CS7641 Machine Learning:
Assignment 4 Markov Decision Processes

Judy Jungeun Cha
jcha64@gatech.edu

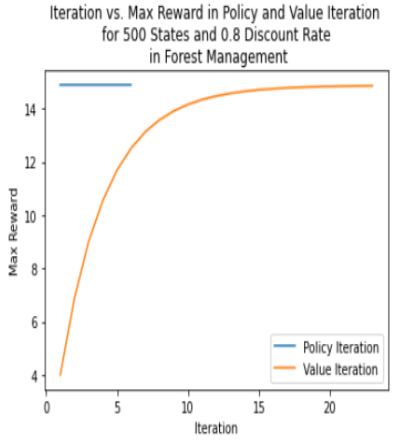
1 INTRODUCTION

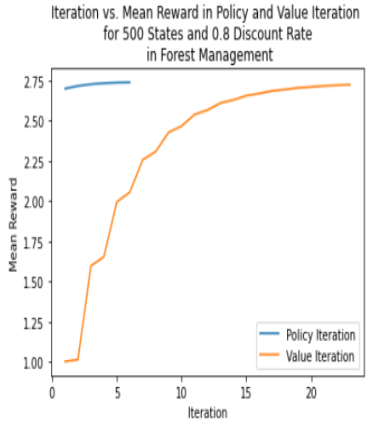
The purpose of this report is to explore Markov Decision Processes. I have chosen two problems of Frozen Lake (FL) problem and the Forest Management (FM) problem. FL has a small state (4x4) of grid world and FM has a large state (500) of non-grid world.

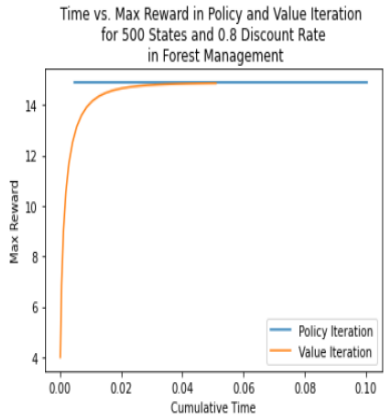
2 FOREST MANAGEMENT (FM)

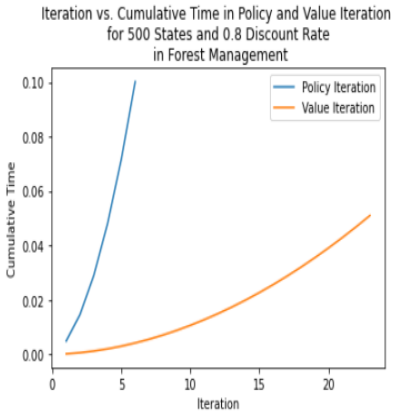
My FM problem has a large state (500) of non-grid world problem. The problem deals with the optimal policy to cut down a forest. There are two actions are involved. If you simply 'WAIT' (Action = 0) for the cut of forest and get a reward of 4 from enjoying wild forest, but when 'CUT' (Action = 1) is performed, the reward is a half which is 2 which makes a profit of selling the wood. The state (S) is the number of years old the forest can be since last cut or burn. And there is a chance that a wildfire occurs (p). It is interesting because it can be closely related to our real life. We often face decision making situation like FM problem to take an action now or wait for later. This exercise could help to see what's the series of actions would be for the optimal results. The below chart shows when it converges, how long take time to converge, mean/max value of rewards and error of policy iteration and value iteration under states changes.

States		200	500	1000	5000
Convergence (# of iterations)	policy	6	6	6	6
	value	23	23	23	23
Time (sec)	policy	0.0362	0.1002	0.6109	19.2479
	value	0.0483	0.0510	0.1086	2.0115
Mean Value	policy	2.8288	2.7362	2.7053	2.6806
	value	2.8148	2.7223	2.6914	2.6668
Max Value	policy	14.8837	14.8837	14.8837	14.8837
	value	14.8632	14.8632	14.8632	14.8632
Error	policy	0.0000	0.0000	0.0000	0.0000
	value	0.0029	0.0029	0.0029	0.0029









Noted that for algorithm calculation (Policy Iteration, Value Iteration, Q-learning), Markov Decision Process (MDP) toolbox is used. The gamma (discount rate) is set as 0.8 for comparison purpose.

