CS 747
Assignment 4 - Report

Goutham Ramakrishnan, 140020039

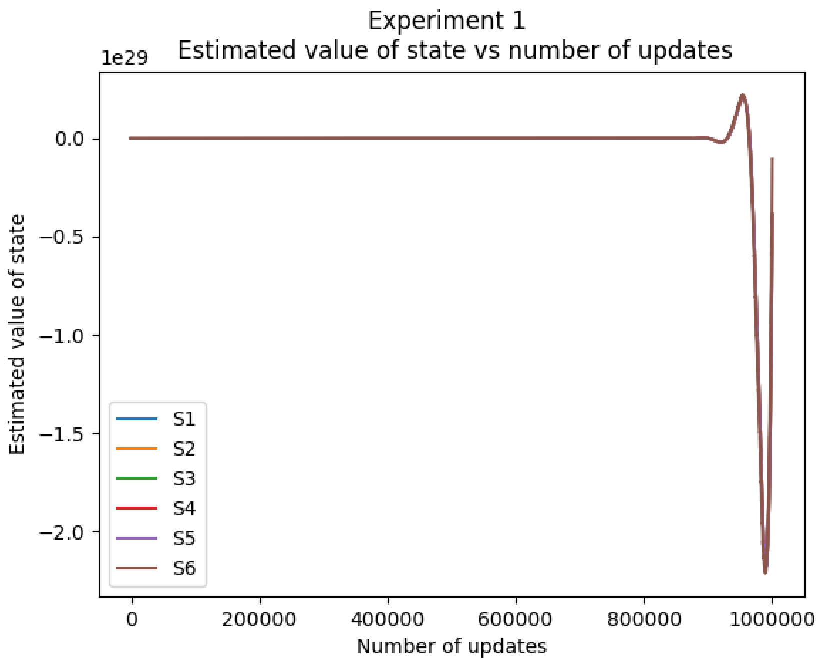
Implementation of algorithms has been done in python, using the equations described in Sutton & Barto.

1.
In experiment 1, the weight vector updates are done in a round robin fashion. This means that states S1-S5 get a higher percentage of updates than are optimal. Optimally, S1-S5 should get ~1% of the updates whereas S6 should get the remaining bulk of updates. Here, each state gets ~16% of the updates.

Due to this mismatch, the algorithm does not always converge to the right value function. Clearly, as seen from the graph below, the value function is diverging(in fact, it is oscillating exponentially. Note that the scale of the axis in the graph is of the order of 10^{29}).

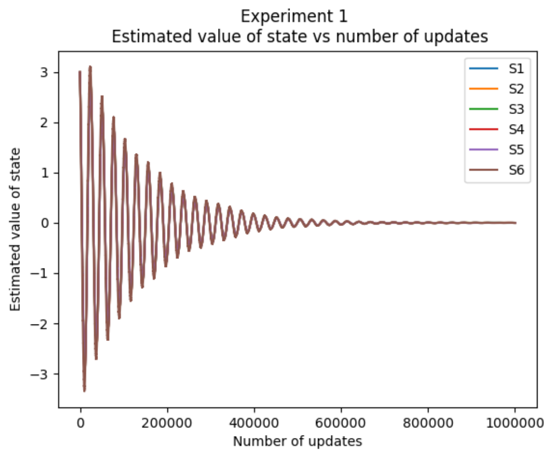
As proved in class for Tsitsilis and Van Roy's counterexample, when each state is given equal weightage, the algorithm does not always converge. It depends upon the values of epsilon(here the analogous value of epsilon is 0.99) and gamma. By a similar computation, one can prove that we run into similar problems in Baird's counterexample. The algorithm does not converge for large values of gamma.

The graph plotted below is for gamma=0.99. The value function is diverging. The exponential oscillation is not clear from the graph as the smaller peaks and troughs are negligible as compared to the scale. Only the last few oscillations are clear.

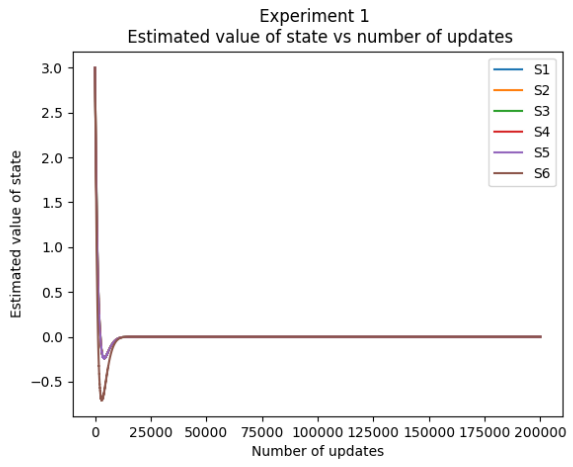


Another thing to note is that the value functions of all the states change in a similar fashion (the graphs appear to be one line). This is to expected as well, due to the symmetry present in the MDP and in the initialization.

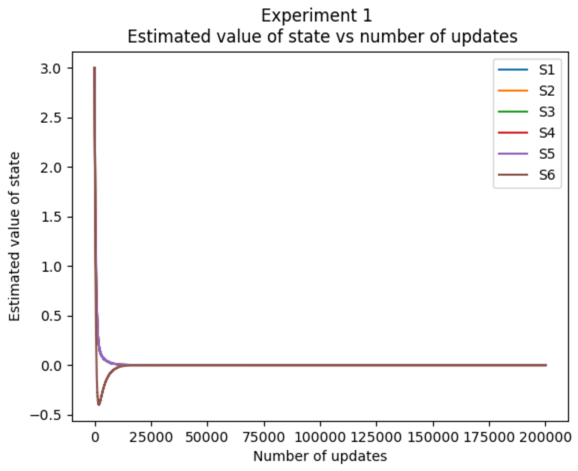
For smaller values of gamma, it is observed that the algorithm converges.
For example, for gamma=0.93, the following graph is observed:



This is a case of damped oscillations. The algorithm is converging to the correct value function, but displays oscillatory behavior in the process.



