# HW2

Brody Kendall

October 2022

## Problem 1

### a

The optimal policy is as follows:

| S = | (1,1) | (1,2) | (1,3) | (2,1) | (2,2) | (2,3) |
|---|---|---|---|---|---|---|
| $\pi^*(S) =$ | right | right | goal | up | up | goal |

### b

| S = | (1,1) | (1,2) | (1,3) | (2,1) | (2,2) | (2,3) |
|---|---|---|---|---|---|---|
| $V_0(S) =$ | 0 | 0 | 6 | 0 | 0 | -7 |
| $V_1(S) =$ | 0 | 3.99 | 6 | 0 | -0.67 | -7 |
| $V_2(S) =$ | 2.65 | 4.31 | 6 | -0.06 | 1.92 | -7 |

Calculations:
$V_1(1,1) =$ (any direction) $.95(.7(0) + .3(0)) = 0$
$V_1(1,2) =$ (right) $.95(.7(6) + .3(0)) = 3.99$
$V_1(2,1) =$ (any direction) $.95(.7(0) + .3(0)) = 0$
$V_1(2,2) =$ (up, left, or down) $.95(.7(0) + .1(-7) + .2(0)) = -.67$
$V_2(1,1) =$ (right) $.95(.7(3.99) + .3(0)) = 2.65$
$V_2(1,2) =$ (right) $.95(.7(6) + .1(-0.67) + .1(3.99) + .1(0)) = 4.31$
$V_2(2,1) =$ (up, left, or down) $.95(.7(0) + .1(-.67) + .2(0)) = -0.06$
$V_2(2,2) =$ (up) $.95(.7(3.99) + .1(-0.67) + .1(-7) + .1(0)) = 1.92$

## Problem 2

### a

Let $\alpha = 0.5, \gamma = 1$, and initialize all q-values to 0.
Update 1: $Q(A, \rightarrow) = .5(0) + .5(4 + 0) = 2$
Update 2: $Q(C, \leftarrow) = .5(0) + .5(3 + 0) = 1.5$
Update 3: $Q(B, \rightarrow) = .5(0) + .5(-1 + 1.5) = 0.25$
Update 4: $Q(A, \rightarrow) = .5(2) + .5(8 + 0.25) = 5.125$
Update 5: $Q(C, \rightarrow) = .5(0) + .5(9 + 0.25) = 4.625$

### b

After all the updates, $Q(A, \rightarrow) = 5.125$

### c

After all the updates, $Q(B, \rightarrow) = 0.25$

### d

After all the updates, $Q(C, \rightarrow) = 4.625$

## Problem 3

### 1

There are 16 states (each square in the grid) and 4 actions (up, down, left, right).

