma r(s',\mu'(s')) + \gamma^2 r(s'',\mu'(s')) + \gamma^3 r(s''',\mu(s''')) + \ldots\right]$

$= V_{\mu'}(s)$

$\therefore \underline{\underline{V_\mu(s) \leq V_{\mu'}(s)}}$

- Step 3 within the outlined policy iteration algorithm above $\left[\mu_{k+1}(s) = \arg\min_a Q(s,a)\right]$ results in the greedy policy $\mu_{k+1}(s)$, by taking the action that maximises the value $V_\mu(s)$ after a one step lookahead.

- This process satisfies the initial proposition $\left[V_\mu(s) \leq Q_\mu(s, \mu'(s))\right]$ and $\therefore$ $\underline{\underline{V_{\mu_{k+1}}(s) \leq V_{\mu_k}(s)}}$, meaning the policy μ_{k+1} is as good or better than μ_k (as maximising in this Question)

- <u>Suppose that $V_{\mu_{k+1}}(s) = V_{\mu_k}(s)$:</u>

$$\therefore \quad V_{\mu_{k+1}}(s) = \max_a E\left[r(s,a) + \gamma V_{\mu_{k+1}}(s')\right]$$

- This is the bellman optimality equation and $\therefore$

$$\Rightarrow V_{\mu_{k+1}}(s) = V_{\mu_k}(s) = V^*(s)$$

$$\Rightarrow \mu_{k+1} \text{ and } \mu_k \text{ are both optimal policies.}$$

<u>Algorithm summary.</u>

$$\mu_0 \xrightarrow[\text{(step2)}]{\text{Eval}} V_{\mu_0} \xrightarrow[\text{(Step3)}]{\text{Improve}} \mu_1 \xrightarrow[\text{(Step2)}]{\text{Eval}} V_{\mu_1} \xrightarrow[\text{(Step3)}]{\text{Improve}} \mu_2 \xrightarrow[\text{(Step2)}]{\text{Eval}} \cdots\cdots \mu^* \rightarrow V^*$$

- Note that step 2 (evaluation) is due to T_{μ_k} being contractive and was shown in previous homeworks.

- The prior pages have shown that the improvement step (3) will produce a policy that is a strict improvement over the prior (unless it is already optimal). As a finite MDP only has a finite number of policies, this process must converge to the optimal policy + value function in finite steps.

Problem 2 [20 points]

Consider a discounted cost MDP with 6 states $\{1, 2, 3, 4, 5, 6\}$. Given a fixed policy, we are interested in approximating its discounted cost vector V as follows:

$$V = \Phi\theta, \quad \Phi = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \qquad \theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

Such a scheme is called hard aggregation since states $\{1, 2, 3\}$ are assumed to have the same discounted cost and the same holds for states $\{4, 5, 6\}$. As studied in class, given a vector V we want to solve the weighted least squares problem

$$\min_{\theta} \|V - \Phi\theta\|_W^2$$

where $\|x\|_W^2 = x^T W x$ and W is some diagonal matrix with positive entries $\{w_1, w_2, \ldots, w_6\}$. Show that the optimal solution to the above problem satisfies

$$\theta_i^* = \sum_{j=1}^{6} a_{ij} V_j, \qquad i = 1, 2$$

where the vectors $\sum_{j=1}^{6} a_{1j} = \sum_{j=1}^{6} a_{2j} = 1$. Also, identify which are the nonzero components in each of the two probability vectors $a_1 = [a_{11}, \ldots, a_{16}]^T$ and $a_2 = [a_{21}, \ldots, a_{26}]^T$.

let $\theta^* = \arg\min_{\theta} \|V - \Phi\theta\|_W^2$ | Solution to least squares problem stated.

