Stanford CME 241 (Winter 2022) - Final Exam

Instructions:

- We trust that you will follow The Stanford Honor Code.

- You have about 48 hours to work on this test, and need to submit your answers by 8:59am Wednesday March 16. However, the estimated amount of work for this test is about 4 -6 hours, depending on your proficiency in typesetting mathematical notations and in writing code. There are 4 problems (with subproblems), with a total of 40 points.

- Include all of your writing, math formulas, code, graphs etc. into a single answers-document. Be sure to write your name and SUNET ID in your answers-document. You can do your work in LaTeX or Markdown or Jupyter or any other typesetting option, which should finally be converted to the single answers-document PDF to be submitted. Note: Code can be included in LaTeX using the *lstlisting* environment, and graphs can be included using *includegraphics*.

- Additionally, submit your code files to the code Gradescope assignment. This means that you will include your code in 2 submissions, first in the PDF answers document, and second directly as python / jupyter notebook files through Gradescope

- **Note** you must make sure that your latex correctly compiles, we **will** penalize solutions with un-compiled latex

- Submit your answers-document on Gradescope. Please ensure you have access to Gradescope before starting the exam so there is ample time to correct the problem. The Gradescope assignment for this final is *here*.

- **DO NOT** upload your solutions to github

Problems:

1. **Problem 1 : Double Q Learning, 8 Points**

   It is known that Q-Learning can suffer from a maximization bias during finite-sample training. In this problem we consider the following modification to the Tabular Q-Learning algorithm called Double Q-Learning:

Algorithm 1 Double Q-Learning

```
Initialize $Q_1(s,a)$ and $Q_2(s,a)$ $\forall s \in \mathcal{N},\ a \in \mathcal{A}$
yield estimate of Q
while True do
    select initial state $s_0$, set $t = 0$
    while $s_t \in \mathcal{N}$ do
        select $a_t$ using $\epsilon$-greedy based on this greedy policy $\pi(s) = \arg\max_a (Q_1(s_t,a) + Q_2(s_t,a))$
        observe $(r_t, s_{t+1})$
        if with 0.5 probability then
            $Q_1(s_t,a_t) \leftarrow Q_1(s_t,a_t) + \alpha(r_t + \gamma Q_1(s_{t+1}, \arg\max_a Q_2(s_{t+1},a)) - Q_1(s_t,a_t))$
        else
            $Q_2(s_t,a_t) \leftarrow Q_2(s_t,a_t) + \alpha(r_t + \gamma Q_2(s_{t+1}, \arg\max_a Q_1(s_{t+1},a)) - Q_2(s_t,a_t))$
        t = t + 1
        $s_t = s_{t+1}$
    yield estimate of Q
```

Algorithm 2 Q-Learning

```
Initialize $Q(s,a)$ $\forall s \in \mathcal{N},\ a \in \mathcal{A}$
yield Q
while True do
    select initial state $s_0$, set $t = 0$
    while $s_t \in \mathcal{N}$ do
        select $a_t$ using $\epsilon$-greedy based on this greedy policy $\pi(s) = \arg\max_a Q(s_t,a)$
        observe $(r_t, s_{t+1})$
        $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha(r_t + \gamma Q(s_{t+1}, \arg\max_a Q(s_{t+1},a)) - Q(s_t,a_t))$
        t = t + 1
        $s_t = s_{t+1}$
    yield Q
```

We have provided the code skeleton for this problem in *f_22_p1_skeleton.py*. You should pull the most recent version of the RL-Book repository to make sure the code is compatible with the rest of the library. Some parts of this question involve running the code and plotting quantities, you may use whatever environment you like to do so, but we would recommend a jupyter notebook. Be sure to include all the code you write (including code to generate / plot results) in your final submission.

1. **3 points:** Implement the *Double Q – Learning* and *Q – Learning* Algorithms within the supplied class only modifying the parts of the code which you are instructed to.

Solution: See Code Solutions

2. **3 points:** Implement the following MDP in the *f_22_p1_skeleton.py* file. We have included the skeleton of the MDP class and the State class, you should only modifying code where you are instructed to.

   - There are two non-terminal states: A, B
   - From state A there are two actions: a_1, a_2
     - action a_1 results in a reward of 0 and leads to state B
     - action a_2 results in a reward zero and a transition to a terminal state
   - From state B there are n actions: $\{b_i\}_{i=1}^n$
     - each action b_i results in a reward drawn iid from $\mathcal{N}(-0.1, 1)$ and a transition to a terminal state

Solution: See Code Solutions

3. **1 point:** Now run both the Double Q-Learning and Q-Learning algorithms on the MDP you just implemented with $n = 10$. Run each algorithm 100 times, starting from the initialization

$$Q(a, s) = 0 \ \forall \ a, s$$

for 400 episodes each run. Each episode should start in state A.