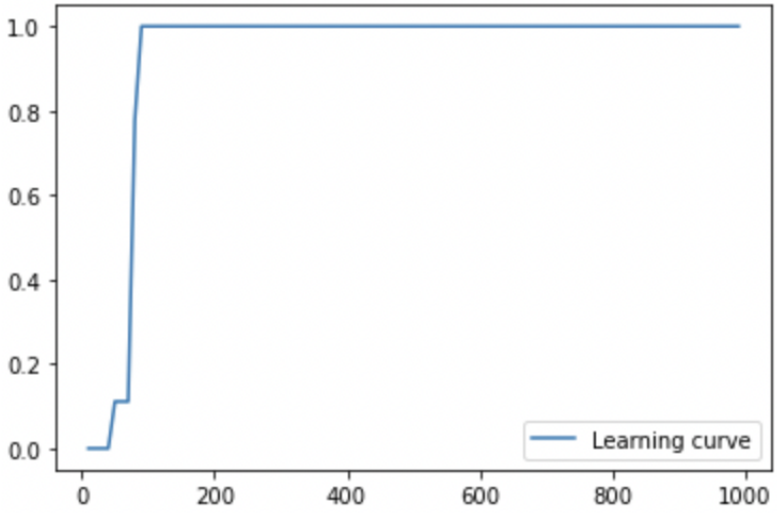
**2**

```
print(q_table)
```

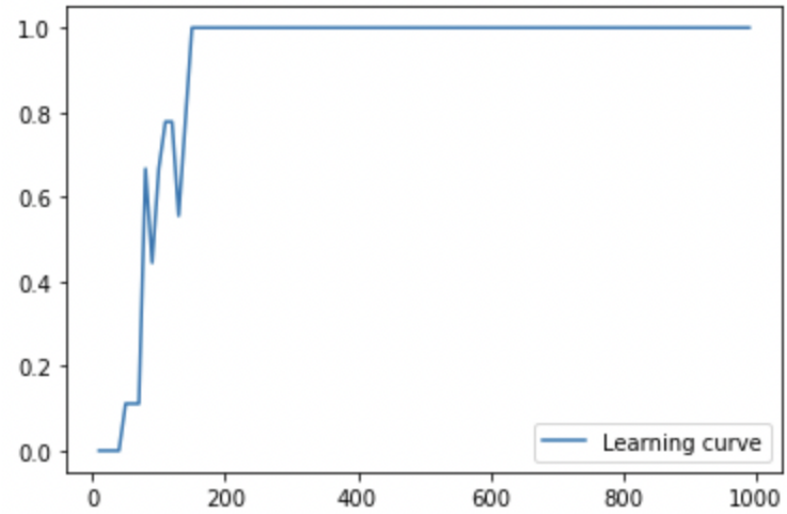
```
[[0.      0.      0.59049 0.     ]
 [0.      0.      0.6561  0.     ]
 [0.      0.729   0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.81    0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.9     0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      0.      0.     ]
 [0.      0.      1.      0.     ]
 [0.      0.      0.      0.     ]]
```

This Q-table does make sense. Essentially, the algorithm learned a successful path and then never strayed from that path.

**3**




**4**



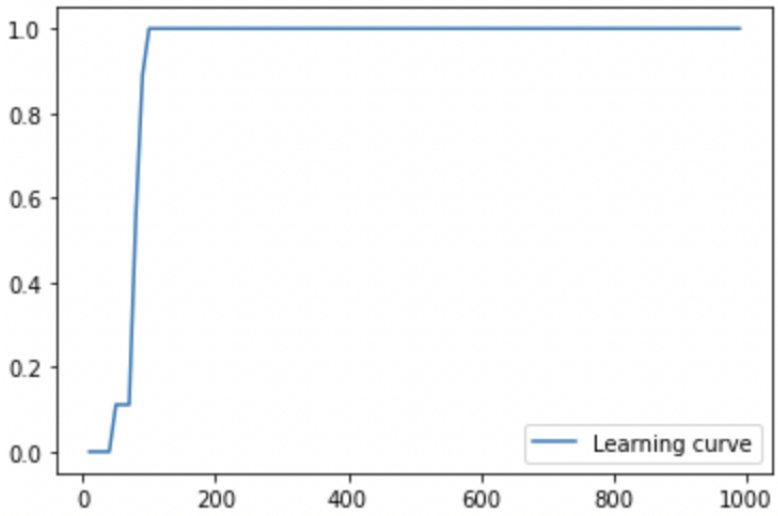

Changing alpha to 0.7 and gamma to 0.2 increases the rate of convergence from around 90 to around 150, and it makes the learning curve a little more unstable pre-convergence. The results of a few other pairs of values are as follows: alpha = 0.2, gamma = 0.7, convergence rate = 90; alpha = 0.9, gamma = 0.1, convergence rate = 510; alpha = 0.9, gamma = 0.9, convergence rate = 90; alpha = 0.1, gamma = 0.1, convergence rate > 1000; alpha = 0.1, gamma = 0.9, convergence rate = 130. For this example, it seems that a high gamma and

a relatively high alpha are good, with a low gamma leading to slow convergence.

**5**

```
[[0.3051837  0.         0.59049    0.        ]
 [0.22473688 0.         0.6561     0.29521599]
 [0.14358595 0.729      0.         0.33623048]
 [0.         0.         0.         0.        ]
 [0.         0.         0.         0.        ]
 [0.         0.         0.         0.        ]
 [0.         0.81       0.         0.48634569]
 [0.         0.         0.         0.        ]
 [0.         0.         0.         0.        ]
 [0.         0.         0.         0.        ]
 [0.         0.9        0.         0.36449468]
 [0.         0.         0.         0.        ]
 [0.         0.         0.         0.        ]
 [0.         0.         0.         0.        ]
 [0.         0.         1.         0.68396896]
 [0.         0.         0.         0.        ]]
```

