Stanford RL Pset 1

Andrew Koulogeorge

January 2024

1 Reward choices [15 pts]

Consider a tabular MDP with $0 < \gamma < 1$ and no terminal states. The agent will act forever. The original optimal value function for this problem is V_1^* and the optimal policy is π_1^*.

(a) Now someone decides to add a small reward bonus c to all transitions in the MDP. This results in a new reward function $\hat{r}(s, a) = r(s, a) + c;\ \forall s, a$ where $r(s, a)$ is the original reward function. What is an expression for the new optimal value function? Can the optimal policy in this new setting change? Why or why not? [5 pts]

Solution: With the new reward function $\hat{r}(s, a) = r(s, a) + c$, all rewards received by an agent are increased by a constant factor c. I claim that this new reward function does not change the optimal policy for the MDP. Consider any policy, π in the tabular MDP with the updated reward function. By definition of the state-value function, we know that:

$$\hat{v}_\pi(s) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k \times \hat{R}_{t+k+1} | S_t = s\right]$$

where $\hat{R}_{t+k+1}$ is the expected reward under policy π in our MDP at time step $t + k + 1$ with the new reward function. $\hat{v}_\pi(s)$ is the value of the policy under the new reward function and $v_\pi(s)$ is the value of the value of the policy under the original reward function.

We know that $\hat{R}_{t+k+1} = R_{t+k+1} + c$. Expanding out our definition for the state-value function, we see that:

$$\begin{aligned}
\hat{v}_\pi(s) &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k \times (R_{t+k+1} + c) | S_t = s\right] \\
&= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} (\gamma^k \times R_{t+k+1}) + (\gamma^k \times c) | S_t = s\right] \\
&= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} (\gamma^k \times R_{t+k+1}) + \sum_{k=0}^{\infty} (\gamma^k \times c) | S_t = s\right]
\end{aligned}$$

By linearity of expectation, we know that for any two random variables X and Y, we know that

$$\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$$

which implies we can break out these two terms in our expectation of the policy π:

$$\hat{v}_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \times R_{t+k+1} | S_t = s \right] + \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \times c | S_t = s \right]$$

From here, we observe that the first term is simply the value of the policy π in the MDP with the original reward function and the right hand term is a constant that is not effected by the stochasticity of the MDP:

$$\hat{v}_\pi(s) = v_\pi(s) + \frac{c}{1-\gamma}$$

Thus, for any given policy π, the value of that policy in the new MDP can be expressed as the value of that policy in the old MDP plus a constant.

Recall the definition for the optimal state-value function:

$$v_*(s) = \max_\pi v_\pi(s), \ \forall s \in S$$

Thus, the expression for the new optimal value function will be the same as the original optimal value function except the optimal value of each state will be increased by $\frac{c}{1-\gamma}$. The optimal policy, however, wont change at all since its the argmax over all policies of the value of that policy.

$$\hat{\pi}_* = \arg\max_\pi \hat{v}_\pi(s) = \pi_*$$

Thus, the optimal policy in the new MDP is the same as the optimal policy in the old MDP $\square$

(b) Instead of adding a small reward bonus, someone decides to multiply all rewards by an arbitrary constant $c \in \mathbb{R}$. This results in a new reward function $\hat{r}(s,a) = c \times r(s,a); \ \forall s, a$ where $r(s,a)$ is the original reward function. Are there cases in which the new optimal policy is still π_1^* and the resulting value function can be expressed as a function of c and V_1^* (if yes, give the resulting expression and explain for what value(s) of c and why? If not, explain why not)? Are there cases where the optimal policy would change (if yes, provide a value of c and a description of why the optimal policy would change. If not, explain why not)? Is there a choice of c such that all policies are optimal? [5 pts]

Solution: With the new reward function $\hat{r}(s,a) = r(s,a) \times c$, we can express the value of a policy π under this new MDP as:

$$\hat{v}_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \times c \times R_{t+k+1} | S_t = s \right]$$

Using basic expectation laws, we know that given a random variable X and a real value α:

$$\mathbb{E}[\alpha X] = \alpha \mathbb{E}[X]$$

Applying this to our expectation over our policy and MDP, we see that:

$$\hat{v}_\pi(s) = c \times \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \times R_{t+k+1} | S_t = s \right]$$

$$\hat{v}_\pi(s) = c \times v_\pi(s)$$

Note that the value of c effects what the optimal policy will be for the new MDP.