Plot the average values $\bar{Q}_i^{dbl.}(A, a_1)$ and $\bar{Q}_i^{std.}(A, a_1)$ for $i \in (0, 400)$, where $\bar{Q}_i(\cdot)$ is the i'th iterate returned by your function averaged across the 100 runs. Explain the results that you see.

Solution: See Code solutions for plots.

4. **1 point:** What are the benefits and drawbacks of the Double Q-Learning algorithm for general MDPs?

Solution: Benefits: deals with maximization bias

Drawbacks: double the memory requirements

2. **Problem 2: Policy Gradient, 5 Points**

   1. **5 points:** Recall the Natural Policy Gradient algorithm from lecture. Assume as in lecture that we have our policy $\pi(s, a; \theta)$ as an arbitrary function approximation parameterized by θ, that we take the critic to be $Q(s, a; w)$ a linear function approximation parameterized by weights w, and that we enable Compatible Function Approximation by setting

$$Q(s, a; w) = \sum_{i=1}^{m} \frac{\partial \log \pi(s, a; \theta)}{\partial \theta_i} \cdot w_i$$

Recall that Natural Policy Gradient updates the policy parameters in the direction of w

Prove that small changes in the policy parameters result in small changes in the policy, i.e. with a step size of α we have

$$\pi(a; s, \theta') = \pi(a; s, \theta)(1 + \alpha Q(s, a; w)) + O(\alpha^2)$$

Solution:

Take a taylor expansion of π around θ, and use the fact that $\theta' = \theta + \alpha w$

$$\begin{aligned}
\pi(a; s, \theta') &= \pi(a; s, \theta) + (\theta' - \theta)^T \nabla_\theta \pi(a; s, \theta) + O((\|\theta' - \theta\|)^2) \\
&= \pi(a; s, \theta) + (\alpha w)^T \nabla_\theta \pi(a; s, \theta) + O(\alpha^2) \\
&= \pi(a; s, \theta) + \alpha w^T (\pi(a; s, \theta) \nabla_\theta \log(\pi(a; s, \theta))) + O(\alpha^2) \\
&= \pi(a; s, \theta) + \alpha \pi(a; s, \theta) w^T \nabla_\theta \log(\pi(a; s, \theta)) + O(\alpha^2) \\
&= \pi(a; s, \theta) + \alpha \pi(a; s, \theta) Q(s, a; w) + O(\alpha^2)
\end{aligned}$$

3. **Problem 3: Car Sales, 12 Points**

Imagine you own a car which you need to sell within a finite window of N days. At the beginning of each day your receive an offer from a dealership which is randomly distributed over the interval $[m, M]$, $0 < m < M$ with a known continuous distribution Q on this support; the offers each day are i.i.d. After receiving an offer the you have to decide immediately whether or accept or reject this offer. If you reject the offer, it is lost and you have to pay a parking cost for the car of $c \geq 0$, you must pay this cost after each day you do not sell the car. After N days the car has to be sold. m, M, and c are all fixed positive real numbers. You want to maximize the sale proceeds.

1. **4 points:** With proper mathematical notation, model this the fully problem as an MDP specifying the states, actions, rewards, state-transition probabilities and discount factor. What particular kind of MDP is this?

Solution: This is an optimal stopping problem. The nonterminal states are

$$\mathcal{N} = \{(x, n)\}$$

where n is the number of days left until the window closes and x is the current offer. There is one terminal state, call it $\{T\}$.

The actions from each state are

$$\mathcal{A}(x, n) = \begin{cases} \{\text{accept, reject}\} & \text{if } n > 0 \\ \{\text{accept}\} & \text{otherwise} \end{cases}$$

The state transitions and rewards are

$$P(T|(x, n), \text{accept}) = 1, \ \mathcal{R}((x, n), \text{accept}) = x$$

$$P((y, n-1)|(x, n), \text{reject}) = Q(y), \ \mathcal{R}((x, n), \text{reject}) = -c$$

In this case we choose the discount factor $\gamma = 1$

2. **3 Points:** Solve this MDP analytically for the optimal policy.

Solution: We have the following cutoff c_n where if the offer is above c_n then we sell and if the offer is below c_n we do not

$$c_n = -c + c_{n-1} \int_m^{c_{n-1}} Q(x)dx + \int_{c_{n-1}}^M xQ(x)dx, \quad c_1 = \mathbb{E}_Q[x] - c$$

3. **5 Points:** Consider the case where $c = 0$ and $Q(x) = U_{[m,M]}(x)$ (the uniform distribution on $[m, M]$), solve for as closed form a solution of the optimal policy *as is possible.* To make this concrete, the functional form of your optimal policy should be explicitly defined, but can depend on coefficients which can be recursively defined. You should not have integrals in your solution.