$= \arg\min_{\theta} [V - \Phi\theta]^T W [V - \Phi\theta]$ | by weighted norm definition

Calculating gradient:

$$\nabla_\theta [V - \Phi\theta]^T W [V - \Phi\theta] = -\Phi^T W (V - \Phi\theta) - \Phi^T W (V - \Phi\theta)$$
$$= -2\Phi^T W (V - \Phi\theta)$$

Setting gradient to 0 to find θ^*:

$$0 = -2\Phi^T W (V - \Phi\theta^*)$$
$$= \Phi^T W (V - \Phi\theta^*)$$
$$= \Phi^T W V - \Phi^T W \Phi\theta^*$$

$\Rightarrow \quad \Phi^T W \Phi\theta^* = \Phi^T W V$

$\Rightarrow \quad \theta^* = (\Phi^T W \Phi)^{-1} \Phi^T W V$

$$\phi^T W \phi = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} w_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & w_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_6 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} w_1 + w_2 + w_3 & 0 \\ 0 & w_4 + w_5 + w_6 \end{bmatrix}$$

$$\Rightarrow (\phi^T W \phi)^{-1} = \begin{bmatrix} \frac{1}{w_1 + w_2 + w_3} & 0 \\ 0 & \frac{1}{w_4 + w_5 + w_6} \end{bmatrix}$$

$$\phi^T W V = \begin{bmatrix} w_1 & w_2 & w_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_4 & w_5 & w_6 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} v_1 w_1 + v_2 w_2 + v_3 w_3 \\ v_4 w_4 + v_5 w_5 + v_6 w_6 \end{bmatrix}$$

$$\Rightarrow \quad \theta^* = \begin{bmatrix} \frac{v_1 w_1 + v_2 w_2 + v_3 w_3}{w_1 + w_2 + w_3} \\ \frac{v_4 w_4 + v_5 w_5 + v_6 w_6}{w_4 + w_5 + w_6} \end{bmatrix}_{2 \times 1}$$

<u>Placing into provided form from Question.</u>

$$\Rightarrow \quad \theta^* = \begin{bmatrix} \theta_1^* \\ \theta_2^* \end{bmatrix}, \quad \theta_1^* = \sum_{j=1}^{6} a_{1j} V_j, \quad a_{1j} = \frac{1}{w_1 + w_2 + w_3} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\theta_2^* = \sum_{j=1}^{6} a_{2j} V_j, \quad a_{2j} = \frac{1}{w_4 + w_5 + w_6} \begin{bmatrix} 0 \\ 0 \\ 0 \\ w_4 \\ w_5 \\ w_6 \end{bmatrix}$$

$\therefore$ non zero components of:

$a_1 = a_{11}, a_{12}, a_{13}$

$a_2 = a_{24}, a_{25}, a_{26}$

$a_1 = [a_{11}, a_{12} \dots a_{16}]^T$

$a_2 = [a_{21}, a_{22} \dots a_{26}]^T$

//

Problem 3 [40 points]

An individual desires to sell her car. A new offer is made to her every day and she must decide whether to accept or not the offer. Once rejected, the offer is lost. Suppose that successive offers are independent of each other and take on the value i with probability P_i, $i = 0, 1, \ldots, N$. Suppose also that the car has a maintenance cost C per day, while future costs are discounted at rate α. Let the state at time k be the corresponding offer. Answer the following questions (note in this case we have a minimization problem):

1. Using the context of MDP, determine the Bellman equation satisfied by the optimal value $V^*(i)$ for all $i \in \{0, 1, \ldots, N\}$ [10 points].
2. Based on your answer in (1), the optimal policy should have a threshold form: there exists i^* such that the individual should accept any offer i such that $i \leq i^*$ and she should reject any offer such that $i < i^*$. Give the description of i^* [10 points].
3. Let μ_i be the policy which accepts any offer greater than or equal to i. For this policy, find the conditional expected discounted cost given K, which is the number of rejected offers [10 points].
4. Based on your answer in (3), consider the expected discounted cost by averaging over K and be more explicit about how i^* should be chosen [10 points].

3.1) $$V^*(s) = \min_a E\left[r(s,a) + \gamma \sum_{s' \in S} P_{ss'}(a)\, V^*(s')\right]$$

with: • $$r(s,a) = \begin{cases} C_0\alpha^k - i & \mid a = \text{sell},\ k = \text{step \#} \\ C_0\alpha^k & \mid a = \text{pass},\ k = \text{step \#} \\ 0 & \mid a_{k-1} = \text{sell} \end{cases}$$ (sold the car ∴ no more costs, terminal)

• assuming that when accepting an offer, the days maintence costs <u>are</u> payed $(C_0\alpha^k)$

• actions = $\{\text{sell}, \text{pass}\}$

• state space = $S = \{i_0, i_1, i_2 \ldots i_N, \text{sold}\}$

$= \{0, 1, 2 \ldots N, \text{sold}\}$ | where sold is the terminal state.