- When $c > 0$, we reduce to the case in part (a) where the orderings of the rewards will be the same so our optimal policy will choose the same actions, resulting in the same in both MDPs.
- When $c < 0$, the optimal policy can change because actions that use to be very desirable are not very bad. As an example, if we applied a negative constant to the reward function for chess where the agent receives a 1 when checkmating and a -1 when getting checkmated, the new optimal policy would act to get checkmated as soon as possible.
- When $c = 0$, all policies are optimal since each action yields the same amount of reward of 0.

□

(c) If the MDP instead has terminal states that can end an episode, does that change your answer to part (a)? If yes, provide an example MDP where your answers to part (a) and this part would differ. [5 pts]

Solution: Yes, if we include terminal states into our MDP the optimal policy can change if we apply a constant translation to the reward function. For example, consider an MDP with deterministic reward and deterministic transition dynamics with single starting state s_{start} and and a single ending state s_{end}. Let there be 2 paths from s_{start} to s_{end} where one of the paths has 1 state in-between and the other path has a 10 states in-between. If the reward on the single path is very large and the reward in-between each of the states on the long path is small, the optimal policy would be to take the action leading down the shorter path. Now, consider an updated reward function where we add k to each of the rewards where k is larger than the reward received on the shorter path. Now, the optimal policy would be to take the longer path because the additional reward given to each of the states, k, overpowers the reward from taking the short path. □

2 Bellman Residuals and performance bounds [25 pts]

In this problem, we will study Bellman residuals, which we will define below, and how it helps us bound the performance of a policy. But first, some recap and definitions.

Definitions: From lecture, we know that the Bellman backup operator B, defined below is a contraction with the fixed point as V^*, the optimal value function of the MDP. The symbols have

their usual meanings. γ is the discount factor and $0 \leq \gamma < 1$. In all parts, $||v||$ is the infinity norm of the vector.

$$(BV)(s) = \max_a [r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a)V(s')]$$

We also saw the contraction operator B^π with the fixed point V^π, which is the Bellman backup operator for a particular policy given below:

$$(B^\pi V)(s) = \mathbb{E}_{a \sim \pi}[r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a)V(s')]$$

In class, we showed that $||BV - BV'|| \leq \gamma ||V - V'||$ for two arbitrary value functions V and V'.

For the rest of this question, you can also assume that $||B^\pi V - B^\pi V'|| \leq \gamma ||V - V'||$. The proof of this is very similar to the proof we saw in class for B.

(a) Prove that the fixed point for B^π is unique. [3 pts]

Solution: Given above, we can assume for the contraction operator B^π that $||B^\pi V - B^\pi V'|| \leq \gamma ||V - V'||$ holds. This is the definition of a strict contraction, so we can invoke the the contraction mapping theorem from lecture, which states that a function $T : V \to V$ which is a strict contraction has a unique fixed point in V. □

We can recover a greedy policy π from an arbitrary value function V using the equation below.

$$\pi(s) = \arg\max_a [r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a)V(s')]$$

(b) When π is the greedy policy, explain what is the relationship between B and B^π? [2 pts]

Solution: When π is a greedy policy, I claim that the the operators B and B^π are identical. When applying the operator B^π to a value function, we update the value of each state based on the distribution of actions taken by policy π in a given state. When this policy becomes greedy, however, the action we take at each time step becomes deterministic based on the largest value of a given action. The update step can still be thought of as a distribution over actions but now all the probability mass is placed on the action yielding the max reward. Indeed, this is simply the Bellman backup operator B □

Motivation: It is often helpful to know what the performance will be if we extract a greedy policy from a value function. In the rest of this problem, we will prove a bound on this performance.

