CS6700 - REINFORCEMENT LEARNING
PROGRAMMING ASSIGNMENT 3

Team member 1: Gokul Venkatesan (Roll number: ME18B047)
Team member 2: Vishal (Roll number: CH18B013)

ENVIRONMENT DESCRIPTION:

The environment for this task is the taxi domain. It is a 5x5 matrix, where each cell is a position your taxi can stay at. There is a single passenger who can be either picked up or dropped off, or is being transported. There are four designated locations in the grid world indicated by R(ed), G(reen), Y(ellow), and B(lue). When the episode starts, the taxi starts off at a random square and the passenger is at a random location. The taxi drives to the passenger's location, picks up the passenger, drives to the passenger's destination (another one of the four specified locations), and then drops off the passenger. Once the passenger is dropped off, the episode ends.

There are 500 discrete states since there are 25 taxi positions, 5 possible locations of the passenger (including the case when the passenger is in the taxi), and 4 destination locations. Note that there are 400 states that can actually be reached during an episode. The missing states correspond to situations in which the passenger is at the same location as their destination, as this typically signals the end of an episode. Four additional states can be observed right after a successful episodes, when both the passenger and the taxi are at the destination. This gives a total of 404 reachable discrete states.

Passenger locations: 0: R(ed); 1: G(reen); 2: Y(ellow); 3: B(lue); 4: in taxi
Destinations: 0: R(ed); 1: G(reen); 2: Y(ellow); 3: B(lue)
Rewards:

- -1 per step unless other reward is triggered.
- +20 delivering passenger.
- -10 executing ”pickup” and ”drop-off” actions illegally.

For the given environment, we have used two sets of Markov options that are mutually exclusive. Before introducing the options, we list the primitive actions that are available in all the states of the environment.

0: Go South
1: Go North
2: Go East
3: Go West
4: Pick the passenger up
5: Drop the passenger off

SET OF OPTIONS - 1:

a) Go RED:

1	**3**	**0**	**3**	**3**
1	**3**	**0**	**3**	**3**
1	**3**	**3**	**3**	**3**
1	**1**	**3**	**1**	**3**
1	**1**	**3**	**1**	**3**

Initiation states: All states
Termination states: RED state

b) Go YELLOW:

0	**3**	**0**	**0**	**0**
0	**3**	**0**	**3**	**3**
0	**3**	**3**	**3**	**3**
0	**1**	**3**	**1**	**3**
0	**1**	**3**	**1**	**3**

Initiation states: All states
Termination states: YELLOW state

c) Go GREEN:

2	**0**	**2**	**2**	**2**
2	**0**	**2**	**2**	**1**
2	**2**	**2**	**2**	**1**
1	**2**	**1**	**2**	**1**
1	**2**	**1**	**2**	**1**

Initiation states: All states
Termination states: GREEN state

d) Go BLUE:

2	**0**	**2**	**0**	**3**
2	**0**	**2**	**0**	**3**
2	**2**	**2**	**0**	**3**
1	**2**	**1**	**0**	**3**
1	**2**	**1**	**0**	**3**

Initiation states: All states
Termination states: BLUE state

SET OF OPTIONS - 2:
a) Go RED:

1	**3**	**0**	**0**	**0**
1	**3**	**0**	**0**	**0**
1	**3**	**3**	**3**	**3**
1	**1**	**1**	**1**	**1**
1	**1**	**1**	**1**	**1**

Initiation states: All states
Termination states: RED state

b) Go YELLOW:

0	**0**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**
0	**3**	**3**	**3**	**3**
0	**1**	**1**	**1**	**1**
0	**1**	**1**	**1**	**1**

Initiation states: All states
Termination states: YELLOW state

c) Go GREEN:

0	**0**	**2**	**2**	**2**
0	**0**	**2**	**2**	**1**
2	**2**	**2**	**2**	**1**
1	**1**	**1**	**1**	**1**
1	**1**	**1**	**1**	**1**

Initiation states: All states
Termination states: GREEN state

d) Go BLUE:

0	**0**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**
2	**2**	**2**	**0**	**3**
1	**1**	**1**	**0**	**3**
1	**1**	**1**	**0**	**3**

Initiation states: All states
Termination states: BLUE state

Along with a set of options (option set 1 or 2), we also included the primitive actions **0, 1, 2, 3, 4 (Pick the passenger up)** and **5 (Drop the passenger off)** during the learning phase.