(note here I will use s for a state vs i, as i is used multiple times throughout this question)

• Transition probabilities $$P^a_{ss'} = \begin{cases} P^{sell}_{ss'} = \begin{bmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & 1 \\ \vdots & \cdots & \vdots & \vdots \\ 0 & \cdots & 0 & 1 \end{bmatrix} \\ P^{pass}_{ss'} = \begin{bmatrix} P_1 & P_2 & \cdots & P_N & 0 \\ \vdots & \vdots & & \vdots & \vdots \end{bmatrix} \end{cases}$$

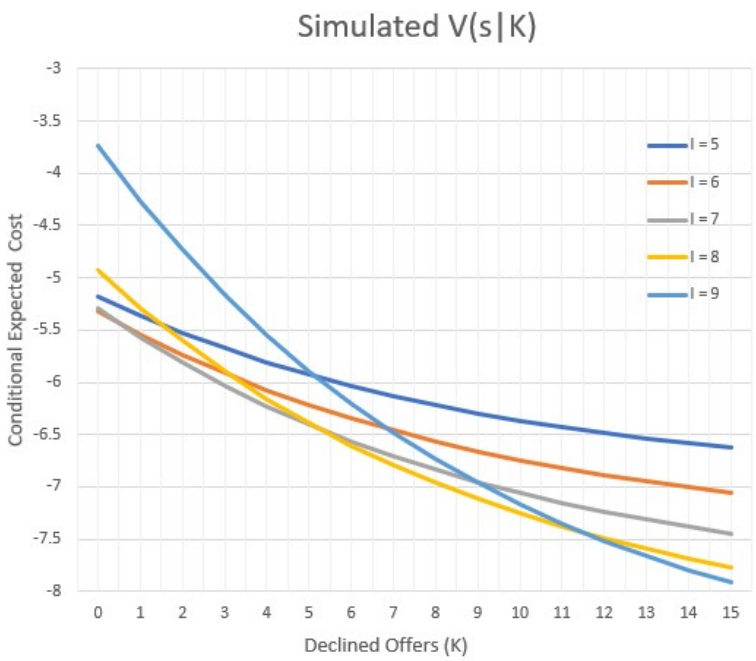
3.2) $\mu^*(s) = \arg\min_a E\left[r(s,a) + \gamma \sum_{s' \in S} P(a) V^*(s')\right]$

* There exists an offer i^* where the expected future costs are outweighed by the current offer i, and should therefore be accepted.

* Note that following the assumptions I have listed within Q3.3, the resultant optimal policy will not have a constant i^*, and will instead vary based on the number of previously declined offers.

. Below is a table displaying an example $V_{\mu_i}(s|K)$, (from 3.3 calcs), where μ_i is the policy which accepts any offer $i \geq I$. The following variables are used:

$\left[C = 1,\ \alpha = 0.9,\ \gamma = 1,\ N = 9,\ \text{with offers sampled from a uniform distribution with replacement.}\right]$



The optimal policy in this case is $\mu^* = \{I=6 \mid K=0,\ I=7 \mid 1 \leq K < 6,\ I=8 \mid 6 \leq K < 12,\ I=9 \mid 12 \leq K\}$ which result in the minimization of expected future costs

3.3) List of assumptions as the question is vague:

- "Once rejected, an offer is lost" - meaning that previous offers cannot be accepted at a later date.
- Offers can however be repeated (0 can be offered infinitely many times)
- The distribution from which these offers are drawn is known.
- "Find the conditional expected cost given K (# of rejected offers)" - meaning that so far, K offers have been rejected. nothing is stated about future offers/actions. The current (K+1 offer) and offers into the future can still be declined.

