CME241 Assignment4

Quinn Hollister

January 2022

1 Problem 1

Manually calculate $q_l(.,.)$ and then calculate the greedy policy for $l = 1, 2$

The formula for $q_l(s_i, a_i)$ will be defined as follows:

$$q_l(s_i, a_j) = \mathcal{R}(s_i, a_i) + \sum_{k \in 1,2,3} (\mathcal{P}(s_i, a_j, s_k) \cdot V_{l-1}(s_k)) \tag{1}$$

As can be easily seen, this is an iterative scheme, and I will show in a table how the values for q depend on the state, action space as well as which iteration index it's at:

Iterative Q functions		
itr index	action 1	action2
i = 1	$q_1(s_1, a_1) = 10.6$	$q_1(s_1, a_2) = 11.2$
i = 1	$q_1(s_2, a_1) = 4.3$	$q_1(s_2, a_2) = 4.3$
i = 2	$q_2(s_1, a_1) = 12.82$	$q_2(s_1, a_2) = 11.98$
i = 2	$q_2(s_2, a_1) = 5.65$	$q_2(s_2, a_2) = 5.89$

The second iteration uses the updated value function values where we use the max of the q values for each state. We can also show the greedy policy improvement's as a function of the q values.

$$V_l(s_i) = \max_a q_l(s_i, a) \tag{2}$$

$$\pi(s_i) = arg \max q_l(s_i, a) \tag{3}$$

Thus, we get the following value functions and greedy policy as a function of our iteration index. We broke ties by choosing the action with the smallest index.

$$V_1(s) = (11.2, 4.3, 0)^T, \pi_1(s) = (a_2, a_1)^T$$
$$V_2(s) = (12.82, 5.65, 0)^T, \pi_2(s) = (a_1, a_2)^T$$

Now argue that the greedy policy will be invariant under future iterations

We know that the greedy policy for a given iteration depends on whether action 1 or action 2 results in a greater q-value for a given s. Thus, in order for the current π not to change, we need to show that for state 1, the action 1 will always be greater than action 2, and the reverse for state 2. In equations, this is equivalent to:

$$\begin{aligned} q_{i-1}(s_1, a_1) &\overset{?}{\geq} q_{i-1}(s_1, a_2) \\ q_{i-1}(s_2, a_1) &\overset{?}{\leq} q_{i-1}(s_2, a_2) \end{aligned} \tag{4}$$

which is equivalent to

$$\begin{aligned} v_{i-1}(s_1) + 4v_{i-1}(s_2) &\overset{?}{\geq} 20 \\ v_{i-1}(s_1) &\overset{?}{\leq} 10 \end{aligned} \tag{5}$$

In order to show that these are true, we need to understand how the value function changes as we keep applying the bellman optimality operator. When we apply the operator, we take a maximum over the actions of the q-values. These q-values are themselves monotonically increasing functions of the previous non-negative value functions. Thus, this new value function must have values greater than the previous for each state. In other words, for a fixed model of transition probabilities and reward functions, $V_i(s) \leq V_{i+1}(s)$. Thus, our conditions for a non-changing policy function are satisfied for i = 2, since $v_1(s_1) = 11.2, v_1(s_2) = 4.3$ which satisfies our system of inequalities. Also, our inequalities will be satisfies for any value function with components as large or larger than our current $V_1(s)$ values. Thus, for any iteration index larger than or equal to 2, the greedy policy will not change, and this must be the optimal policy.

2 Problem 2

Model this two-stores inventory control problem and verify that the optimal policy makes intuitive sense for different parameters

I was able to model the two-store inventory control problem in a similar fashion as the one-store model, but instead with a 4-tuple as the state space, i.e. $s = \{\alpha_1, \beta_1, \alpha_2, \beta_2\}$, and the action space as a 3-tuple, i.e. $a = \{swap, order_1, order_2\}$, where swap may take on values from $-\alpha_1$ to α_2. This number represents the number of bicycles we swap from store B to store A. The added constraint implicit in the problem, is that at the end of the day the store's inventory position may not exceed the store's capacity.

I ran this model on the value iteration algorithm on different parameters: different store capacities, different poisson demand parameters for each store, different holding, stockout, transportation and supplier costs. I tested the resulting optimal policies on some asymptotic cases to test that the intuitive results held. For example, I modeled with both stores having limited capacity, and zero transfer cost. The difference between the two stores was that one had a very

high stockout cost relative to the error, meaning we should expect the optimal policy to transfer as many bikes to store B as possible, if store A had any to give, and so long as $\alpha_B + \beta_B - swap$ did not exceed the store capacity.

The value iteration result function gave policies that mirrored this expectation. For example, under the state $3, 0, 0, 0$ the optimal policy was $-3, 2, 0$, i.e. transfer all of the store A capacity to store B, and order 2 bikes to store A. We can't order any bikes to store B, because then we might exceed store capacity the following evening.

Similar intuitive conclusions followed for asymptotic cases. You may play with the parameter space by accessing the jupyter notebook.