Policy Iteration (PI)

PI has two interacting processes of policy evaluation and policy improvement. PI makes the value function consistent with the current policy(cooperating) and has the policy greedy to the current value function to yield a better policy(competing). It repeats until convergence has occurred to the optimal value function and an optimal policy, but it can be defined to stop when the delta in policy improvement is smaller than the threshold. In MDP toolbox set the threshold by default. Each policy guarantees improvement from the previous policy, and we can obtain a series of improving polices and value functions.

Based on the above cumulative time in policy and value iteration graph, PI typically shows great increase in the speed of convergence of policy iteration (two processes) by increasing iteration since the value iteration changes little to the next policy (one process).

As the first two above graphs shows, PI converges in few iterations compared to value iteration. PI takes only 6 iterations compared to 23 iterations of VI. But as PI and VI ends up with the same level of reward in the graph, that means both algorithms converge to the optimal value function and an optimal policy at the end.

Value Iteration (VI)

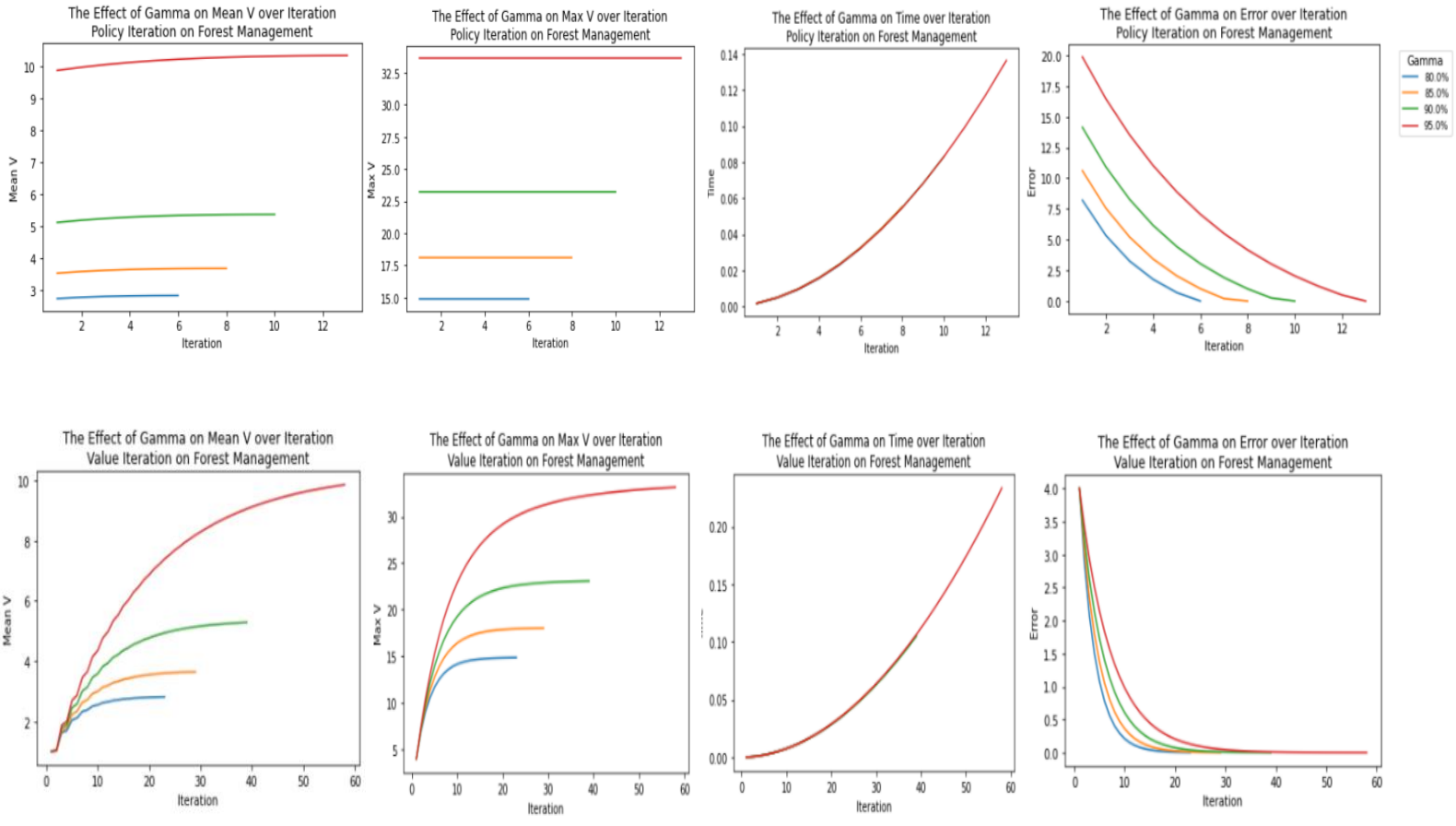
The difference of PI, VI stops after just one update of each state of policy evaluation in between each policy improvement. It improves the policy with just one step. VI also guarantees the convergence to the optimal policy like PI. If you look at VI calculation, VI update shows the identical policy evaluation except for maximizing over all actions instead of using the utility of the policy in PI.