Notation

- let μ_i be the policy which accepts an offer iff $i \geq I$ (I = sell price).
- let $P_1 = P(i \geq I)$ (probability any offer $> I$)
- $P_2 = 1 - P_1 = P(i < I)$ (probability offer declined)

$$V_{\mu_i}(s|K) = P_1\left(C\alpha^K - i_K\right) + P_2\left(C\alpha^K\right) +$$
$$+ (P_2\gamma)\left[P_1\left(C\alpha^{K+1} - i_{K+1}\right) + P_2\left(C\alpha^{K+1}\right)\right]$$
$$+ (P_2\gamma)^2\left[P_1\left(C\alpha^{K+2} - i_{K+2}\right) + P_2\left(C\alpha^{K+2}\right)\right]$$
$$+ (P_2\gamma)^3\left[\ldots\ldots\right] + (P_2\gamma)^4\left[\ldots\right] + \ldots\ldots$$

$$= \sum_{l=0}^{\infty} (p_2\gamma)^l \left[p_1\left(c\alpha^{k+l} - i_{k+l}\right) + p_2\left(c\alpha^{k+l}\right)\right]$$

$$= \sum_{l=0}^{\infty} (p_2\gamma)^l \left[p_1\left(c\alpha^k\alpha^l - i_{k+l}\right) + p_2\left(c\alpha^k\alpha^l\right)\right]$$

$$= \sum_{l=0}^{\infty} (p_2\gamma\alpha)^l \left(p_1 c\alpha^k + p_2 c\alpha^k\right) - \sum_{l=0}^{\infty} (p_2\gamma)^l i_{k+l} p_1$$

$$= \sum_{l=0}^{\infty} (p_2\gamma\alpha)^l c\alpha^k - \sum_{l=0}^{\infty} (p_2\gamma)^l i_{k+l} p_1$$

$$= \frac{c\alpha^k}{1-p_2\gamma\alpha} - \frac{p_1 E(i \mid i \geq I)}{1-p_2\gamma}$$

$$\therefore V_{M_i}(s \mid k) = \frac{c\alpha^k}{1-p_2\gamma\alpha} - \frac{p_1 E(i \mid i \geq I)}{1-p_2\gamma}$$

If $\gamma = 1$:

$$V_{M_i}(s \mid k) = \underbrace{\frac{c\alpha^k}{1-p_2\alpha}}_{\text{expected future costs, based on \# of expected future rejected offers}} - \underbrace{E(i \mid i \geq I)}_{\text{expected offer following } M_i}$$

3.4)

$$V_{\mu i}(s \mid K=0) = \frac{C}{1-P_2\alpha} - E(i \mid i \geq I)$$

$$I^* = \underset{I}{\arg\min}\left[\frac{C}{1-P_2\alpha} - E(i \mid i \geq I)\right]$$

or if $\gamma \neq 1$:

$$V_{\mu i}(s \mid K=0) = \frac{C}{1-P_2\gamma\alpha} - \frac{P_1\, E(i \mid i \geq I)}{1-P_2\gamma}$$

$$I^* \mid K=0 = \underset{I}{\arg\min}\left[\frac{C}{1-P_2\gamma\alpha} - \frac{P_1\, E(i \mid i \geq I)}{1-P_2\gamma}\right]$$

* This value of I^* will be the optimal price at which any offer $i > I$ should be accepted, based on the distribution of offers $(P_1, P_2, E(i \mid i \geq I))$ the maintence cost (C), discount of maintence cost (α) and the discount rate (γ), and critically $K=0$

* The optimal policy will therefore have a varying I value dependent on the number of declined offers, which can be calculated using the above $I^* \mid K$ formula.

$$\therefore \mu^*(K) = \{ I^* \mid K=0,\ I^* \mid K=1,\ I^* \mid K=2 \ldots \}$$

* γ can be used to tune the depreciating value of future money (as getting money sooner has higher utility)

Problem 4 - Coding question [20 points]

Questions: Write a simulation program to implement Q-learning in the tabular setting for the cliff-walking problem. In your simulation, consider a number of episodes, where each episode runs until the program terminates or at most 50 steps. You can choose your own behavior policy but here is one suggestion: the behavior policy of each new episode could be the policy returned by the last run. The initial behavior policy could be a randomized policy.

Consider the Cliff-walking problem shown in Fig. 1 and given in Homework 3. Submit your code and explain what's the optimal policy your algorithm can return? Plot a curve to show the total reward returned by each episode, i.e., the number of points should be equal to the number of episode.