Solution:
the policy is to sell if $n = 0$ or the offer is greater than or equal to c_n where the cutoff c_n is given by

$$c_n = \frac{\alpha_n M + (\beta_n - \alpha_n)m}{\beta_n}$$

Where

$$\beta_n = 2^{2^n - 1}$$

$$\alpha_1 = 1,\ \alpha_{n+1} = \alpha_n^2 + \beta_n^2$$

We show this via induction

The base case is easy,

$$c_1 = \frac{M + m}{2}$$

now assume

$$c_n = \frac{\alpha_n M + (\beta_n - \alpha_n)m}{\beta_n}$$

Note that

Solution:

$$\begin{aligned}
c_{n+1} &= c_n \int_m^{c_n} \frac{1}{M-m} dx + \int_{c_n}^{M} \frac{x}{M-m} dx \\
&= \frac{1}{M-m}(c_n(c_n - m) + \frac{1}{2}(M^2 - c_n^2) \\
&= \frac{1}{M-m}(\frac{c_n^2 - 2mc_n + M^2}{2}) \\
&= \frac{1}{M-m}(\frac{(\frac{\alpha_n M+(\beta_n-\alpha_n)m}{\beta_n})^2 - 2m\frac{\alpha_n M+(\beta_n-\alpha_n)m}{\beta_n} + M^2}{2}) \\
&= \frac{1}{M-m}(\frac{(\alpha_n M + (\beta_n - \alpha_n)m)^2 - 2m\beta_n(\alpha_n M + (\beta_n - \alpha_n)m) + \beta_n^2 M^2}{\beta_n^2 * 2}) \\
&= \frac{1}{(M-m)(\beta_n^2 * 2)}(\alpha_n^2 M^2 + 2\alpha_n M(\beta_n - \alpha_n)m \\
&\quad + (\beta_n - \alpha_n)^2 m^2 - 2m\beta_n\alpha_n M - (\beta_n - \alpha_n)2m^2\beta_n + \beta_n^2 M^2) \\
&= \frac{1}{(M-m)(\beta_n^2 * 2)}(M^2(\alpha_n^2 + \beta_n^2) + (2\alpha_n(\beta_n - \alpha_n) - 2\beta_n\alpha_n)mM \\
&\quad + ((\beta_n - \alpha_n)^2 - 2(\beta_n - \alpha_n)\beta_n)m^2) \\
&= \frac{1}{(M-m)(\beta_n^2 * 2)}(M^2(\alpha_n^2 + \beta_n^2) + (-2\alpha_n^2)mM \\
&\quad - (\beta_n^2 - \alpha_n^2)m^2) \\
&= \frac{1}{(M-m)(\beta_n^2 * 2)}(M(\alpha_n^2 + \beta_n^2) + (\beta_n^2 - \alpha_n^2)m)(M-m) \\
&= \frac{M(\alpha_n^2 + \beta_n^2) + (\beta_n^2 - \alpha_n^2)m)}{\beta_n^2 * 2} \\
&= \frac{M\alpha_{n+1} + (\beta_{n+1} - \alpha_{n+1})m)}{\beta_{n+1}}
\end{aligned}$$

4. **Problem 4: Crop Control, 15 Points**

Imagine that you are a farmer and have to decide how to manage your crops. Imagine that you have N acres of land on which you can p lant your crop, and at each month you must make a decision whether or not to harvest and sell each acre of crops. Each acre of crops has an associated integer quality level ranging from 0 to C: $q_t \in \mathbb{Z}^N_{[0,C]}$ for each time step t, which determines how much you can sell the acre of crops for. As the crops mature, their quality usually increases, however the crops grow in a random manner, and there is always the chance that over the next month, an acre of crops may "go bad" and it's quality will go to zero. If an acre i of crops had quality $q_{t,i}$ in month t and it is not harvested, the quality the next month follows the following distribution

$$q_{t+1,i} \sim \begin{cases} 0 & \text{with probability } z \\ U(q_{t,i}, C) & \text{with probability } 1 - z \end{cases}$$

Where $U(x,y)$ is a discrete uniform random variable on the integer-valued interval $[x,y]$. You may assume that each acre grows independently of each other.

As crops increase in quality, they become more valuable, so there is some increasing pricing function $P(q_{t,i}) : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{\geq 0}$ which determines the price you are paid for an acre of crops of quality $q_{t,i}$.

After you harvest an acre, its quality goes down to zero for the next month, furthermore the crops grow automatically on each acre, so there is no need to take an action to plant fresh crops after harvesting or after an acre goes bad.

Your goal is to maximize the expected sum of discounted utility of crop sales. If we take the utility function to be u, the goal is to maximize

$$\mathbb{E}_{t=0}[\sum_{i=0}^{\infty} \gamma^i u(\text{proceeds from sales at time } t = i)]$$

1. **3 points:** With proper mathematical notation, model this the fully problem as an MDP specifying the states, actions, rewards, state-transition probabilities and discount factor.