One of disadvantage to PI has additional policy evaluation step. In the above chart, PI takes longer than VI except for 200 iterations. In case of 5000 iterations, PI takes almost 10 times longer (2 vs. 19 sec). For large number of states, VI converges a lot faster than PI.

As above graphs and chart shows, VI and PI converge to the almost same reward value for both max and mean. PI value has slightly higher reward value, but the differences are minimal. But in terms of error, PI has almost zero error since it seeks the optimal for every policy. But VI has 0.0029 error for all number of states which is still small.

Noticed that as the number of states increases (200, 500, 1000, 5000), there are almost no changes of number of iterations to converge, max value, mean value and error. Only time increases with the number of iterations by nature. The number of states seems there is no impact on the results of VI and PI for forest management problem.

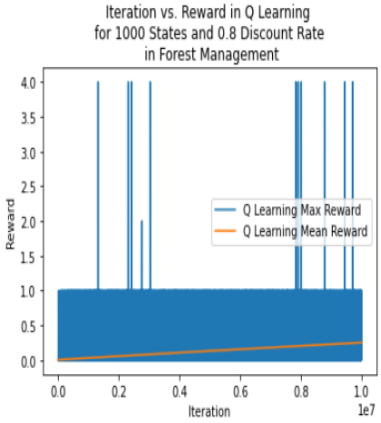
However, the gamma (discount rate) largely impacts on a max value, mean value and error except for time. Based on the below graphs of gamma tuning, when gamma increases from 0.8 to 0.95, mean value over iteration of PI increases from 2.5 to 10 (4x). VI shows similar rises from 2.5 to 10 (0.8 vs 0.95 gamma). This effect happens to max value of PI and VI as well. It shows the gamma and reward value has positive relationship. The higher gamma (lower discount rate) means the less discount of future value and it leads higher total value of rewards. But for error shows inverse relationship with gamma. Because low gamma applies big discount for future reward, it reduces uncertainly from later reward and results in lower error.

