Homework Set 2

Problem 1: In the infinite-horizon discounted setting, we derived the equation

$$V^\pi = (I - \gamma P^\pi)^{-1} R^\pi$$

to perform policy evaluation for a deterministic, stationary policy π. Here, $V^\pi \in \mathbb{R}^{|\mathcal{S}|}$ is the value function for all states $s \in \mathcal{S}$ represented as a column vector, $I \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ is an identity matrix, $P^\pi \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ is a probability transition matrix for the induced Markov chain under policy π, and $R^\pi \in \mathbb{R}^{|\mathcal{S}|}$ is the immediate reward for all states $s \in \mathcal{S}$ under policy π.

1. Prove that $I - \gamma P^\pi$ is an invertible matrix.

2. Derive a similar equation for policy evaluation for a stochastic, stationary policy. Specify clearly how each variable in the new equation can be determined.
   [Hint: Start with the Bellman consistency equation and follow similar steps as we did in class.]

3. Derive a similar equation for policy evaluation for a stochastic, stationary policy when the reward function is stochastic, i.e., $R : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$. Specify clearly how each variable in the new equation can be determined.
   [Hint: Start with the definition of the value function in this setting to arrive at a slightly modified Bellman consistency equation and then follow similar steps as we did in class.]

Problem 2: In the proof of the theorem on Bellman optimality, we used two inequalities that will be proved in this problem (the first one also reappeared in the proofs related to the value iteration and the policy iteration algorithms).

1. Let $g_1 : X \to \mathbb{R}$ and $g_2 : X \to \mathbb{R}$ denote two scalar-valued functions defined over the same domain. Show that

   $$\left| \max_{x \in X} g_1(x) - \max_{x \in X} g_2(x) \right| \leq \max_{x \in X} |g_1(x) - g_2(x)| .$$

   [Hint: Visualizing two such functions, e.g., defined over a finite, discrete domain may help.]

2. Show that the result of joint maximization is higher than or equal to that of sequential maximization. In particular, show that

   $$\max_{x \in X, y \in Y} f(x, g(y)) \geq \max_{x \in X} f(x, \max_{y \in Y} g(y)),$$

   for two scalar-valued functions $f : X \times Z \to \mathbb{R}$ and $g : Y \to Z \subseteq \mathbb{R}$.
   [Hint: Use the fact that for a function $h : W \to \mathbb{R}$, $\max_{w \in W} h(w) \geq h(w')$ for any $w' \in W$.]

Problem 3: Consider a Markov decision process (MDP) in the infinite-horizon discounted setting with state space $\mathcal{S} = \{b, c\}$, action space $\mathcal{A} = \{x, y\}$, transition function $P(s'|s, a)$ with

$$\begin{aligned} P(b|b,x) &= 1.0\,, \quad P(c|b,x) = 0.0\,, \\ P(b|b,y) &= 0.2\,, \quad P(c|b,y) = 0.8\,, \\ P(b|c,x) &= 0.0\,, \quad P(c|c,x) = 1.0\,, \\ P(b|c,y) &= 0.6\,, \quad P(c|c,y) = 0.4\,, \end{aligned}$$

reward function $R(s, a)$ with

$$\begin{aligned} R(b,x) &= 0\,, \quad R(b,y) = 0\,, \\ R(c,x) &= 1\,, \quad R(c,y) = 1\,, \end{aligned}$$

and discount factor γ.

1. Suppose we are running a value iteration algorithm over the known MDP model to find the optimal value function. If at iteration $t = 6$, the value function is

   $$V_6(b) = 10\,, \quad V_6(c) = 5\,,$$

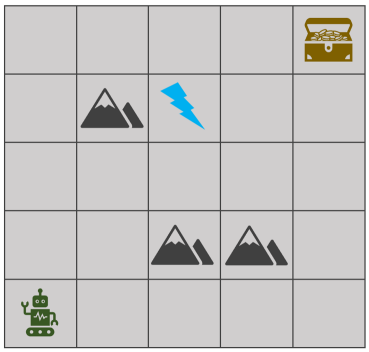
   what will the value function be at iteration $t = 7$? The answer will be in terms of γ.

2. Suppose we are running a policy iteration algorithm over the known MDP model to find an optimal policy. If at iteration $t = 8$, the value function for policy π_8 is

   $$V^{\pi_8}(b) = 5\,, \quad V^{\pi_8}(c) = 15\,,$$

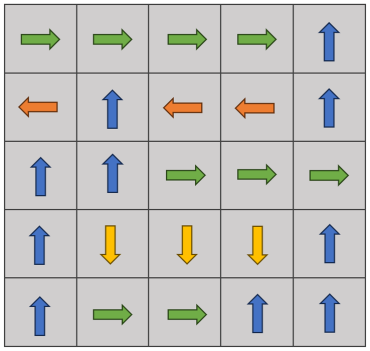
   what will the policy be at iteration $t = 9$?

Problem 4: Consider an MDP over the grid-world illustrated below, where the goal is to take the agent to the treasure chest while avoiding the lightning.



- The state space contains the cells of the grid-world.
- The agent starts at the bottom left corner.
- The action space is $\{\text{up}, \text{down}, \text{left}, \text{right}\}$.
- The agent can only move to its adjacent cells, i.e., the cells that are above, below, to the left, or to the right of its current cell. If the agent is not at the boundary cells, each action will take the agent to the expected cell with probability 0.85 and to one of the remaining three cells, each with probability 0.05. If the agent is at the boundary, it will remain at its current cell with the sum of probabilities that would have taken it outside the grid-world. The cells with a mountain cannot be accessed, i.e., if adjacent to the mountain, the agent will remain at its current cell with the probability that would have taken it to the mountain. All actions in the cell with the lightning bolt, the one with the treasure chest, and the ones with a mountain will keep the agent in its current cell.
- The agent receives a reward of 0 in every cell for all actions except two cells. If at the cell with a lightning bolt, it will receive a reward of -1 for all actions, and if at the cell with the treasure chest, it will receive a reward of $+1$ for all actions.
- The discount factor is 0.95.

1. Evaluate the deterministic, stationary policy shown in the figure below, where each arrow represents the prescribed action in each cell.



   (a) Apply the analytical solution for policy evaluation. Report the value function at all states.

   (b) Implement and run the iterative solution for approximate policy evaluation with a zero initialization for the value function. Pick the number of iterations in a way to ensure 0.01 accuracy in the final computed value function, i.e., $\|V_T - V^\pi\|_\infty \leq 0.01$ without using the results of the previous part. Justify how you picked the number of iterations. Report the value function at all states.

   (c) Plot the sequence of errors in the value function from the approximate policy evaluation with respect to the iterations. In particular, plot $\|V_t - V^\pi\|_\infty$ against $t \in \mathbb{N}_0$, where V_t is the value function at iteration t and V^π is the value function computed from the analytical solution.

2. Implement and run a value iteration algorithm to compute an optimal policy in this MDP. Initialize the value function at zero. Pick the number of iterations in a way to ensure 0.01 accuracy in the final computed value function. Demonstrate the learned policy and report the value function at all states.

3. Implement and run a policy iteration algorithm to compute an optimal policy in this MDP. Initialize the policy randomly, by using a uniform distribution over the actions. Pick the number of iterations in a way to ensure 0.01 accuracy in the final computed value function. Demonstrate the learned policy and report the value function at all states.

[**Note:** For this problem, attach all your code in a programming language of your choice to the end of your submission.]