Recall that a value function is a $|S|$-dimensional vector where $|S|$ is the number of states of the MDP. When we use the term V in these expressions as an "arbitrary value function", we mean that V is an arbitrary $|S|$-dimensional vector which need not be aligned with the definition of the MDP at all. On the other hand, V^π is a value function that is achieved by some policy π in the MDP. For example, say the MDP has 2 states and only negative immediate rewards. $V = [1, 1]$ would be a valid choice for V even though this value function can never be achieved by any policy π, but we

can never have a $V^\pi = [1, 1]$. This distinction between V and V^π is important for this question and more broadly in reinforcement learning.

Why do we care about setting V to vectors than can never be achieved in the MDP? Sometimes algorithms, such as Deep Q-networks, return such vectors. In such situations we may still want to extract a greedy policy π from a provided V and bound the performance of the policy we extracted aka V^π.

Bellman Residuals: Define the Bellman residual to be $(BV - V)$ and the Bellman error magnitude to be $||BV - V||$. As we will see through the course, this Bellman residual is an important component of several popular RL algorithms such as the Deep Q-networks, referenced above. Intuitively, you can think of it as the difference between what the value function V specifies at a state and the one-step look-ahead along the seemingly best action at the state using the given value function V to evaluate all future states (the BV term).

(c) For what V does the Bellman error magnitude equal 0? [1 pts]

Solution: The Bellman error magnitude equals 0 when $BV = V$ which is at the fixed point of the Bellman operator, V^*, the optimal value function. □

(d) Prove the following statements for an arbitrary value function V and any policy π. [5 pts]
Hint: Try leveraging the triangle inequality by inserting a zero term.

$$||V - V^\pi|| \leq \frac{||V - B^\pi V||}{1 - \gamma}$$

$$||V - V^*|| \leq \frac{||V - BV||}{1 - \gamma}$$

Solution: Given an arbitrary value function V and any policy π:

$$||V - V^\pi|| = ||V - B^\pi V + B^\pi V - V^\pi|| \leq ||V - B^\pi V|| + ||B^\pi V - V^\pi||$$

Given operator B^π, we know that vector V^π, the value function for policy π, is a fixed point for B^π. Thus,

$$||V - V^\pi|| \leq ||V - B^\pi V|| + ||B^\pi V - B^\pi V^\pi||$$

Since B^π is a strict contraction operator, we have that:

$$||V - B^\pi V|| + ||B^\pi V - B^\pi V^\pi|| \leq ||V - B^\pi V|| + \gamma||V - V^\pi||$$

$$||V - V^\pi|| \leq ||V - B^\pi V|| + \gamma||V - V^\pi||$$

$$||V - V^\pi|| \leq \frac{||V - B^\pi V||}{1 - \gamma}$$

The exact same proof can be applied in the case when the policy π is greedy with respect to the value function. In this case, $B^\pi = B$ and $V^\pi = V^*$ and we get the second result above. □

The result you proved in part (d) will be useful in proving a bound on the policy performance in the next few parts. Given the Bellman residual, we will now try to derive a bound on the policy performance, V^π.

(e) Let V be an arbitrary value function and π be the greedy policy extracted from V. Let $\varepsilon = ||BV - V||$ be the Bellman error magnitude for V. Prove the following for any state s. [5 pts]
Hint: Try to use the results from part (d).

$$V^{\pi}(s) \geq V^{*}(s) - \frac{2\varepsilon}{1-\gamma}$$

Solution: From the previous question, we know that for a greedy policy π,

$$||V - V^{\pi}|| \leq \frac{||V - B^{\pi}V||}{1-\gamma}$$

Recall that if π is a greedy policy then $B = B^{\pi}$- the optimal bellman operator is equal to the bellman operator with respect to the greedy policy π.

