**Application**

**2-state MDP** (F19F)

Consider a 2-state MDP, the row and column represents from-state and to-state. For example, the probability of transition from to by taking action is . The reward function is and , discount factor

| action |  |  |
| --- | --- | --- |
|  | 0.2 | 0.8 |
|  | 0.6 | 0.4 |

| action |  |  |
| --- | --- | --- |
|  | 0.6 | 0.4 |
|  | 0.2 | 0.8 |

(1) Let denote the optimal state value function of state . Write the Bellman optimality equations.

(2) Prove

Based on the BOE for , we have

(3) Find optimal value

Find max value from (1), where , then solve the function, get

**Policy gradient** (F19F)

Q: Consider , compute the following equations.

A:

**Deep Q-Network** (F19F)

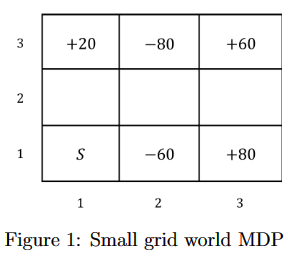
(1) Consider the loss function . Rewrite the loss function with the fixed Q learning target.

(2) Rewrite in terms of

(3) What method is widely used to iteratively find the optimal without computing the expectation?

Stochastic gradient descent (SGD)

**TD(0) / TD()**



Consider a grid-world and the following three episodes from runs of the agent through this grid world:

; ;

(1) Using a learning rate of , and assuming initial Q value is 0, what updates to Q((2,2),E) does on-line TD(0) method make after the above three episodes?

There is no update to this point after three episodes, Q((2,2), E)=0

(2) What updates to Q((2,2),E) does on-line backward-view TD() method make after the above three episodes?

There is no update to Q((2,2),E) after the first episode. The accumulating eligibility trace of the state-action pair ((2,2),E) is 0,0,1, during the second and the third episode. Therefore, the update to Q((2,2),E) after the second episode is:

And the update to Q((2,2),E) after the third episode is:

(3) Consider a feature based representation of the Q-value function: , where is the row number of the state s, is the column number of the state s, and x\_3(s,a) = 1,2,3,4 if a is N,E,S,W, respectively. For example, if s=(1,1) and a=N, then . Given that all are initially 0, what are their values using on-line backward-view TD() after the first episode? Use and learning rate .

By the definition of features, we have . The accumulating eligibility trace is . So the sequence of eligibility traces in the first episode is , and . Therefore, the update to weights after the first episode is:

**on-line vs off-line update** (HW4)

(1) Difference between on and off line:

In on-line updating, the update are done during the episode, as soon as the increment is computed. In off-line updating, on the other hand, the increments are accumulated “on the side” and are not used to change value estimates until the end of the episode.

(2) Consider an episode: A,+1,B,+2,A,+1,T from an undiscounted MDP, learning rate . What is total update to V(A) on-line and off-line every-visit constant- MC method makes after the episode finishes?

On-line: ,

Off-line: , , Total update is

(3) Total update of online and off-line TD(0)?

On-line: ,

Off-line: , , total update is

(4) On-line/Off-line forward/backward view of TD()

,

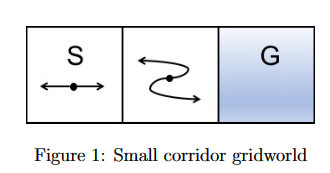
On-line forward: ,

Off-line forward: , , total update is

On-line backward: ,

Off-line backward: , , total update is 0.325

**Linear function approximation** (HW4)



(1) Consider a discounted experiment with actions right, right, right, left. Reward is -1. For two parameters , if , calculate the -return corresponding to this episode for :

(2) Using forward-view TD(), right the sequence corresponding to the right action, :

(3) Define TD() trace , for the right action, :

Where the linear value function approximation in trace is

The sequence of eligibility traces corresponding to right action should be:, , ,

(4) Backward-view TD() sequence of updates to weight , what is the total update to weight ?

(5) When using off-line updates and linear function approximation, are forward is equivalent to backward?

Yes. Forward-view and backward-view TD(λ) is equivalent to each other.

**DQN** (HW5)

Why using “experience delay” and “fixed Q-targets” in DQN? Explain why using these two can help stabilise DQN algorithm when the correlations present in the sequence of observations?

Because the experience replay can step out of the correlation that comes with data, which is called independent and identically distributed random variables environment. The fixed Q-targets then, can also step out the correlation between action values and the target, and stabilise the algorithm of DQN.

In sequence of observations like Atari games, the DQN is prefer to forgot the previous situation, so the experience replay can be useful to avoid this case and maintain the previous situation. Fixed Q-targets can replace the attributes which are not good and replace by the new network, which can make the training easier.

**DPG / DDPG** (HW5)

Why discretise the continuous action space may not help in practice in DQN? How DDPG solve this problem?

Because the discretion of the continuous action space needs to maximise the Q value for each step, which is performance-costing and give processor too much pressure. Also, the maximising the Q value for each step needs a lot of storage space, to make the computing more complicated, which makes the continuous space may not be able to explore all states.

DDPG is the fusion of the DPG and DQN, which presenting as a off-policy algorithm, achieved by deep network, so it can handle with this problem.