Ex 3

Saturday, October 8, 2022 6:23 PM

Submission by: Aadhar Bansal

Discussed with: Christopher Anton Dominic, Dhavan Sanghvi

1. **1 point.** (RL2e 3.25 – 3.29) *Fun with Bellman.*
Written:

(a) Give an equation for v_* in terms of q_*.
(b) Give an equation for q_* in terms of v_* and the four-argument p.
(c) Give an equation for π_* in terms of q_*.
(d) Give an equation for π_* in terms of v_* and the four-argument p.
(e) Rewrite the four Bellman equations for the four value functions (v_π, v_*, q_π, q_*) in terms of the three-argument function p (Equation 3.4) and the two-argument function r (Equation 3.5).

a) $V_*(s) = \max_a q_*(s,a)$

b) $q_*(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma V_*(s')\right]$

c) $\pi_*(s) = \arg\max_a q_*(s,a)$

d) $\pi_* = \arg\max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma V_*(s')\right]$

e) Bellman eqⁿ

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma V_\pi(s')\right]$$

$$p(s'|s,a) \doteq \Pr\{S_t = s' \mid S_{t-1}=s, A_{t-1}=a\} = \sum_{r\in\mathcal{R}} p(s',r|s,a). \quad \text{— (3.4)}$$

$$r(s,a) \doteq \mathbb{E}[R_t \mid S_{t-1}=s, A_{t-1}=a] = \sum_{r\in\mathcal{R}} r \sum_{s'\in\mathcal{S}} p(s',r|s,a), \qquad (3.5)$$

$$V_\pi(s) = \sum_a \pi(a|s)\left[\sum_{s',r}\left[p(s',r|s,a)\, r\right] + \sum_{s',r}\left[p(s',r|s,a)\,\gamma(v_\pi(s'))\right]\right]$$

$$V_\pi(s) = \sum_a \pi(a|s)\left[r(s,a) + \sum_{s'}(s'|s,a)\,\gamma\,(V_\pi(s'))\right]$$

$$V_*(s) = \max_a\left[r(s,a) + \sum_{s'}(s'|s,a)\,\gamma\,(V_\pi(s'))\right]$$

$$q_\pi(s,a) = r(s,a) + \sum_{s'} p(s'|s,a)\cdot\gamma \sum_{a'} \pi(a'|s')\, q_\pi(s',a')$$

$$q_*(s,a) = r(s,a) + \sum_{s'} p(s'|s,a)\,\gamma \max_a q_*(s',a')$$

Q.2

2. **1 point.** (RL2e 4.5, 4.10) *Policy iteration for action values.*
Written:

(a) How would policy iteration be defined for action values? Give a complete algorithm for computing q_*, analogous to that on page 80 for computing v_*. Please pay special attention to this exercise, because the ideas involved will be used throughout the rest of the book.
(b) What is the analog of the value iteration update Equation 4.10 for action values, $q_{k+1}(s,a)$?

Initialization:

$Q(s,a) \in \mathbb{R}$ & $\pi(s) \in A(s)$ arbitrarily $\forall$ all $s \in C$, $a \in A$

Policy Evaluation:
Loop:
$\Delta \leftarrow 0$
Loop for each $s \in S$ & $s \in A$
$q = Q(s,a)$
$Q(s,a) \leftarrow \sum_{s',r} p(s',r|s,a)\left[r + \gamma \sum_{a'} \pi(a'|s')\, Q(s',a')\right]$
$\Delta \leftarrow \max(\Delta, |q - Q(s,a)|)$
until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

Policy Improvement
policy-stable $\leftarrow$ true
For each $s \in S$ & $a \in A$:
old-action $\leftarrow \pi(s)$
$\pi(s) \leftarrow \arg\max_a Q(s,a)$
if old-action $\neq \pi(s)$, then policy-stable $\leftarrow$ false

if policy-stable, then stop return $Q = q_*$ & $\pi = \pi_*$ else goto policy Evaluation