$$||V - V^{\pi}|| \leq \frac{||V - BV||}{1-\gamma}$$

Now, consider the relationship between the optimal value function and B shown in the previous question:

$$||V - V^{*}|| \leq \frac{||V - BV||}{1-\gamma}$$

Let $\varepsilon = ||V - BV||$ and add the previous 2 inequalities together to yield:

$$||V - V^{\pi}|| + ||V - V^{*}|| \leq \frac{2 * \varepsilon}{1-\gamma}$$

By the triangle inequality, we have that:

$$||V^{*} - V^{\pi}|| \leq \frac{2 * \varepsilon}{1-\gamma}$$

Note that this norm between the optimal value function and the value of policy π is the L_{∞} norm:

$$\max_{s}[|V^{*}(s) - V^{\pi}(s)|] \leq \frac{2 * \varepsilon}{1-\gamma}$$

From the definition of the L_{∞} norm:

$$|V^{*}(s) - V^{\pi}(s)| \leq \frac{2 * \varepsilon}{1-\gamma}, \ \forall \ s \in S$$

By the definition of the optimal value function, we know that $V^{*}(s) \geq V^{\pi}(s) \implies V^{*}(s) - V^{\pi}(s) \geq \ 0 \ \forall \ s \in S$:

$$V^{*}(s) - V^{\pi}(s) \leq \frac{2 * \varepsilon}{1-\gamma}$$

$$V^{*}(s) - \frac{2 * \varepsilon}{1-\gamma} \leq V^{\pi}(s)$$

□

A little bit more notation: define $V \leq V'$ if $\forall s$, $V(s) \leq V'(s)$.

What if our algorithm returns a V that satisfies $V^* \leq V$, i.e., it returns a value function that is better than the optimal value function of the MDP. Once again, remember that V can be any vector, not necessarily achievable in the MDP but we would still like to bound the performance of V^π where π is extracted from said V. We will show that if this condition is met, then we can achieve an even tighter bound on policy performance.

(f) Using the same notation and setup as part (e), if $V^* \leq V$, show the following holds for any state s. [5 pts]
Hint: Recall that $\forall \pi$, $V^\pi \leq V^*$. (why?)

$$V^\pi(s) \geq V^*(s) - \frac{\varepsilon}{1-\gamma}$$

Solution: Following the setup where had in the previous question, we know that:

$$||V - V^\pi|| \leq \frac{\varepsilon}{1-\gamma}$$

$$\max_s[|V(s) - V^\pi(s)|] \leq \frac{\varepsilon}{1-\gamma}$$

$$V(s) \leq V^\pi + \frac{\varepsilon}{1-\gamma}$$

By assumption, the optimal value function V^* is smaller than V:

$$V^*(s) \leq V^\pi + \frac{\varepsilon}{1-\gamma} \implies V^*(s) - \frac{\varepsilon}{1-\gamma} \leq V^\pi(s)$$

□

(g) It's not easy to show that the condition $V^* \leq V$ holds because we often don't know V^* of the MDP. Show that if $BV \leq V$ then $V^* \leq V$. Note that this sufficient condition is much easier to check and does not require knowledge of V^*. [4 pts]
Hint: Try to apply induction. What is $\lim_{n\to\infty} B^n V$?

Solution: First, I will show that B is monotonic. Then, I will use induction to show that $B^k V \leq V \ \forall \ k \in \mathbb{N}$. Let X and Y be any 2 vectors in $\mathbb{R}^{|S|}$ such that $X \leq Y$ where $X \leq Y \iff X[i] \leq Y[i] \forall\ i$. I claim that $X < Y \implies BX < BY$. This follows from the definition of the optimal bellman operator B:

$$(BX)(s) = \max_a[r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a)X(s')]$$

$$(BY)(s) = \max_a[r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a)Y(s')]$$