The performance of Q-learning on the cliffwalking-v0 environment is displayed within figure 2. On average the optimal policy is found within 49.4 episodes, with performance over 100 runs displayed within figure 4.

The optimal policy is one that moves around the cliff, while staying as close as possible and never falling off. Following this policy, an agent is able to reach the terminal node within 13 steps, the mininimum number of steps possible. The algorithim implemented is able to solve for this policy, and is displayed within figure 3, with the optimal path highlighted in yellow (which is always taken as the agent acts greedily).

Code is attached in full within the separate HW4_Q4.py file.

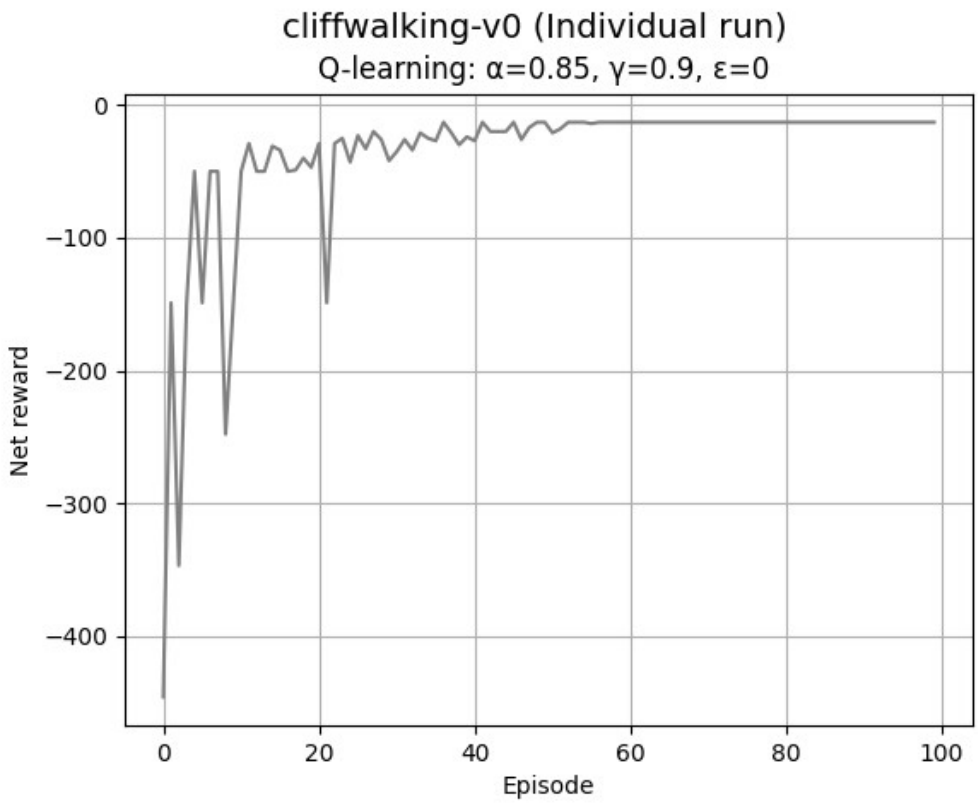


Figure 2: Individual run performance for cliffwalking-v0

↑	→	→	→	→	→	→	→	→	→	→	↓
↑	→	→	→	→	↓	→	→	→	→	→	↓
→	→	→	→	→	→	→	→	→	→	→	↓
↑	↑	→	→	→	↑	↑	→	←	→	←	↑