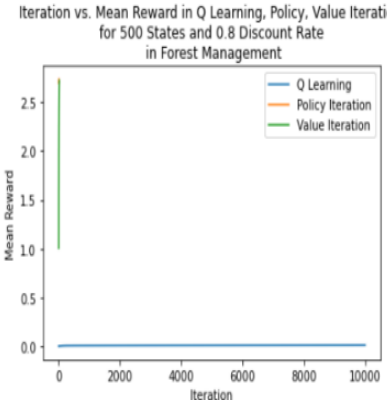


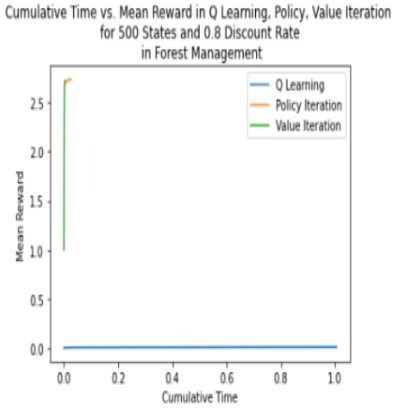
Q-Learning (QL)

My favorite reinforcement learning algorithm is QL with an epsilon greedy exploration strategy. VI and PI assume full knowledge of the MDP and both update estimates of the values of states on a basis of estimates of the values of successor states. However, QL does not update estimates based on other estimates and does not require a model. QL only needs experience, no prior knowledge of environment. It is expected to have worse results from QL compared to VI and PI, but it is closer to reality. Based on below graphs, mean reward in QL is almost close to zero compared to VI (1 – 2.7) and PI (2.8) at one iteration. That does not improve much neither with more iteration and longer time.

QL tries to learn by exploring based on what we know (exploitation). This exploration and exploitation are determined by alpha (learning rate) and epsilon(greedy) and both decays as we take more actions. In the pure greedy method, epsilon is set as zero that we take the highest value each iteration and find the optimal policy with it. This will decrease exploration and lead to local targets. In the below first two graphs, the mean rewards of QL slowly improves a little bit by increasing iterations but max reward in 1000 iteration shows a several peaks bigger than 1.0 reward, this shows the local optima based on greedy strategy.