SMDP Q-LEARNING:

Two different sets of options were tried for SMDP.

Parameters:

Gamma: 0.90

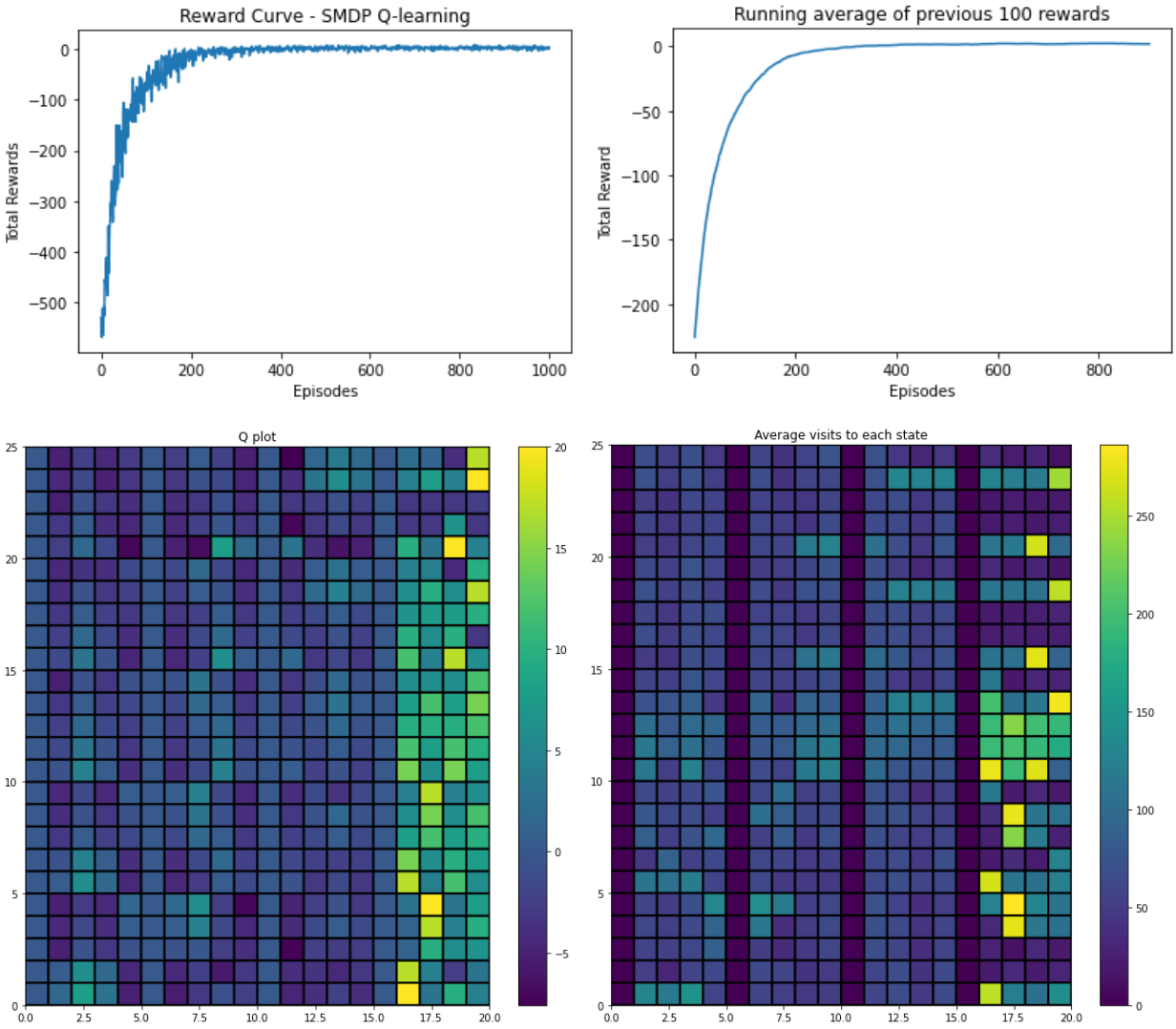
Alpha: 0.90

Epsilon: 0.10

Number of runs: 10

Number of episodes: 2000

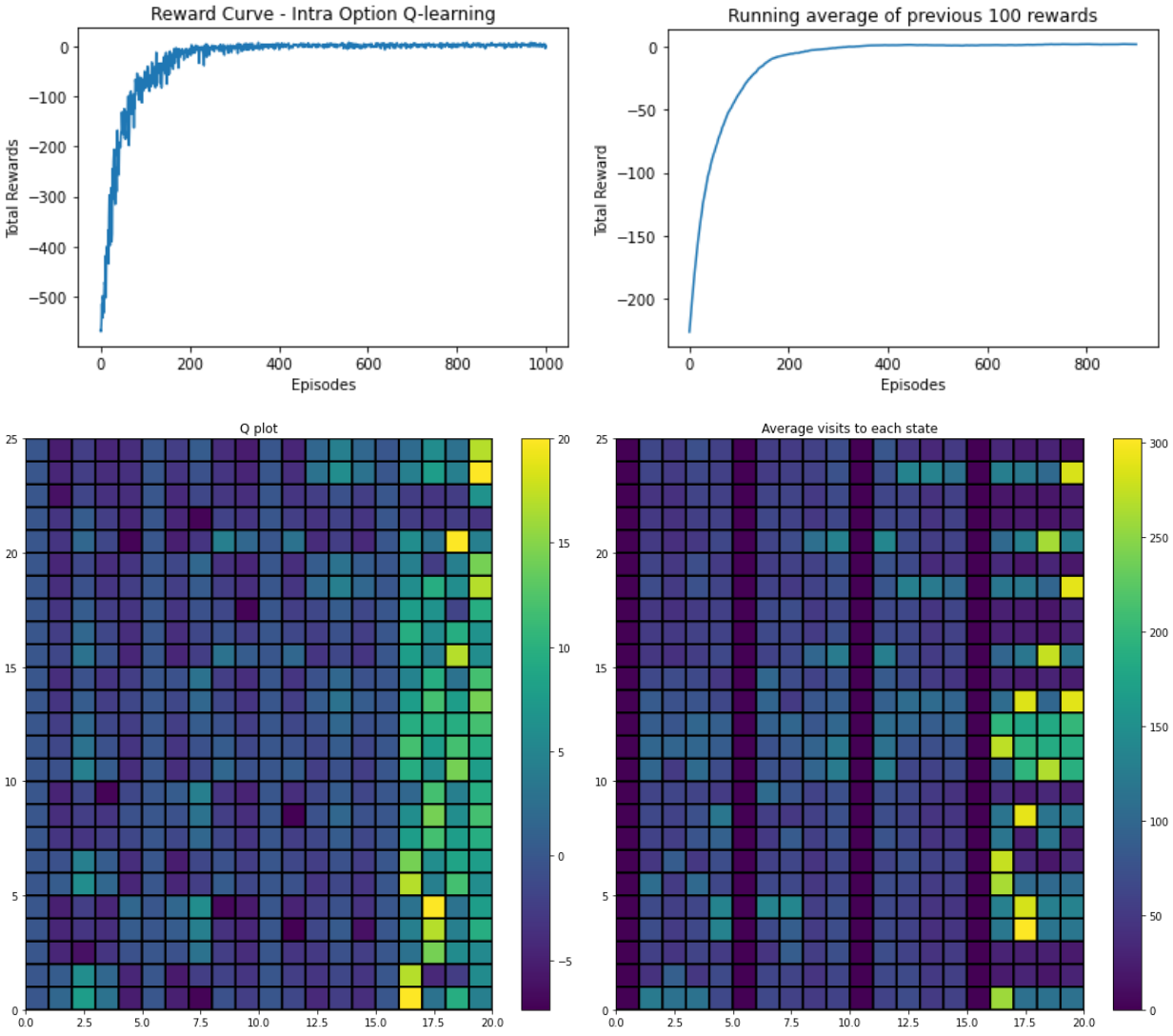
Reward curves and learned Q-values for first set of options:



From the Q-values, we can see that the states RED, GREEN, YELLOW, and BLUE have the maximum state-action value of 20 corresponding to the action 5 (Drop the passenger off). At the beginning of the episode, the taxi moves to the location of the passenger (using the appropriate option). Once it reaches the passenger location, the taxi moves to the destination (once again using the appropriate option). Once the destination is reached, the passenger is dropped, gaining a reward of +20. The information regarding the destination and the passenger location is obtained from the states. Because of the options, the taxi is able to learn faster. Otherwise, the

taxi should also learn to move to each location (RED, GREEN, YELLOW, GREEN) by itself, only using primitive actions.