Figure 3: Policy generated after 100 episodes

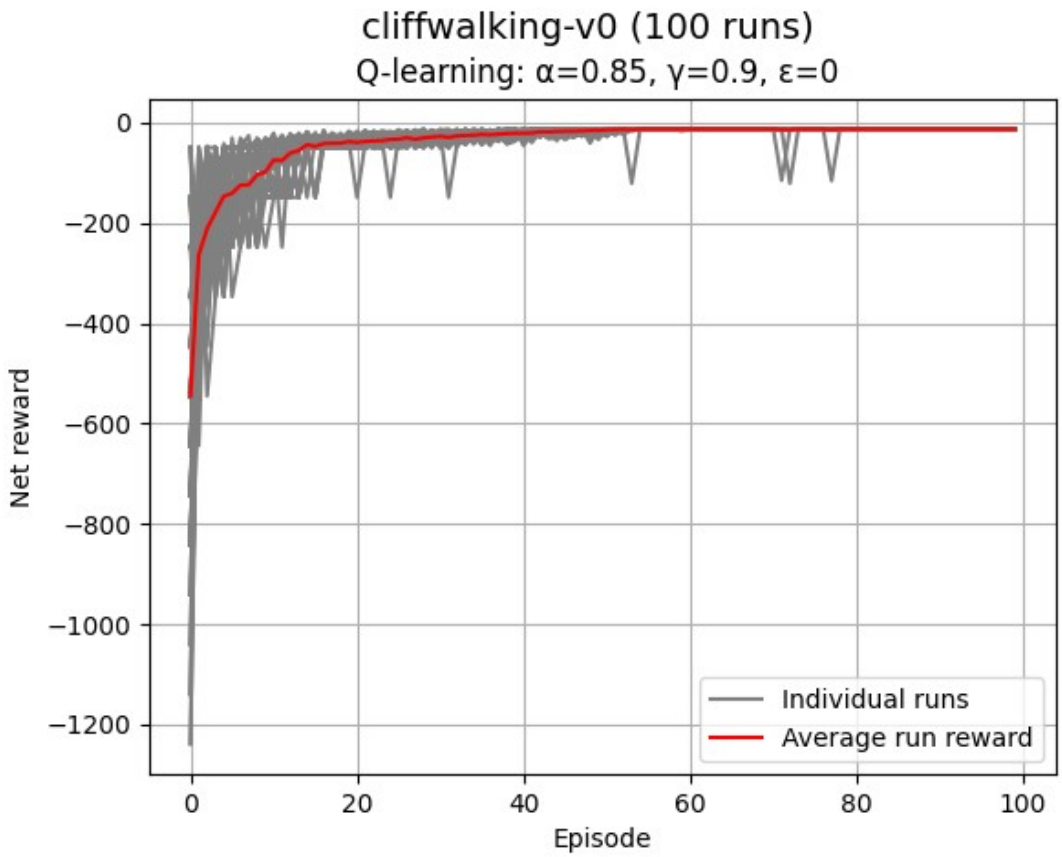


Figure 4: Multi (100) run performance for cliffwalking-v0 with average

Problem 5 - Coding question [20 points]

Consider the Cartpole environment from OpenAI Gym, which can be found here:
https://github.com/openai/gym/blob/master/gym/envs/classic_control/cartpole.py

In this question, we will consider Q-learning with linear function approximation using Fourier basis [1]. For this problem, consider discount factor is $\gamma = 0.9$ and a behavior policy a randomized policy.

[1.] Value Function Approximation in Reinforcement Learning using the Fourier Basis. George Konidaris and Sarah Osentoski and Philip Thomas.

Questions: Write a simulation program to implement Q-learning with linear function approximation for the cart-pole problem. In your simulation, consider a number of episodes, where each episode runs until the program terminates. You can choose your own behavior policy but here is one suggestion: the behavior policy of each new episode could be the policy returned by the last run. The initial behavior policy could be a randomized policy.

Submit your code and explain what's your linear function approximation? Plot a curve to show the total reward returned by each episode, i.e., the number of points should be equal to the number of episode. Does your returned policy solve the problem, i.e., how long your program last by running that's policy?

Description:

The linear function approximation used is a 3rd order Fourier basis. Combinations of 0, 1, 2 and 3 are used for each row of the coefficent vector, with variable coupling addressed by setting maximum number of non zero values in each row equal to 3 (allowing for interactions between state elements). The 174x4 coefficent vector is listed attached within coefficent_vector_Q5.txt. The following equation as outlined within the afformentioned paper is then used to generate the Fourier basis used for value function approximation.

$$\varphi(x) = \cos(\pi c \cdot x) \qquad [c = \text{coefficent vector}, x = \text{state vector}]$$

Performance of this method is outlined within figure 2, displaying a singular run along with the decaying values of alpha and epsilon used. Figure 3 displays performance over multiple runs.