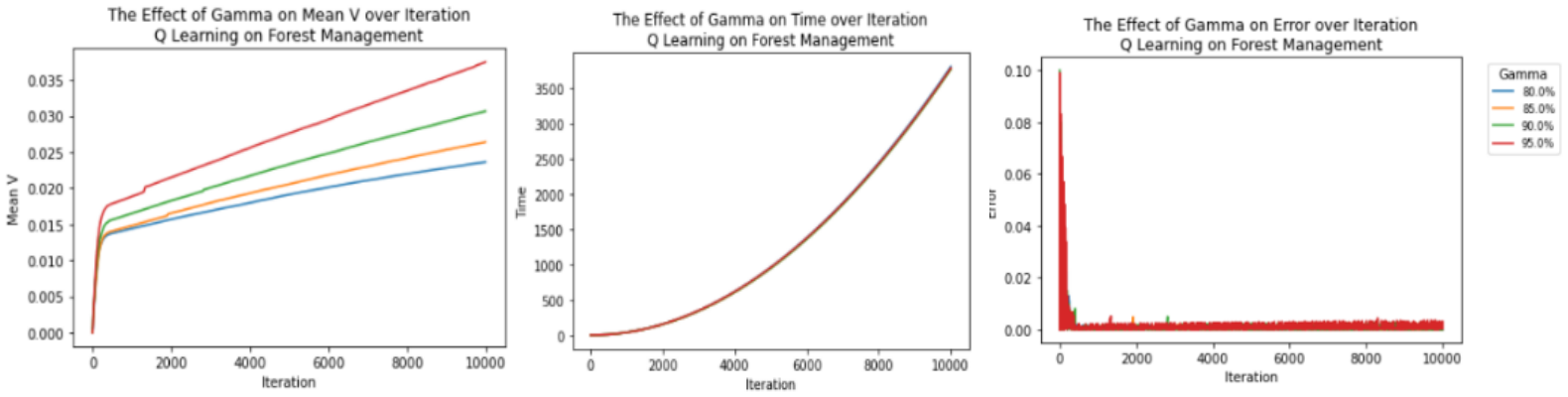






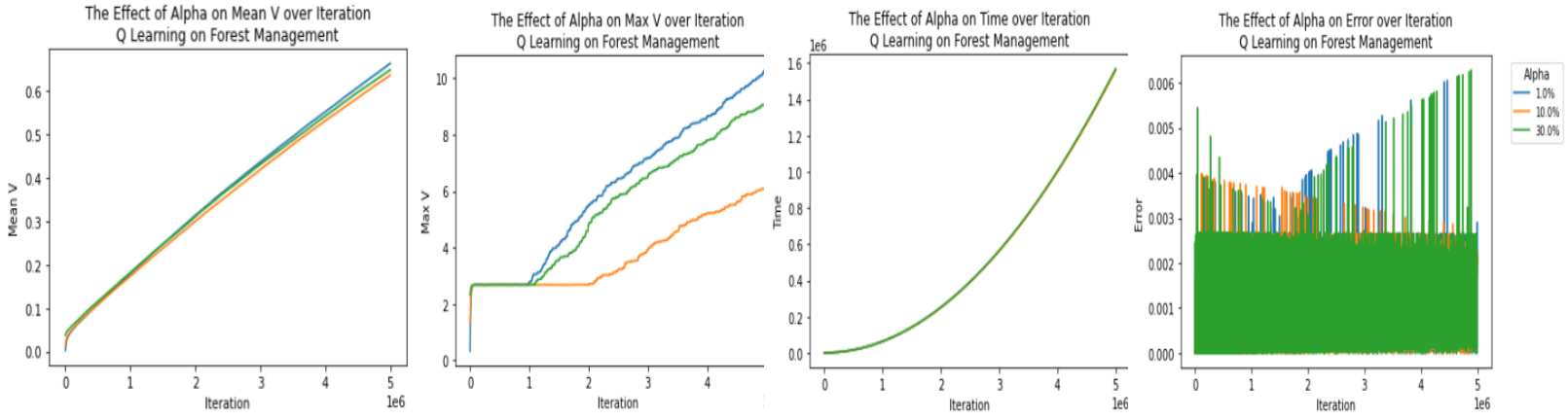
Gamma Tuning

The below QL gamma tuning shows the similar positive relationship of gamma and reward for VI and PI except for the effect of gamma on error. The higher gamma (0.95) shows extremely high error at the beginning of iteration because QL needs to learn from the limited experience without initial environment information. The higher gamma (less discount) will create a big error, but error will be stabilized at the low level after a few iterations. In addition, QL takes a significant amount of time (3500 sec) compared to 0.6 sec of PI and 0.1 sec of VI for 1000 iteration since QL needs time to learn. But there is no effect of gamma changes on time for QL.



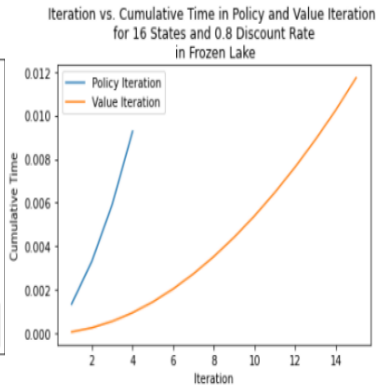
Alpha tunning

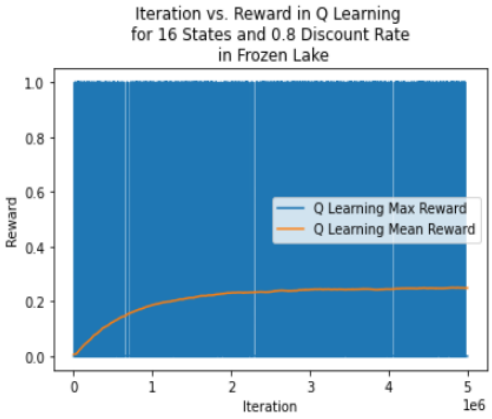
In my setting (gamma = 0.8), alpha of 1% performs the best. Next is 30% and last performer is 10%. It seems there is no linear relationship between alpha and reward in QL. But with all of three learning rates (1%, 10%, 30%), mean value of reward increases with higher iteration. But there is no impact on time over iteration in QL. For error, at the beginning 30% alpha creates higher error but error of 1% and 30% increase together later. But 10% error decreases continually over iteration. That shows some dynamics which is nonlinearity relationship between error and learning rate in QL.