Reward curves and learned Q-values for second set of options:
Parameters:
Gamma: 0.90
Alpha: 0.90
Epsilon: 0.10
Number of runs: 10
Number of episodes: 2000



We can see that both the set of options perform similarly. That is because both the options have an optimal policy to perform the composite action i.e going to a particular destination. If one set of options had a sub-optimal policy, we could see a difference in the reward curves.

INTRA-OPTION Q-LEARNING:

Along with a set of options (option set 1 or 2), we also included the primitive actions **4 (Pick the passenger up),** and **5 (Drop the passenger off)** during the learning phase.

SET OF OPTIONS - 1:
Parameters:
Gamma: 0.90
Alpha: 0.90
Epsilon: 0.10
Number of runs: 10
Number of episodes: 2000

Reward Curve:

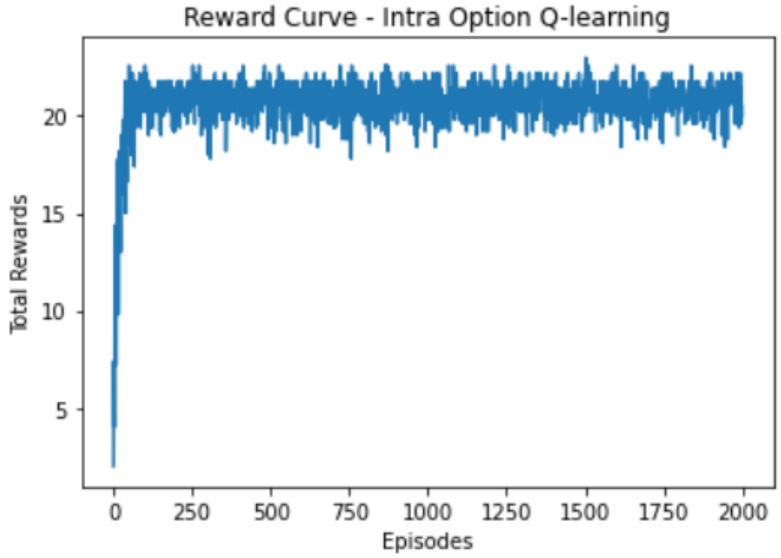


Figure 1: Average reward curve over 10 runs - option set 1

Figure 2: Running average of previous 100 rewards - option set 1

Q values:

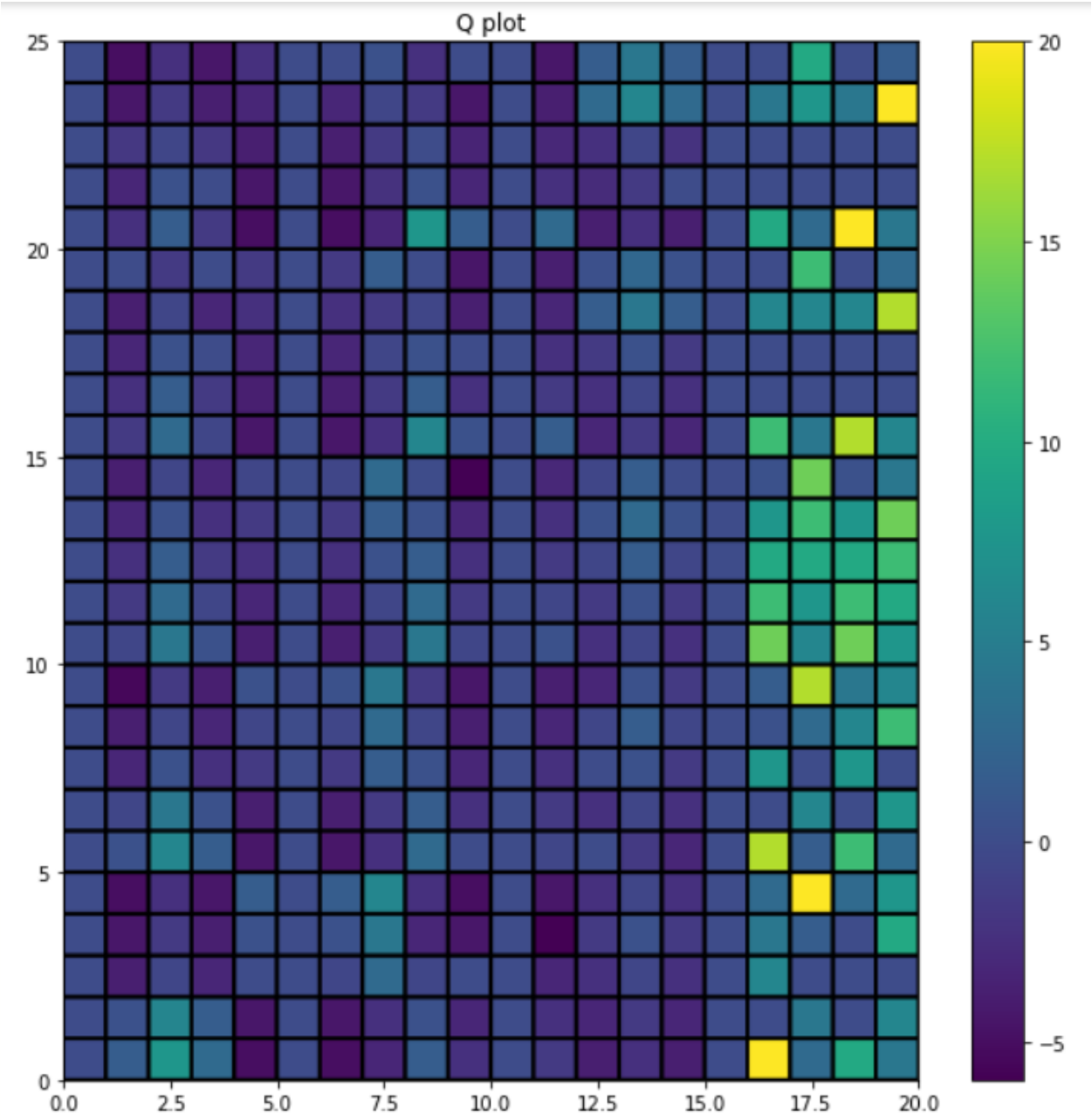


Figure 3: Maximum Q value of each state - option set 1

It was observed that a learning rate (alpha) greater than 1.2 led to poor performance and the reward curve started to diverge. From the Q-values (figure 3), we can see that the states RED, GREEN, YELLOW, and BLUE have the maximum state-action value of 20 corresponding to the action 5 (Drop the passenger off). At the beginning of the episode, the taxi moves to the location of the passenger (using the appropriate option). Once it reaches the passenger location, the taxi moves to the destination (once again using the appropriate option). Once the destination is reached, the passenger is dropped, gaining a reward of +20. The information regarding the destination and the passenger location is obtained from the states. Because of the options, the taxi is able to learn faster. Otherwise, the taxi should also learn to move to each location (RED, GREEN, YELLOW, GREEN) by itself, only using primitive actions.

SET OF OPTIONS - 2:

Parameters:

Gamma: 0.90

Alpha: 0.90

Epsilon: 0.10

Number of runs: 10

Number of episodes: 2000

Reward Curve:

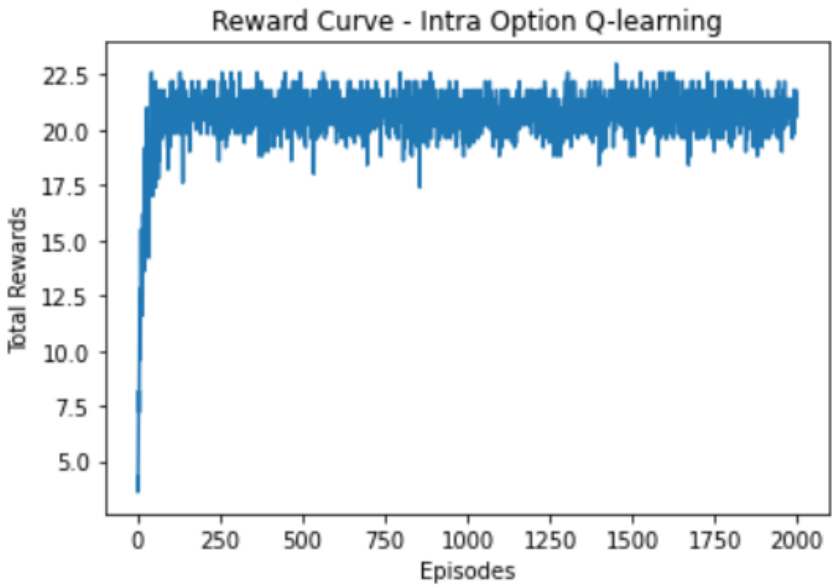


Figure 4: Average reward curve over 10 runs - option set 2

Figure 5: Running average of previous 100 rewards - option set 2

Q values:

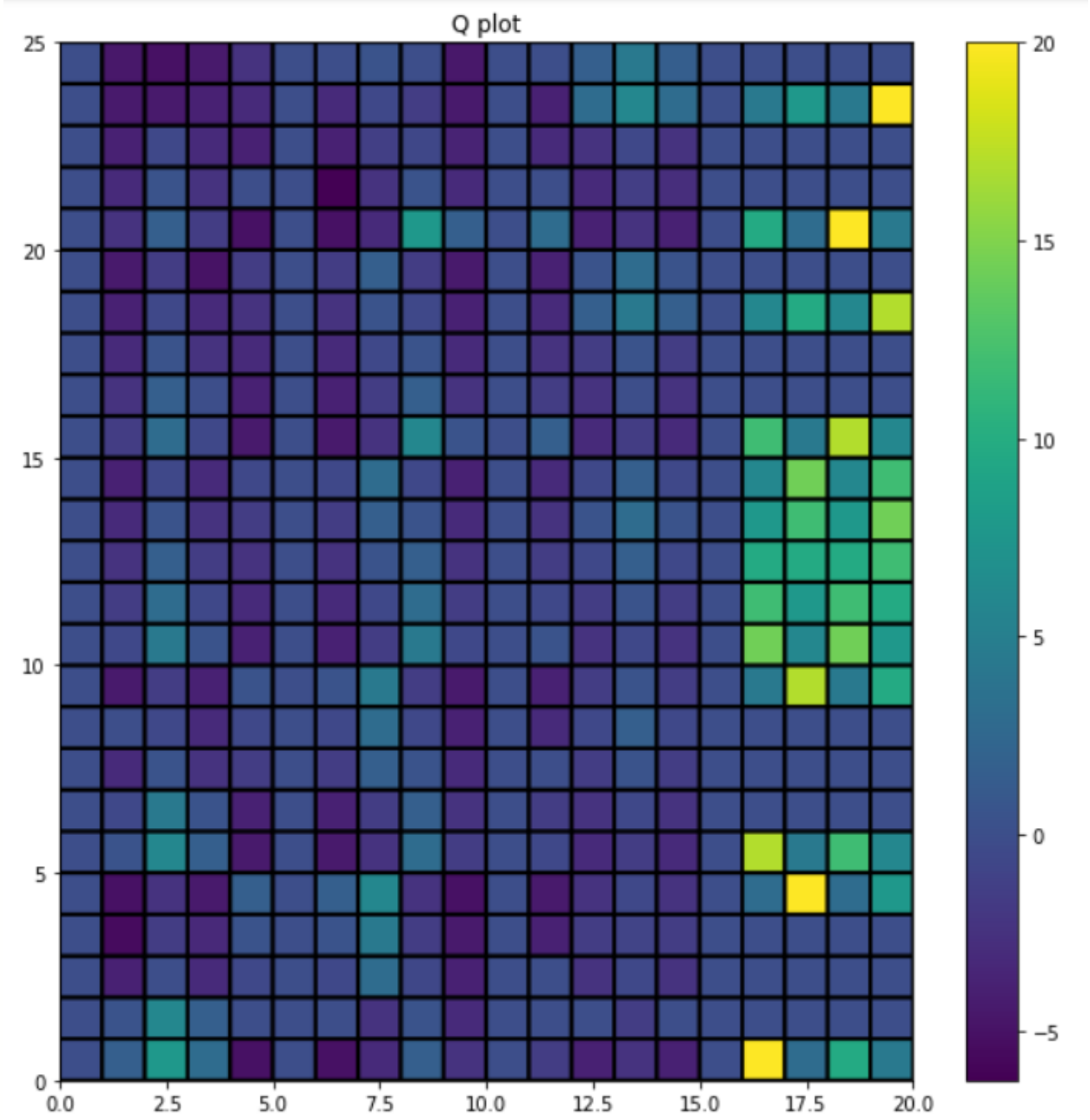


Figure 6: Maximum Q value of each state - option set 2

We can see that both sets of options perform similarly. That is because both the options have an optimal policy to perform the composite action i.e going to a particular destination. If one set of options had a sub-optimal policy, we could see a difference in the reward curves.

In comparison to SMDP Q learning, we can see that Intra-option Q-learning performs better and converges faster. In SMDP Q-learning, the update is done only after an option is completed. On the contrary, in Intra-option Q-learning, updates are done in each step. Hence, Intra-option Q-learning updates the Q-values more frequently than SMDP Q-learning.