The oscillations become more and more damped as the value of gamma is decreased further. The graph for gamma=0.5 is to the left.

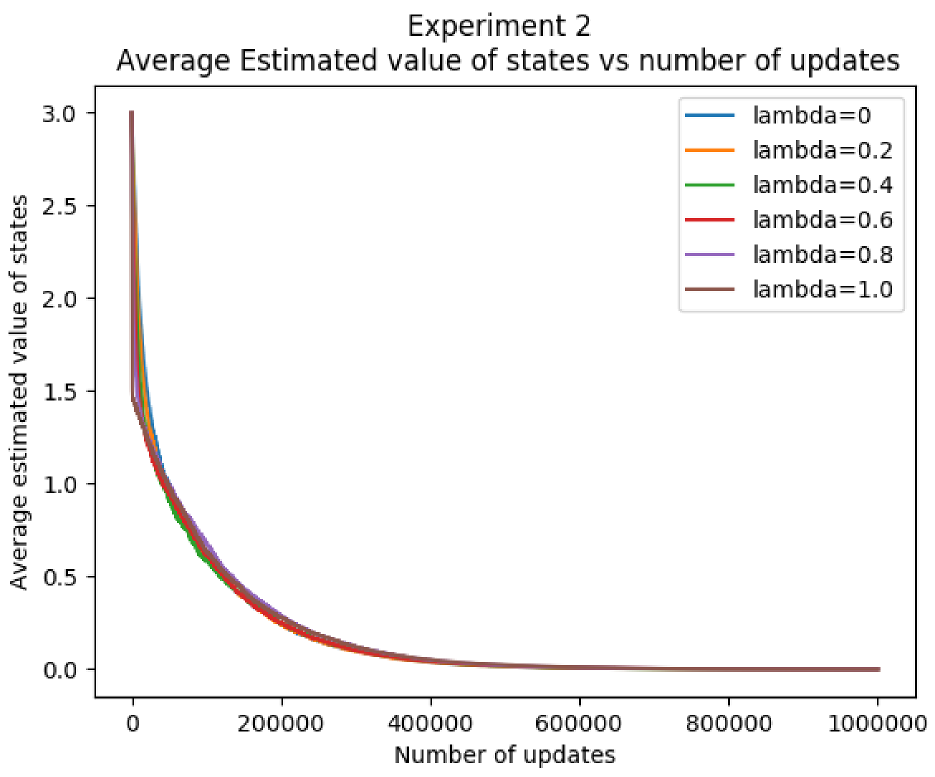


The oscillatory behaviour is present even for gamma=0.0 as seen to the left! They die down extremely fast. In fact, only the value of S6 becomes negative, the rest of the states converge in an exponential manner.

An interesting observation can be made here that the rate of converence appears to be inversely proportional to the value of gamma.

Also, one can observe that the value function of S6 displays the greater oscillatory behavior. This is also reasonable as the coefficient of w7 is 2, whereas it is 1 for all other states. W7 gets updated at every update step, whereas the rest of the weights dont.

2.
The graph below plots how the average value of states changes with the number of updates, for 6 different values of lambda.



One can make a number of interesting observations:

- ➔ The average values of the value functions converge for all values of lambda.
- ➔ The graphs are almost overlapping, which shows that the algorithm is robust to the value of lambda. Some values of lambda seem to converge faster initially, but towards the end, they all converge after almost the same number of update steps.
- ➔ The problems seen with experiment 1 do not affect experiment 2. As we follow the dynamics of the MDP until the episode ends, the updates are automatically weighted according to the stationary probabilities of the states. Thus even for gamma=0.99, the algorithm converges, for all values of lambda.
- ➔ The stochastic gradient descent works well in experiment 2, as we have weighted the mean square error correctly, indirectly by following the dynamics of the MDP.

3.
The final values that the weight vector coverges to depends upon the initialization. But, the algorithm converges to the correct value function irrespective of the initilialization and even lambda. The value of lambda only seems to affect the rate of convergence.

The final weight vector always converges to k*[1,1,1,1,1,4,-2], where k is any real number. This is the solution to the following homogeneouos matrix equation defined by the structure of the MDP.

$$\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 2 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 2 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

For example,
1. When w is initialized to [1,1,1,1,1,1,1], it converges to [0.28,0.28,0.28,0.28,0.28,1.12,-0.56]
2. When w is initialized to [10,21,35,-4,5,6,7], it converges to [3.08,3.08,3.08,3.08,3.08,12.32,-6.16]
3. When w is initialized to [-1,-5,8,7,4,3,9], it converges to [0.28,0.28,0.28,0.28,0.28,1.12,-0.56]

Note that despite the drastically different intializations in 1 and 3, it converged to the same weight vector.
Also note that the resulting value function in experiment 2 is always as expected, all zeros.

<u>Attached:</u>
Shell script to start client: startmdp.sh
Primary python script: assignment_4.py (set plot=True for displaying graphs)

<u>References:</u>

- Python Documentation
- http://incompleteideas.net/sutton/book/ebook/node87.html
- Sutton and Barto
- StackOverFlow