2 (b)

$$q_{k+1}(s,a) = E\left[R_{t+1} + \max_{a'} \gamma\, q_k(s',a')\right]$$

$$= \sum_{s',r} p(s',r|s,a)\left[r + \max_a \gamma\, q_k(s',a')\right]$$

Q.3

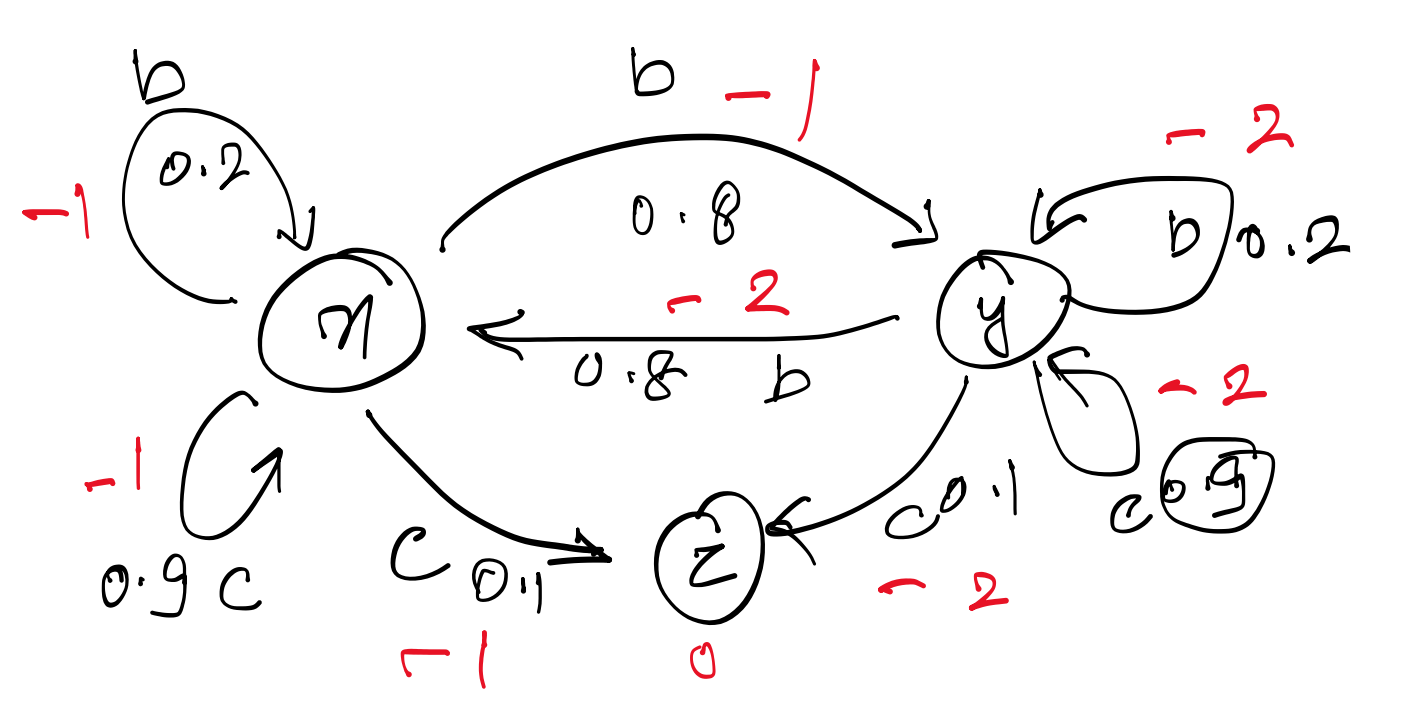
3. **2 points.** *Policy iteration by hand.*
Written: Consider an undiscounted MDP having three states, x, y, z. State z is a terminal state. In states x and y there are two possible actions: b and c. The transition model is as follows:

- In state x, action b moves the agent to state y with probability 0.8 and makes the agent stay put (at state x) with probability 0.2.
- In state y, action b moves the agent to state x with probability 0.8 and makes the agent stay put (at state y) with probability 0.2.
- In either state x or state y, action c moves the agent to state z with probability 0.1 and makes the agent stay put with probability 0.9.

The reward model is as follows:

- In state x, the agent receives reward -1 regardless of what action is taken and what the next state is.
- In state y, the agent receives reward -2 regardless of what action is taken and what the next state is.

