# CS 7641: Assignment 4

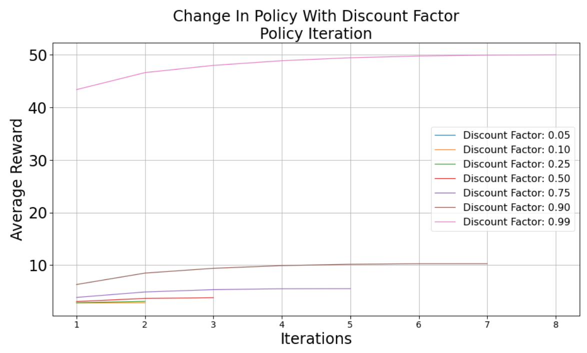
## Background

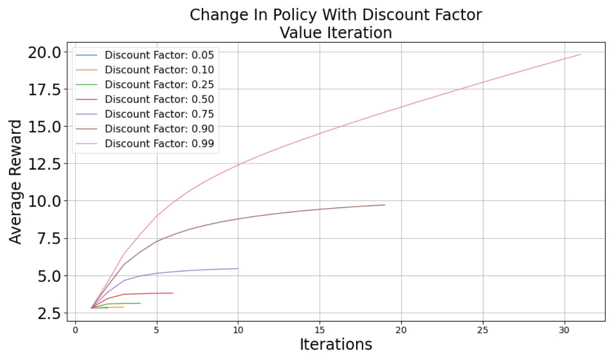
For this project, I explored the Frozen Lake (grid-based) and Forest Management (non-grid) problems. The Frozen Lake problem consists of an agent that must navigate frozen tiles (safe) and avoid the holes (dangerous) to go from the starting point to the goal. Each cell in the grid represents a state, and the agent can take various actions (up, down, right, or left). The Frozen Lake problem is interesting to me because of the sparse reward structure embedded into the problem. The sparse reward structure demonstrates the challenges of learning and mimics real-world scenarios. The Forest Management problem is made up of two actions: "wait" is to maintain the old forest and "cut" is to cut wood for selling. One of these two actions are decided each year along with the probability *p* that a fire burns the forest down. The agent learns from the interactions with the forest environment over time to adapt its decision-making policy. I found the Forest Management problem to be fascinating because it involves complex interactions between different factors such as tree growth, harvesting, and reforestation. Additionally, the Markov Decision Process (MDP) formulation incorporates uncertainty into the problem particularly through the probability of a fire occurring. This MDP framework also involves tradeoffs between exploitation and exploration. I also ran the Q-Learning model-free reinforcement algorithm on both the Frozen Lake and Forest Management problems. Q-Learning aims to learn the values of state-action pairs (Q-values) through both exploration and exploitation. It learns from experiences to iteratively update and improve estimates of Q-values.

I believed that for the Forest Management problem, increasing the discount factor (gamma) would prioritize long-term rewards and result in higher cumulative rewards. I also hypothesized that increasing alpha (learning rate) will result in faster convergence at the cost of exploration, so I believed that a lower alpha value would be optimal. I also thought that higher epsilon values will promote greater exploration and slow down convergence. For the Frozen Lake problem, I believed that increasing gamma values would increase the amount of time it takes to iterate through the state-action pairs. I also hypothesized that bigger map sizes would result in slower convergence and involve more states.

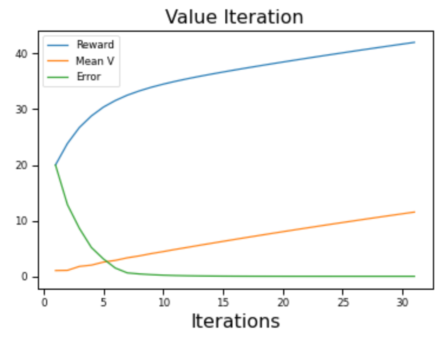
## Forest Management Problem

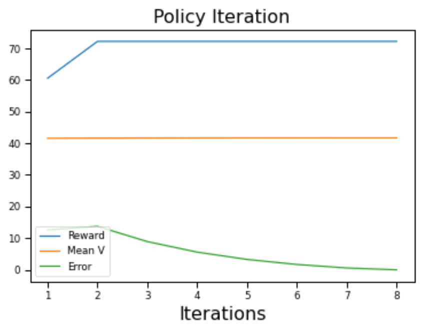
To construct the Forest Management problem, I set the size = 10 states, R1 = 10, and R2 = 20. R1 refers to the reward associated with a favorable state, while R2 refers to the reward associated with unfavorable states. I also set the probability of a fire occurring in the forest to be 0.3. Then, I examined how changing the discount factor (gamma value) can impact the average reward at different iterations for both value iteration and policy iteration.