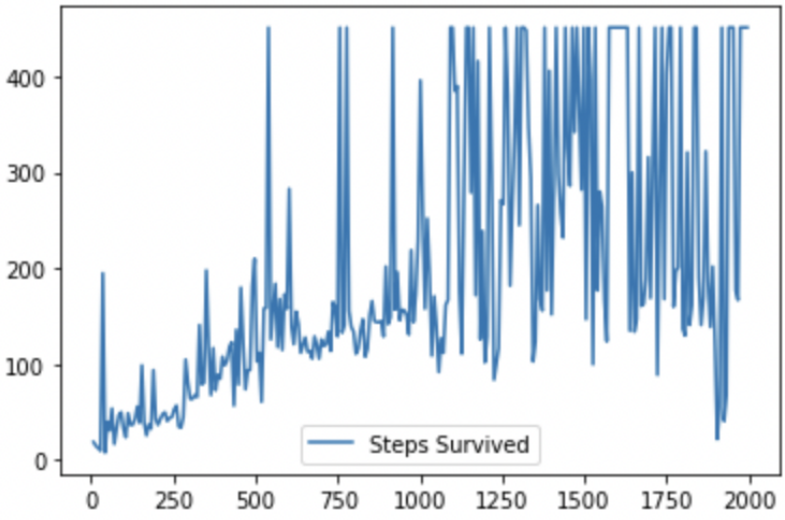
After implementing epsilon greedy exploration, the Q-table changes. We see that some of the values that were 0 in the Q-table from part 3 are now no longer 0. Notably, all the values that were nonzero in the part 3 Q-table have still converged to the same value with epsilon greedy exploration. In addition, only one of the values that changed from 0 to nonzero between Q-tables is larger than any of the values that remained nonzero between both tables, and the highest value action in each state remained unchanged across both tables. This suggests that with epsilon-greedy exploration, the learner did find other successful sequences of actions, but that none of them could considered better than the path discovered by the original. In fact, all of the new non-zero actions in the table either keep the agent in the same place (by running into a wall) or reverse along the same path. Therefore, while the learner was technically able to find other successful sequences of actions, the actual path that the agent will likely take will be unchanged.



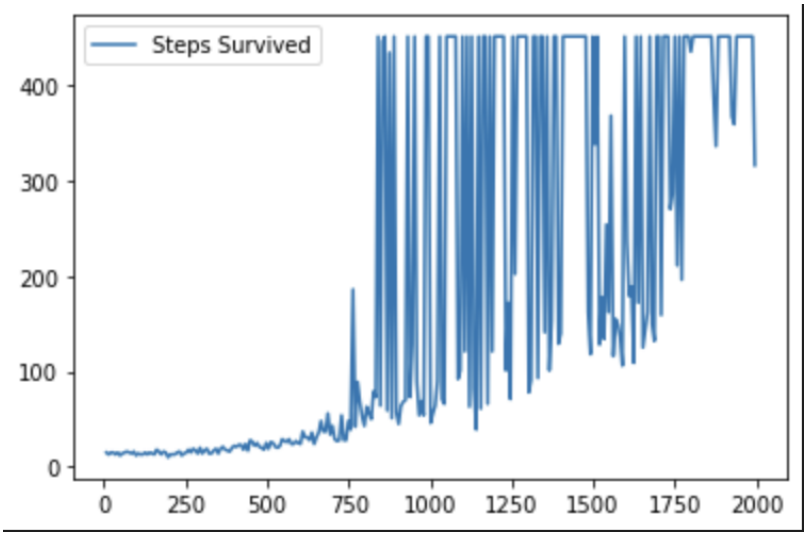

The convergence rate only slightly increased from 90 (no epsilon greedy exploration) to 100 (with epsilon greedy exploration).

# Problem 4

## 1



**2**




From the attempts that I tried, changing gamma, the decay factor, or the number of hidden units in either direction seemed to hurt performance.

Performance improved when I increased the number of times you train per episode from 20 to 50, but performance got worse when I increased this value further.

Increasing the learning rate to 0.01 hurt performance, but decreasing to 0.0005 improved performance, and decreasing again to 0.0001 improved performance again. However, when I decreased the learning rate to 0.00001, the performance got significantly worse.

The plot shows the learning curve with the following parameters: gamma = 0.95 (unchanged), number of times you train per episode = 50 (increased), epsilon threshold decay rate = 0.995 (unchanged), number of hidden units = 128 (unchanged), optimizer learning rate = 0.0001 (decreased). It resulted in an average number of survival steps of 340.83 for the last 100 episodes, an increase from an average of 292.70 for the last 100 episodes from the original parameter set.

Of course, this analysis should all be taken with several grains of salt - this is not a rigorous analysis, testing each change of parameters a total of 1 time.