For a given state s, the terms that differ are the state values for X and Y and we know by assumption that $X[i] \leq Y[i] \ \forall\ i \implies BX \leq BY$. Now, I claim that $B^k V \leq V \ \forall \ k \in \mathbb{N}$. In

the base case when $k = 1$ we have that $BV \leq V$ which is true by assumption. Assume for the Inductive Hypothesis that $B^k V \leq V$. We want to show that $B^{k+1} V \leq V$.

$$B^k V \leq V \implies B(B^k)V \leq BV$$

Since B is monotonic,

$$B(B^k)V = B^{k+1}V \leq BV \leq V \quad \square$$

Since $\lim_{k\to\infty} B^k V = V^*$, by applying the above fact that the repeated application of the bellman operator is monotonic, we now know that $V^* \leq V$ $\square$

3 Frozen Lake MDP [25 pts]

Now you will implement value iteration and policy iteration for the Frozen Lake environment from Gymnasium and originally from OpenAI Gym. We have provided custom versions of this environment in the starter code.

Setup: This assignment needs to be completed with Python3 (we recommend versions >= 3.8). We have provided a `requirements.txt` file that contains all the associated Python libraries you will need to complete this assignment. You can go through the file and install all the libraries by running `pip install -r requirements.txt` in the terminal. We highly recommend that you do this in a virtual environment dedicated for this assignment to avoid any dependency conflicts. **While programming, please ignore any `DeprecationWarning` you may see**.

Submission: There is a `Makefile` provided that will help you submit the assignment. Please run `make clean` followed by `make submit` in the terminal and submit the resulting zip file on Gradescope.

(a) **(coding)** Read through `vi_and_pi.py` and implement `policy_evaluation`, `policy_improvement` and `policy_iteration`. The stopping tolerance (defined as $\max_s |V_{old}(s) - V_{new}(s)|$) is tol = 10^{-3} . Use $\gamma = 0.9$. Return the optimal value function and the optimal policy. [10pts]

Solution: See code

(b) **(coding)** Implement `value_iteration` in `vi_and_pi.py`. The stopping tolerance is tol = 10^{-3}. Use $\gamma = 0.9$. Return the optimal value function and the optimal policy. [10 pts]
Solution: See Code.

(c) **(written)** Run both methods on the Deterministic-4x4-FrozenLake-v0 and Stochastic-4x4-FrozenLake-v0 environments. In the second environment, the dynamics of the world are stochastic. How does stochasticity affect the number of iterations required, and the resulting policy, quantitatively and qualitatively? [5 pts]

Solution: When running the above algorithms on the deterministic MDP, it took 7 steps for my policy iteration to converge. My policy iteration and my value iteration algorithm both converged to the same value function and policy:

Value Function (Deterministic):

$$[0.59, 0.656, 0.729, 0.656, 0.656, 0, 0.81, 0, 0.729, 0.81, 0.9, 0, 0, 0.9, 1, 0$$

Policy (Deterministic):

$$[1, 2, 1, 0, 1, 0, 1, 0, 2, 1, 1, 0, 0, 2, 2, 0]$$

For the stochastic MDP, it took 6 steps for my policy iteration to converge.

Value Function (Stochastic):

$$[0.064, 0.058, 0.072, 0.053, 0.088, 0, 0.111, 0, 0.143, 0.246, 0.299, 0, 0, 0.379, 0.639, 0$$

Policy (Stochastic):

$$[0, 3, 0, 3, 0, 0, 0, 0, 3, 1, 0, 0, 0, 2, 1, 0]$$

We see that the policies for the Stochastic MDP and the Deterministic MDP are different, with the deterministic policy simply learning to take the best action in each state which will move them towards the goal state. The Stochastic policy, on the other hand, has to take into account the fact that you can move perpendicular to the intended direction which can result in the agent falling in ice. As a result, the agent learns to be more conservative and wants to avoid falling in the ice. For example, consider the policy the agent learns in the first state. It learns to move left, which has a two-thirds chance of the agent staying still and a one-third chance of the agent moving downward, which is the direction of the goal. In this way, the agent learned a policy to mitigate the negative effects of the stochastic transition dynamics and ensure it never falls into ice on the first move. □