On average over 100 consecutive runs, CartPole-v0 is able to be solved within 109.9 episodes. The time to solve the problem on average is 1.71s, with the distribution displayed within figure 4.

Code is attached in full within the separate HW4_Q5.py file.

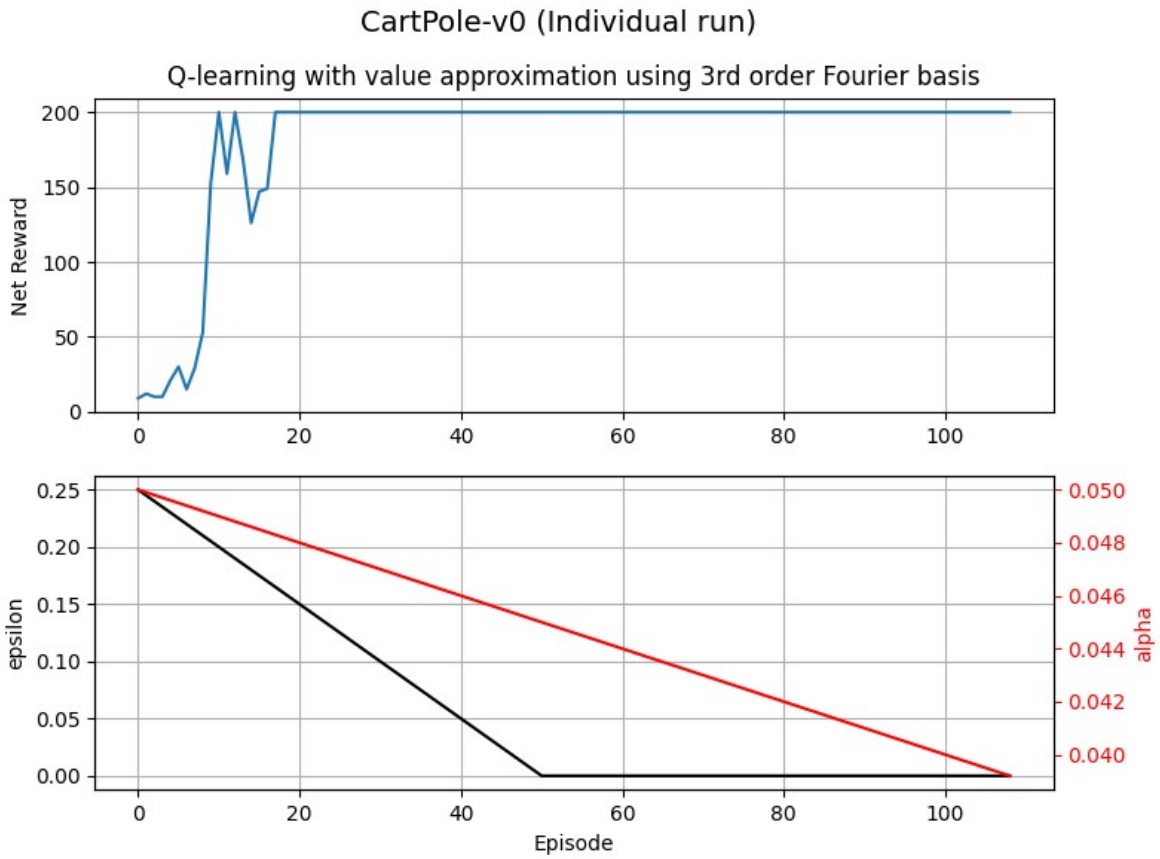


Figure 2: Individual run performance, decaying alpha and epsilon values.