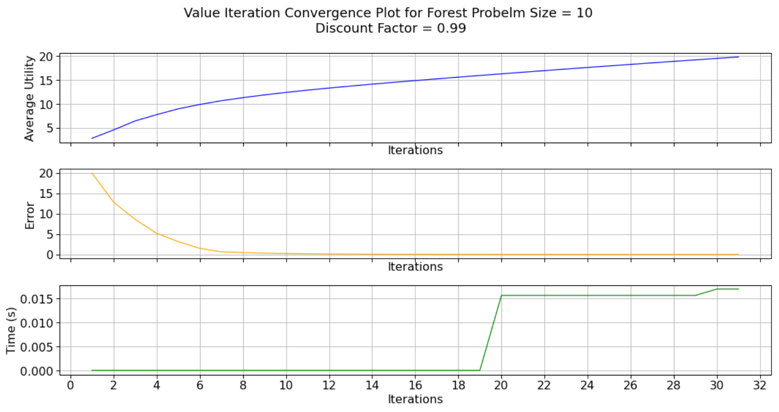
It is apparent from the plots above that the average rewards generally stay the same or increase as the number of iterations increase. For both policy iteration and value iteration, the average reward increases and converges at higher iterations at higher values of the discount factor. The value iteration plot is interesting in that convergence isn't achieved when the discount factor is 0.9 or 0.99. However, convergence is achieved at all values of the discount factor for policy iteration. The lack of convergence for some discount factor values for value iteration may come from the inherent differences in the update mechanisms for value iteration versus policy iteration. Value iteration updates all state values in each iteration, while policy iteration alternates between policy evaluation and improvement steps. This might explain why at higher discount factors where long-term rewards are heavily prioritized, value iteration may struggle to achieve convergence due to the mechanism of simultaneously updating all the states.

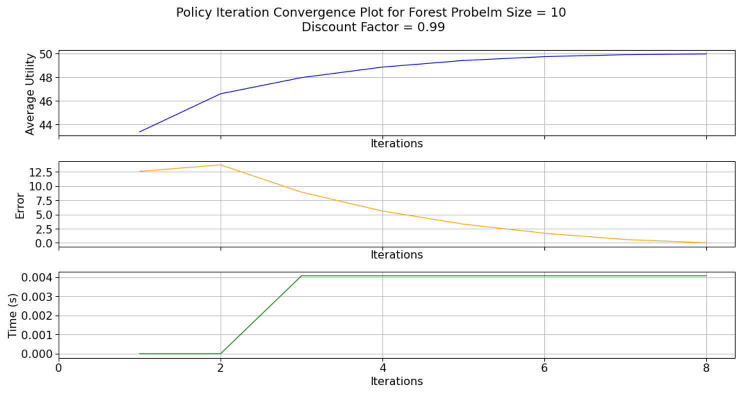


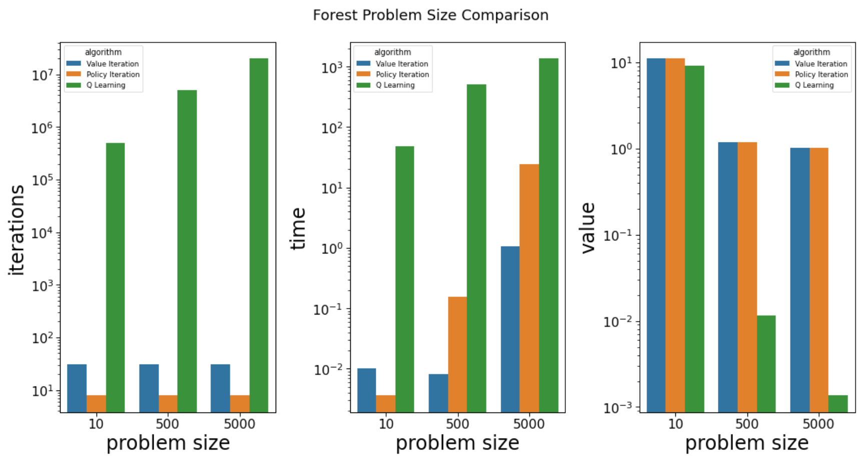


The plots above depict the relationship between the number of iterations and the following values: reward, mean value function, and error. As the number of iterations increases for value iteration, the error decreases while the reward and mean value function generally increases. Meanwhile, the reward is constant after reaching 2 iterations and the mean value function is constant at all values of iterations for the policy iteration. These differences again reflect the update mechanisms of value iteration and policy iteration in that policy iteration has less variability and incrementally updates.

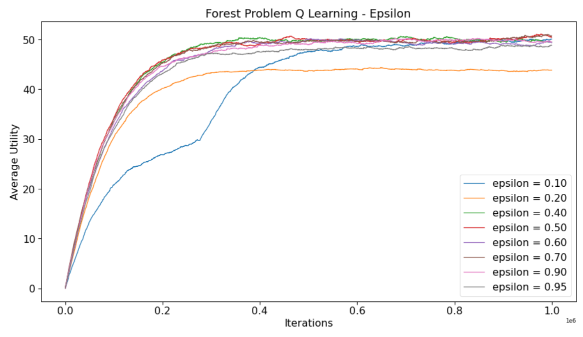
The plots below reflect the average utility, time (seconds), and error for different values of iterations for both value iteration and policy iteration when the discount factor is 0.99.