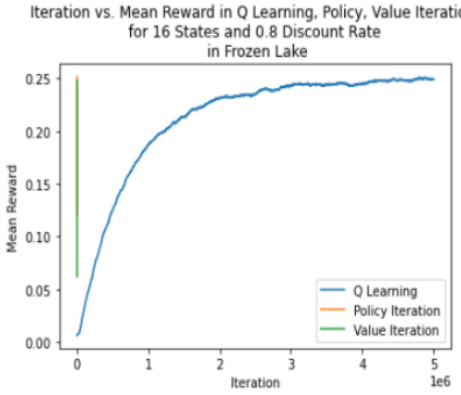


3 FROZEN LAKE (FL)

The FL navigates the frozen lake, avoids the hole, and reaches the thrown frisbee for a return. We will receive reward of 1 when we reach the goal. The games ends when it falls into hold or get to the goal. FL will have new goal on the lake every time and action needs to be taken accordingly. It is interesting to see how reinforcement learning can solve the randomness of this problem. My FM problem has a small state (4 x 4) of grid world problem. Gym library (FrozenLake-v1) is used to set up environment for FL.







Refer to the Forest Management for description of VI, PI and QL and how to their convergence is defined, since this is the same for Frozen Lake. The gamma (discount rate) is set as 0.8 as same as FM is.

Policy Iteration (PI)

Just like FM problem, PI converges in few iterations (4 iterations) compared to value iteration (15 iterations). PI iteration time (0.007 sec) is shorter than VI iteration time (0.01 sec) but both PI and VI converges to the same mean value of reward (0.25) later. PI still shows great increase in the speed of convergence of policy iteration

by increasing iteration since the value iteration changes little to the next policy (one process). Max value of reward of both is converged to 1.63 together as well. If we look at mean reward changes per cumulative time (sec), VI performs better at the beginning, next PI overpasses VI, and finally both converges to the same level of reward (Time vs. Mean value reward graph).

Value Iteration (VI)

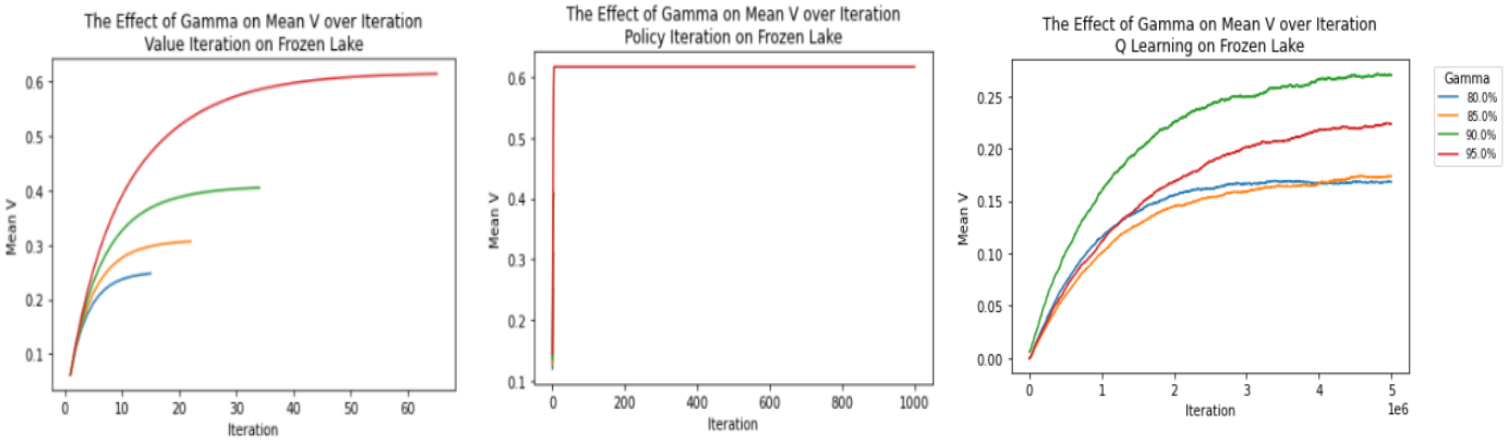
Contrary to FM (large state), VI takes longer than PI for Frozen Lake. In fact, FM shows VI takes longer than PI as well only for relatively small state of 200. In small state, VI takes longer than PI for both FM and FL but it changes VI becomes faster over iterations in large state.