Solution: This is an infinite-horizon MDP, the states are

$$\mathcal{S} = \mathcal{N} = \mathbb{Z}^N_{[0,C]}$$

The actions are whether or not to harvest each acre

$$\mathcal{A} = \{0,1\}^N$$

The rewards are simply

$$\mathcal{R}(s,a) = U(P(s)^T a)$$

Where $P(s)$ is just the pricing function applied element-wise to the state s.
The probability transitions are

$$p(s,a,s') = \prod_{i=1}^{N} \tilde{p}(s_i, a_i, s'_i)$$

where

$$\tilde{p}(s_i, a_i, s_i') = \begin{cases} 0 & \text{if } a_i = 1 \text{ and } s_i' > 0 \\ 1 & \text{if } a_i = 1 \text{ and } s_i' = 1 \\ 0 & \text{if } a_i = 0 \text{ and } s_i' \in [1, s_i) \\ z + (1-z)/(C - s_i + 1) & \text{if } a_i = 0 \text{ and } s_i' = 0 \text{ and } s_i = 0 \\ z & \text{if } a_i = 0 \text{ and } s_i' = 0 \text{ and } s_i \neq 0 \\ \frac{1-z}{C-s_i+1} & \text{if } a_i = 0 \text{ and } s_i' \in [s_i, C] \text{ and } s_i \neq 0 \end{cases}$$

2. **3 points:** Consider the case where you have
   (a) $N = 1$ acre of land
   (b) the prices are linear $P(q) = q$
   (c) you have linear utility
   (d) you have a discount factor of $\gamma = 0.9$

   Implement this as a *FiniteMarkovDecisionProcess* and plot the value function as function of q for $C = 100$, $z = 0.1$. What do you notice about the optimal value function and associated optimal policy?

Solution:
See solution file, the optimal policy is to harvest if the quality is greater than or equal to 36 and to leave it alone otherwise.
So, we assume our value function has the following form.

$$V(q = 0) = V_0$$

$$V(q > 0) = \begin{cases} q + \gamma V_0 & \text{if } q \geq 36 \\ \gamma(zV_0 + \frac{1-z}{C-q+q} \sum_{i=q}^{C} V(i)) & \text{otherwise} \end{cases}$$

3. **3 points:** Now, use the results from part 2 to solve this MDP using linear function approximation for one of the RL control algorithms covered in class (i.e. Q-Learning, SARSA, etc.), correctly choosing your features such that you can recover the optimal policy and value function exactly. Justify your choice of features, but you do not need to rigorously prove that they are sufficient to represent the value function exactly. Implement this and again plot the value function and associated optimal policy, and confirm that your solution is converging correctly.

   Note:
   (a) your features may *not* simply be the value function
   (b) your learned value function does not need to have converged *exactly* to the true one, but it should be quite close, your optimal policy should match exactly
   (c) in our implementation, training to this point took no more than a couple of minutes

Solution:

See solution file, we use q-learning.

Choose the feature functions to be

$$\Phi(q,a) = \begin{bmatrix} 1 \\ \mathbb{I}_{q<36}(1-a)q \\ \mathbb{I}_{q<36}aq \\ \mathbb{I}_{q\geq 36}(a) \\ \mathbb{I}_{q\geq 36}(a)q \end{bmatrix}$$

Note that the first feature is technically redundant but it speeds up training

4. **2 points:** Does your solution to the case when $N = 1$ enable you to solve the general case? Prove why or explain why not. Repeat this for the case of log utility. For log utility take $u(x) = \log(1 + x)$ to avoid issues when there are no proceeds for a period.

Solution: It does for the linear case. The expected utility of the set of acres is the same as the sum of the expected utilities of the individual acres, thus maximizing each individually is equivalent to maximizing them jointly.

It does not for the log utility case.

5. **4 points:** Consider now the case where the price offered becomes a function of the quality and some baseline price process, i.e.

$$P_t(q) = P(p_t, q)$$

Where $p_t \in \mathbb{R}$ is a mean-reverting baseline price process and we assume that p_t is observed at time t.

Adapt your MDP to fit this formulation and explain how you would use RL to solve this control problem for a general utility function, given access to historical price trends p_t, knowledge of the pricing function, and the distribution of crop growth. Be specific about the algorithmic choices you would make and why.

Solution: The MDP simply needs to track p_t in the state as well, where the state transitions for the price process are independent of the crop growth.

We accepted a wide variety of approaches, however all fully correct answers needed to identify how to deal with

(a) generating training data / experiences: build a simulator etc.

(b) how to deal with state *and* action space complexity, both grow exponentially in the number of acres