Q1. Feature-Based Q-Learning

1. When using features to represent the Q-function is it guaranteed that the feature-based Q-learning finds the same optimal $Q^*$ as would be found when using a tabular representation for the Q-function?

No, if the optimal Q-function $Q^*$ cannot be represented as a weighted combination of features, then the feature-based representation would not have the expressive power to find it. For example, consider the following MDP with deterministic transitions:

\[
\begin{array}{c}
A \quad 1 \quad B \quad 1 \\
\quad \quad 0 \\
G
\end{array}
\]

With discount $\gamma = 1$, the optimal $Q$ values are $Q(A, \text{right}) = 2$, $Q(B, \text{right}) = 1$, $Q(G, \text{stay}) = 0$.

Suppose we have just one feature $f$, which depends only on states, with value $f(A) = 1$, $f(B) = 2$, and $f(G) = 0$. There’s no linear function that can map the feature values for the states to the optimal Q values above, so it’s not possible for feature-based Q-learning to find the optimal values.
Q2. Q-learning

Consider the following gridworld (rewards shown on left, state names shown on right).

From state A, the possible actions are right(→) and down(↓). From state B, the possible actions are left(←) and down(↓). For a numbered state (G1, G2), the only action is to exit. Upon exiting from a numbered square we collect the reward specified by the number on the square and enter the end-of-game absorbing state X. We also know that the discount factor \( \gamma = 1 \), and in this MDP all actions are deterministic and always succeed.

Consider the following episodes:

<table>
<thead>
<tr>
<th>Episode 1 (E1)</th>
<th>Episode 2 (E2)</th>
<th>Episode 3 (E3)</th>
<th>Episode 4 (E4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( a )</td>
<td>( s' )</td>
<td>( r )</td>
</tr>
<tr>
<td>A ↓</td>
<td>G1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>G1</td>
<td>exit</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

(a) Consider using temporal-difference learning to learn \( V(s) \). When running TD-learning, all values are initialized to zero. For which sequences of episodes, if repeated infinitely often, does \( V(s) \) converge to \( V^*(s) \) for all states \( s \)?

(Imagine appropriate learning rates such that all values converge.)

Write the correct sequence under “Other” if no correct sequences of episodes are listed.

- E1, E2, E3, E4
- E4, E3, E2, E1
- E1, E2, E1, E2
- E1, E2, E3, E1
- Other
- See explanation below

TD learning learns the value of the executed policy, which is \( V^*(s) \). Therefore for \( V^*(s) \) to converge to \( V^*(s) \), it is necessary that the executing policy \( \pi(s) = \pi^*(s) \).

Because there is no discounting since \( \gamma = 1 \), the optimal deterministic policy is \( \pi^*(A) = \downarrow \) and \( \pi^*(B) = \leftarrow (\pi^*(G1) \) and \( \pi^*(G2) \) are trivially exit because that is the only available action). Therefore episodes E1 and E4 act according to \( \pi^*(s) \) while episodes E2 and E3 are sampled from a suboptimal policy.

From the above, TD learning using episode E4 (and optionally E1) will converge to \( V^*(s) = V^*(s) \) for states A, B, G1. However, then we never visit G2, so \( V(G2) \) will never converge. If we add either episode E2 or E3 to ensure that \( V(G2) \) converges, then we are executing a suboptimal policy, which will then cause \( V(B) \) to not converge. Therefore none of the listed sequences will learn a value function \( V^*(s) \) that converges to \( V^*(s) \) for all states \( s \). An example of a correct sequence would be E2, E4, E4, E4,...; sampling E2 first with the learning rate \( \alpha = 1 \) ensures \( V^*(G2) = V^*(G2) \), and then executing E4 infinitely after ensures the values for states A, B, and G1 converge to the optimal values.

We also accepted the answer such that the value function \( V(s) \) converges to \( V^*(s) \) for states A and B (ignoring G1 and G2). TD learning using only episode E4 (and optionally E1) will converge to \( V^*(s) = V^*(s) \) for states A and B, therefore the only correct listed option is E4, E4, E4, E4.
(b) Consider using Q-learning to learn $Q(s, a)$. When running Q-learning, all values are initialized to zero. For which sequences of episodes, if repeated infinitely often, does $Q(s, a)$ converge to $Q^*(s, a)$ for all state-action pairs $(s, a)$?

(Assume appropriate learning rates such that all Q-values converge.) Write the correct sequence under “Other” if no correct sequences of episodes are listed.