$S^+ \{x, y, z\}$
$z \equiv$ terminal state
$(b, c) \in A$



Answer the following questions:

(a) What can be determined *qualitatively* about the optimal policy in states x and y (i.e., just by looking at the transition and reward structure, *without* running value/policy iteration to solve the MDP)?
(b) Apply policy iteration, showing each step in full, to determine the optimal policy and the values of states x and y. Assume that the initial policy has action c in both states.
(c) What happens to policy iteration if the initial policy has action b in both states? Does discounting help? Does the optimal policy depend on the discount factor (in this particular MDP)?

a) As reward for terminal state z is 0 its better if x & y tries to achieve it

for y moving to state x is also beneficial as the reward for x is -1 & for y = -2

As the action (y) $\xrightarrow{b}$ (x) (less -ve reward state)
∴ it is better for y to have action b
& for x taking action would be problematic & c would be better option
∴ policy would be choosing action b for y & action c for x

b) Policy evaluation

Bellman eqⁿ $\Rightarrow V_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma V_\pi(s')\right]$

Assumption: Action c in both state

$V_{\pi_c}(x) = 0.1 * (-1+0) + 0.9 * (-1 + V_{\pi_c}(x))$
$V_{\pi_c}(y) = 0.1 * (-2+0) + 0.9 * (-2 + V_{\pi_c}(y))$
$V_{\pi_c}(x) = -0.1 + (-0.9 + 0.9 V_{\pi_c}(x))$
$V_{\pi_c}(y) = -0.2 + (-1.8 + 0.9 V_{\pi_c}(y))$
⇓
$-1 = 0.1 V_{\pi_c}(x)$
$V_{\pi_c}(x) = -10$ $\quad \pi(x) \to c$
$-2 = 0.1 V_{\pi_c}(y)$ $\quad \pi(y) \to c$
$V_{\pi_c}(y) = -20$

Policy improvement

when apply policy b
$V_\pi b(x) = 0.2 * (-1 + V_\pi c(x)) + 0.8 (-1 + V_\pi c(y))$
$V_\pi b(y) = 0.2 * (-2 + V_\pi c(y)) + 0.8 (-2 + V_\pi c(x))$

$V_\pi b(x) = -19 \quad < \quad V_\pi c(x) = -10$
$V_\pi b(y) = -14 \quad > \quad V_\pi c(y) = -20$

therefore following actions for policy for x & action b for y
$\pi'(x) \to c$
$\pi'(y) \to b$

$\pi' \neq \pi \longrightarrow$ again Policy Evaluation

Policy Evaluation

$V_{\pi_c}(x) = 0.1 \times (-1+0) + 0.9(-1 + V_\pi c(x)) = -10$
$V_\pi b(y) = 0.2 * (-1 + V_\pi b(y)) + 0.8 * (-2 + V_\pi c(x)) = -12.5$

Policy Improvement

$V_\pi b(x) = 0.2 \times (-1 + V_\pi c(x)) + 0.8 * (-1 + V_\pi b(y))$
$V_\pi c(y) = 0.1 \times (-2+0) + 0.9 * (-2 + V_\pi b(y))$
$V_\pi b(x) = -13 \quad < \quad V_\pi c(x) = -10$
$V_\pi c(y) = -13.25 \quad < \quad V_\pi b(y) = 12.5$

∴ $\pi''(x) \to c$
$\pi''(y) \to b$

$\pi' = \pi''$
↓
policy stable

Q.3 (c)

Bellman eqⁿ $V_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma V_\pi(s')\right]$

Initial policy has action b

$\pi(x) \to b$, $\pi(y) \to b$

$V_\pi b(x) = 0.2 * (-1 + \gamma V_\pi b(x)) + 0.8 * (-1 + \gamma V_\pi b(y))$
$V_\pi b(y) = 0.2 * (-2 + \gamma V_\pi b(y)) + 0.8 * (-2 + \gamma V_\pi b(x))$

No Discounting ← The eqⁿ does not hold & the result cannot be calculated

With Discounting

$$V_\pi b(x) = \frac{-1 + 0.8\gamma}{1 - 0.2\gamma} * \frac{-2 + 0.4\gamma}{1 + 0.4\gamma - 0.6\gamma^2}$$

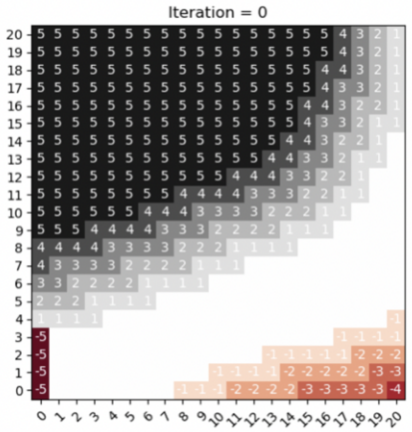
$$V_\pi b(y) = \frac{-2 + 0.4\gamma}{1 - 0.4\gamma - 0.6\gamma^2}$$