Q-Learning (QL)

Based on Iteration vs. Mean reward in QL graphs, the mean reward in QL improves to the same level of PI and VI. It just takes long time and more iterations. PI and VI converges quickly with 0.01 sec but QL take more than 250 sec. In small state, graphs clearly show all three algorithms converges to the same level of rewards at the end.

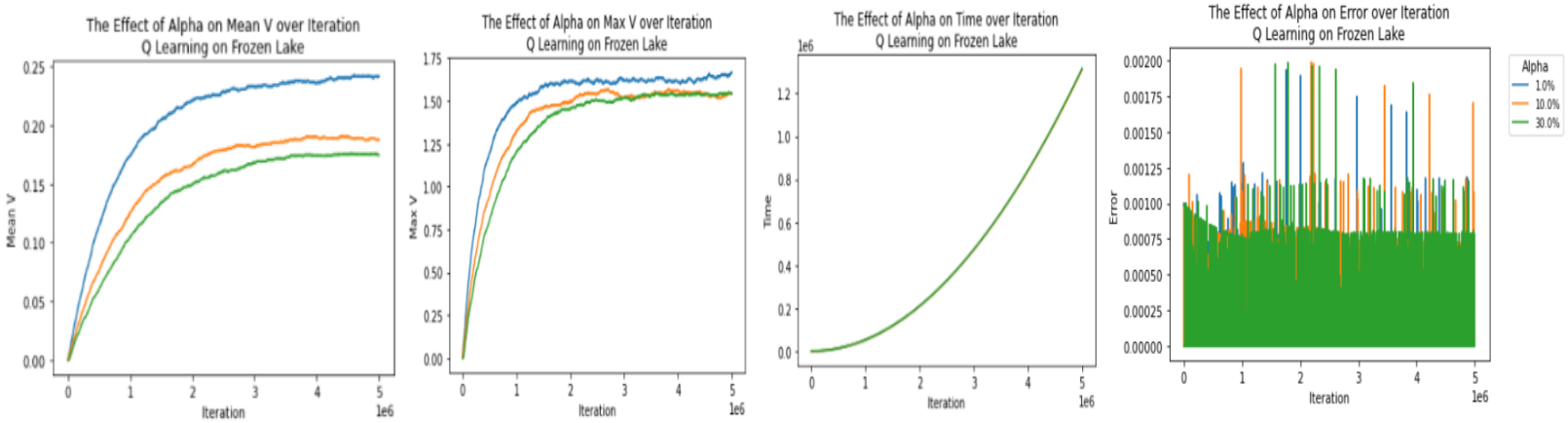
Gamma tunning

As expected, the below VI gamma tuning shows the similar positive relationship of gamma and mean value of reward. For PI, it converges two fast and does not show the performance clearly, but 0.95 gamma reaches the highest mean value of reward in a few iterations. However, QL performs differently compared to a large state of FM. 0.9 gamma performs the best over all over iterations. Next is 0.95 gamma after 1,000,000 iterations. After enough iterations, the third is 0.85 gamma and the 0.8 gamma. There is no linear relationship between gamma and value of rewards in QL. Also, the order changes over iterations between 0.95, 0.85, 0.8. There must be some complicate dynamics in QL. But 0.9 gamma is superior to all others over iterations.