- $E1, E2, E3, E4$
- $E4, E3, E2, E1$
- $E3, E4, E3, E4$
- $E1, E2, E4, E4$
- $E1, E2, E1, E2$
- $E1, E2, E3, E1$
- $E4, E4, E4, E4$

Other __________

For $Q(s, a)$ to converge, we must visit all state action pairs for non-zero $Q^*(s, a)$ infinitely often. Therefore we must take the exit action in states $G1$ and $G2$, must take the down and right action in state $A$, and must take the left and down action in state $B$. Therefore the answers must include $E3$ and $E4$. 
Q3. Reinforcement Learning

Imagine an unknown game which has only two states \{A, B\} and in each state the agent has two actions to choose from: \{Up, Down\}. Suppose a game agent chooses actions according to some policy \(\pi\) and generates the following sequence of actions and rewards in the unknown game:

<table>
<thead>
<tr>
<th>t</th>
<th>(s_t)</th>
<th>(a_t)</th>
<th>(s_{t+1})</th>
<th>(r_t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A</td>
<td>Down</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>B</td>
<td>Down</td>
<td>B</td>
<td>-4</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>Up</td>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>B</td>
<td>Up</td>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>A</td>
<td>Up</td>
<td>A</td>
<td>-1</td>
</tr>
</tbody>
</table>

Unless specified otherwise, assume a discount factor \(\gamma = 0.5\) and a learning rate \(\alpha = 0.5\).

(a) Recall the update function of Q-learning is:

\[
Q(s_t, a_t) \leftarrow (1-\alpha)Q(s_t, a_t) + \alpha (r_t + \gamma \max_{a'} Q(s_{t+1}, a'))
\]

Assume that all Q-values initialized as 0. What are the following Q-values learned by running Q-learning with the above experience sequence?

\[
Q(A, \text{Down}) = \frac{1}{4}, \quad Q(B, \text{Up}) = \frac{7}{4}
\]

Perform Q-learning update 4 times, once for each of the first 4 observations.

(b) In model-based reinforcement learning, we first estimate the transition function \(T(s, a, s')\) and the reward function \(R(s, a, s')\). Fill in the following estimates of \(T\) and \(R\), estimated from the experience above. Write “n/a” if not applicable or undefined.

\[
\hat{T}(A, \text{Up}, A) = \frac{1}{2}, \quad \hat{T}(A, \text{Up}, B) = \frac{1}{2}, \quad \hat{T}(B, \text{Up}, A) = \frac{1}{2}, \quad \hat{T}(B, \text{Up}, B) = \frac{1}{2}
\]

\[
\hat{R}(A, \text{Up}, A) = -1, \quad \hat{R}(A, \text{Up}, B) = \text{n/a}, \quad \hat{R}(B, \text{Up}, A) = 3, \quad \hat{R}(B, \text{Up}, B) = 0
\]

(c) To decouple this question from the previous one, assume we had a different experience and ended up with the following estimates of the transition and reward functions:

<table>
<thead>
<tr>
<th>(s)</th>
<th>(a)</th>
<th>(s')</th>
<th>(\hat{T}(s, a, s'))</th>
<th>(\hat{R}(s, a, s'))</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Up</td>
<td>A</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>A</td>
<td>Down</td>
<td>A</td>
<td>0.5</td>
<td>2</td>
</tr>
<tr>
<td>A</td>
<td>Down</td>
<td>B</td>
<td>0.5</td>
<td>2</td>
</tr>
<tr>
<td>B</td>
<td>Up</td>
<td>A</td>
<td>1</td>
<td>-5</td>
</tr>
<tr>
<td>B</td>
<td>Down</td>
<td>B</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

(i) Give the optimal policy \(\hat{\pi}^*(s)\) and \(\hat{\nu}^*(s)\) for the MDP with transition function \(\hat{T}\) and reward function \(\hat{R}\).

Hint: for any \(x \in \mathbb{R}, |x| < 1\), we have \(1 + x + x^2 + x^3 + x^4 + \cdots = 1/(1-x)\).

\[
\hat{\pi}^*(A) = \text{Up}, \quad \hat{\pi}^*(B) = \text{Down}, \quad \hat{\nu}^*(A) = 20, \quad \hat{\nu}^*(B) = 0
\]

Find the optimal policy first, and then use optimal policy to calculate the value function using a Bellman equation.

(ii) If we repeatedly feed this new experience sequence through our Q-learning algorithm, what values will it converge to? Assume the learning rate \(\alpha_t\) is properly chosen so that convergence is guaranteed.

- the values found above, \(\hat{\nu}^*\)
- the optimal values, \(V^*\)
- neither \(\hat{\nu}^*\) nor \(V^*\)
The Q-learning algorithm will not converge to the optimal values $V^*$ for the MDP because the experience sequence and transition frequencies replayed are not necessarily representative of the underlying MDP. (For example, the true $T(A, \text{Down}, A)$ might be equal to 0.75, in which case, repeatedly feeding in the above experience would not provide an accurate sampling of the MDP.) However, for the MDP with transition function $\hat{T}$ and reward function $\hat{R}$, replaying this experience repeatedly will result in Q-learning converging to its optimal values $\hat{V}^*$. 