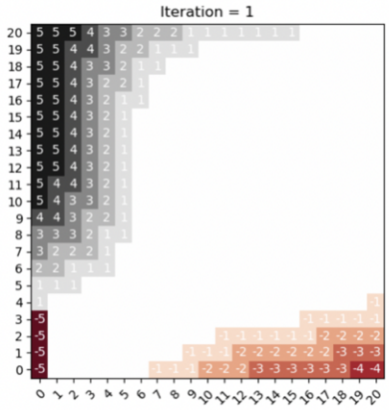
optimal policy depended on the discount factor

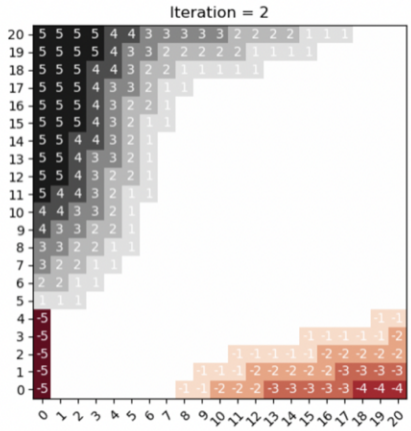
Q. 4

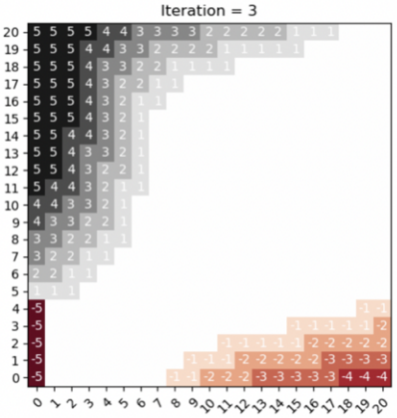
```
==============================
==  Optimal State Value    ==
==============================
[[22.  24.4 22.  19.4 17.5]
 [19.8 22.  19.8 17.8 16. ]
 [17.8 19.8 17.8 16.  14.4]
 [16.  17.8 16.  14.4 13. ]
 [14.4 16.  14.4 13.  11.7]]
==============================
==============================
==     Optimal Policy       ==
==============================
[0, 0] = ['east']
[0, 1] = ['north', 'south', 'west', 'east']
[0, 2] = ['west']
[0, 3] = ['north', 'south', 'west', 'east']
[0, 4] = ['west']
------------------------------
[1, 0] = ['north', 'east']
[1, 1] = ['north']
[1, 2] = ['north', 'west']
[1, 3] = ['west']
[1, 4] = ['west']
------------------------------
[2, 0] = ['north', 'east']
[2, 1] = ['north']
[2, 2] = ['north', 'west']
[2, 3] = ['north', 'west']
[2, 4] = ['north', 'west']
------------------------------
[3, 0] = ['north', 'east']
[3, 1] = ['north']
[3, 2] = ['north', 'west']
[3, 3] = ['north', 'west']
[3, 4] = ['north', 'west']
------------------------------
[4, 0] = ['north', 'east']
[4, 1] = ['north']
[4, 2] = ['north', 'west']
[4, 3] = ['north', 'west']
[4, 4] = ['north', 'west']
------------------------------
```