Alpha tunning

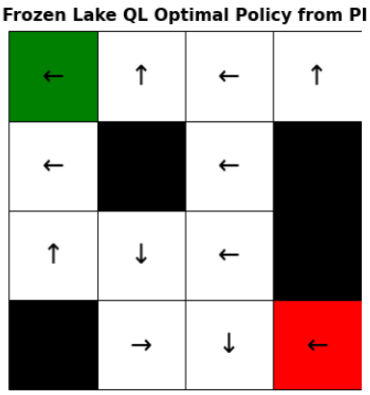
As expected, 1% performs the best in FL QL like FM and alpha changes have no impact on time over iteration in QL. But the second best is 10% alpha, not 30% in large state of FM. Alpha performs differtly in small state. For error it is not clear to see any patterns between alpha.

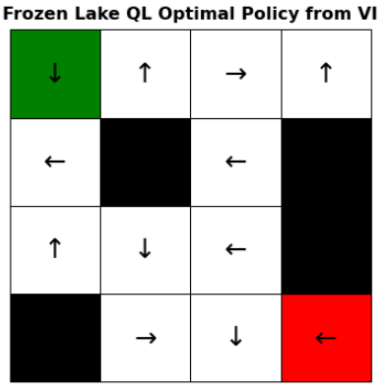


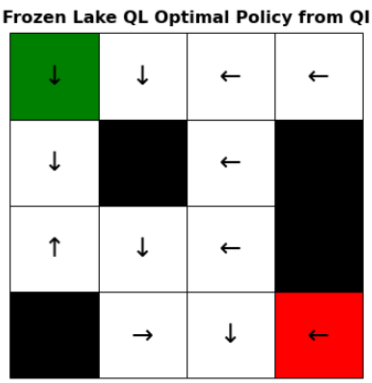
Comparison

Colors in the below grid world indicates white is safe spot, black is hole, green is start, and red is goal. There are four actions of arrow that correspond to the directions (left, down, right, and up).

As below optimal policy shows in grid world (gamma = 0.8), PI and VI converges the same level of reward at the end but the optimal policy details are not exactly same. Compared to QL, both algorithms converge very quickly to the same mean and max value of rewards. Unlike a large state of FM, the QL reaches the same level of reward as well with PI and VI for a small state of FL. But QL takes much longer (250 sec) compared to the less than 1 sec of PI and VI. In case of FL (small state), PI is the fastest algorithm. One of reason that QL takes longer can be row 2 and row 3 of QL strategy has a opposite direction of actions each other. This kind of conflicts do not exist in PI and VI optimal policy. Apparently, if the states become larger, it will greatly impact on QL's performance and time since QL needs to check every possible way to find the optimal policy.







4 CONCLUSION

In terms of time, PI is faster than VI in small state but VI is faster for FM (large state). Both PI and VI can quickly find the optimal strategy since they already know the environment.

Both PI and VI converges to the same level of reward value for both small and large states. But QL has more chance to find the optimal answer in small states. We noticed all three (PI, VI and QL) converged to the optimal strategy in small state (FL). For large state, QL struggles than a small state, taking a lot longer time and producing a much lower value of reward. The states size does not impact on PI and VI with environment knowledge, but the states size matters to QL and QL performs better in a fewer state.

5 REFERENCE

https://pymdptoolbox.readthedocs.io/en/latest/api/mdp.html

https://pymdptoolbox.readthedocs.io/en/latest/api/mdp.html

https://pymdptoolbox.readthedocs.io/en/latest/api/mdp.html

https://gym.openai.com/envs/#toy_text

https://gym.openai.com/envs/FrozenLake-v0/