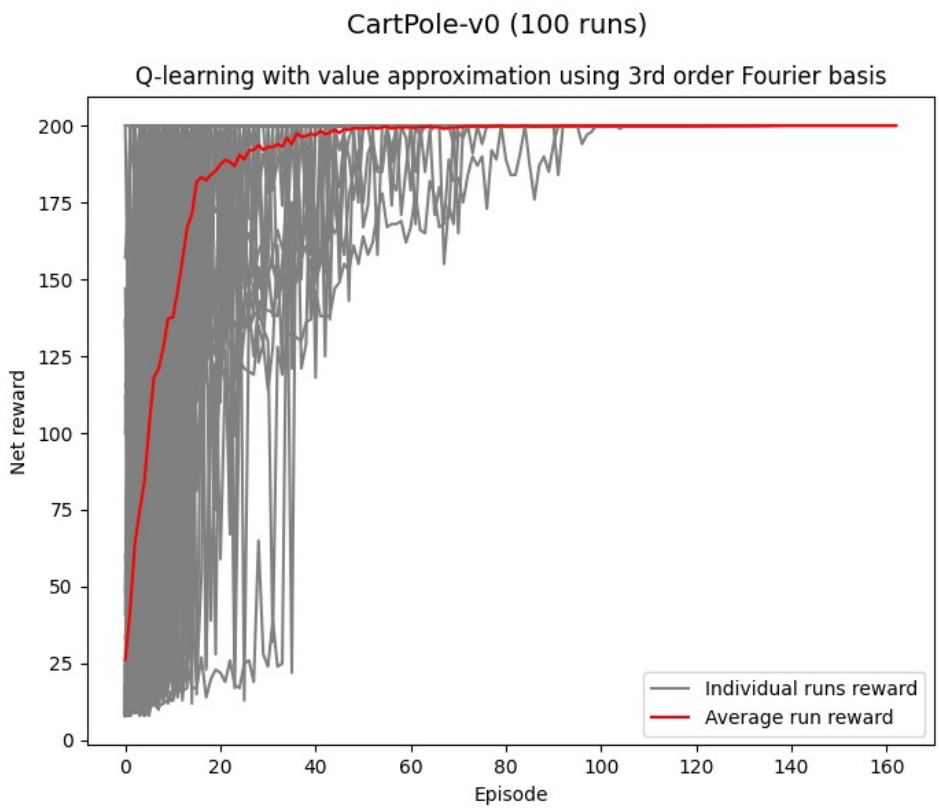


Figure 3: Multi (100) run performance with average run reward.

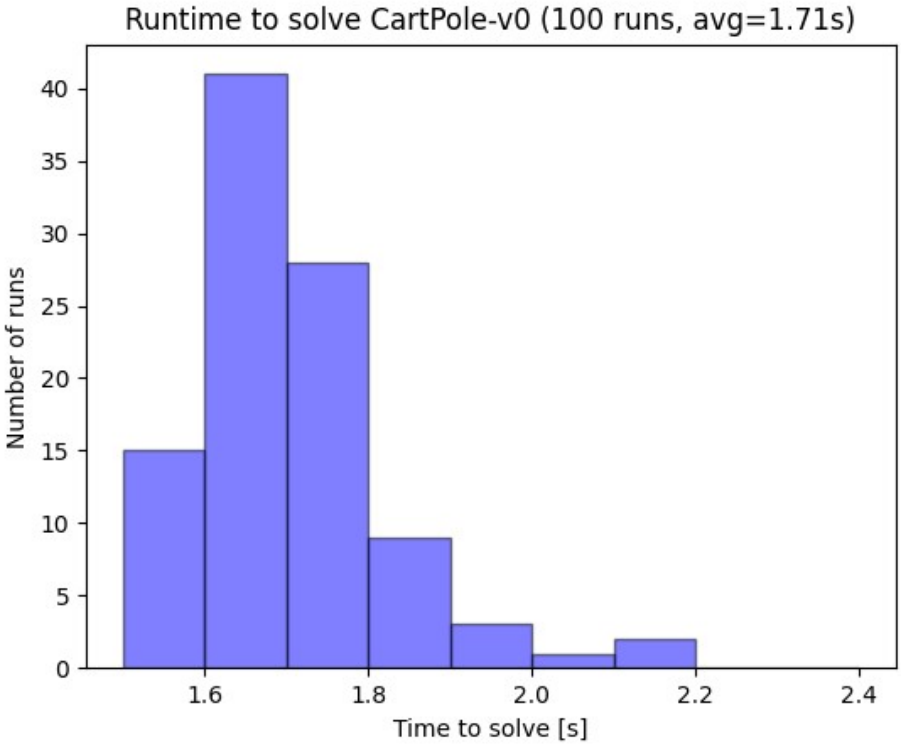


Figure 4: Histogram of time to solve for 100 runs.