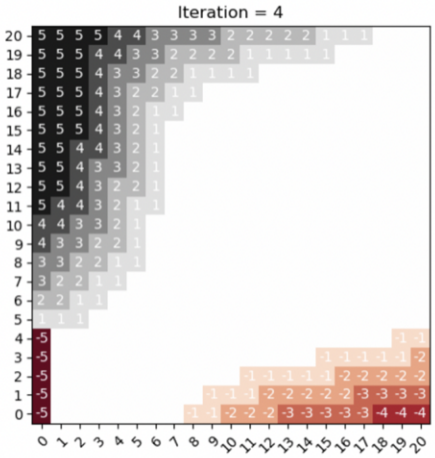
Q.5 a

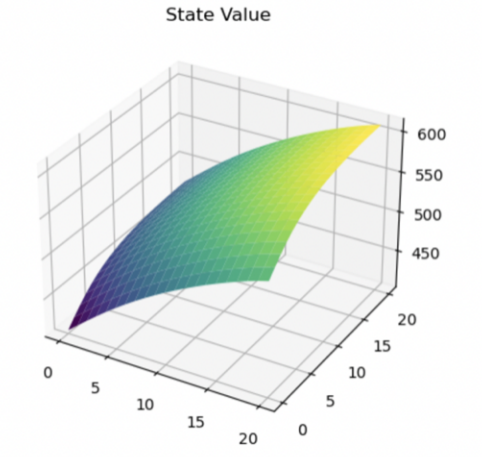

Iteration = 0


Iteration = 1

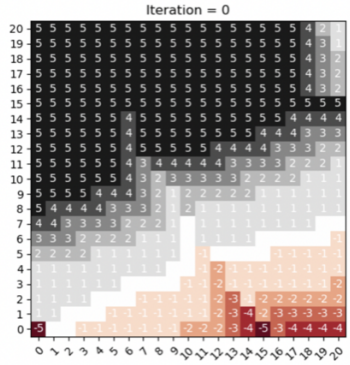

Iteration = 2

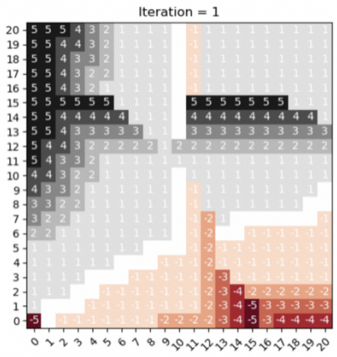

Iteration = 3

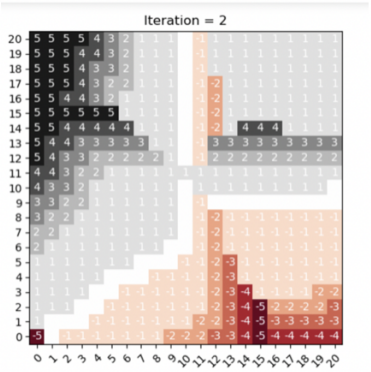

Iteration = 4

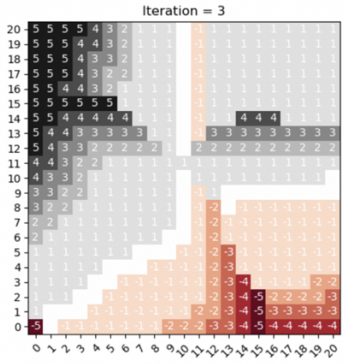

State Value
600
550
500
450
0
5
10
15
20
0
5
10
15
20

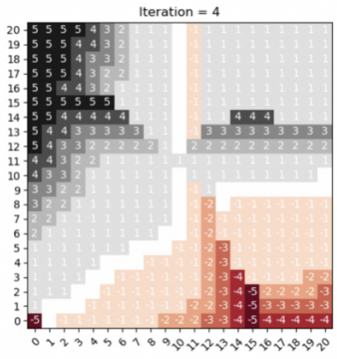
Q 5b

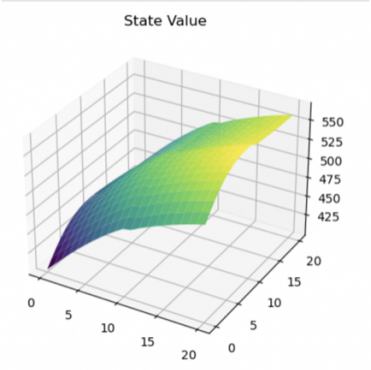

Iteration = 0


Iteration = 1


Iteration = 2


Iteration = 3


Iteration = 4


State Value
550
525
500
475
450
425
0
5
10
15
20
0
5
10
15
20

Reward changes with modifications

- One of Jack's employees at the first location rides a bus home each night and lives near the second location. She is happy to shuttle one car to the second location for free. Each additional car still costs 2, as do all cars moved in the other direction.
- In addition, Jack has limited parking space at each location. If more than 10 cars are kept overnight at a location (after any moving of cars), then a total additional cost of 4 must be incurred to use a second parking lot (independent of how many cars are kept there). (Each location has a separate overflow lot, so if both locations have > 10 cars, the total additional cost is 8.)

As these are the conditions for the reward changes

Reward ← no_of_cars_rented * 10
− 2 * max(no. of cars moved − 1, 0)
− Parking_charges

Resulting ← the location where employees ride the shuttle
the cars moves more in one direction
since cost of moving cars decreased
but because of the parking fees moving
cars becomes costly after certain limit