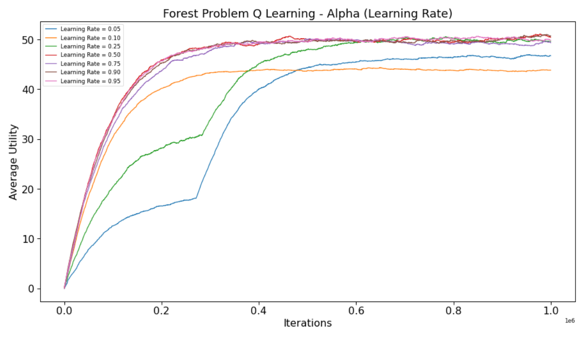






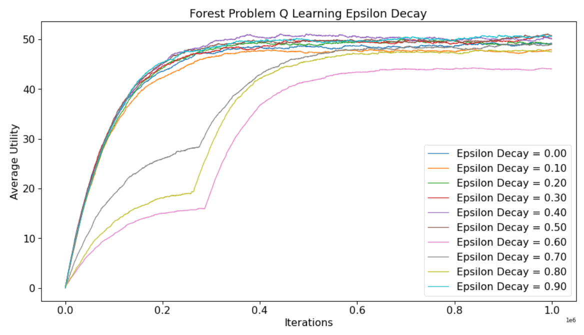
The three plots on the left depict the impact of problem size on the number of iterations, time (seconds), and value for value iteration, policy iteration, and Q-learning. Problem size does not seem to impact the number of iterations for value iteration and policy iteration, which may be due to how they're both model-based methods. As model-based methods, the value function or policy are updated based on the Bellman equations which are not sensitive to the size of the state space. The highest reward was achieved for the smallest problem size of 10 for value iteration, policy iteration, and Q-learning. Additionally, the reward value for problem size of 10 is similar for all three reinforcement learning algorithms. Smaller state spaces can make it easier for these algorithms to reach optimal solutions, thus achieving the highest reward.

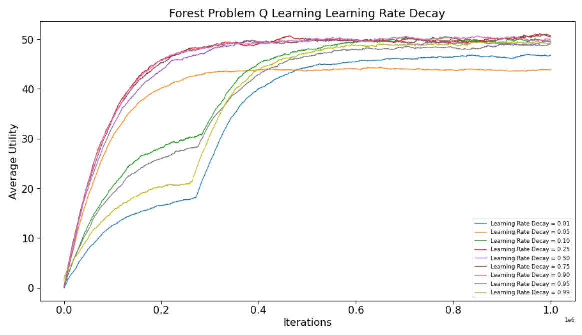




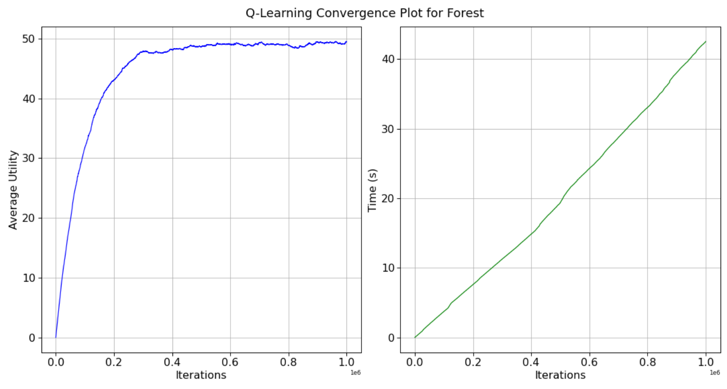
The plots above depict average utility (mean value function) for different epsilon values (left plot) and for different alpha/learning rate values (right plot). For the epsilon plot on the left, the lowest epsilon value begins to converge around 0.5 iterations while the other epsilon values converge at around 0.3. Even then, the mean value function oscillates quite a bit for most of the epsilon values (aside from epsilon = 0.2). For the alpha plot on the right, the learning rate of 0.1

has the best convergence although not the highest mean value function. I also plotted iterations vs average utility plots for epsilon decay values and alpha decay values as shown below.



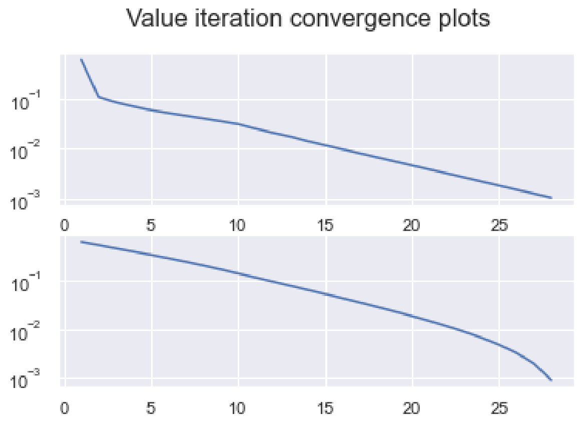


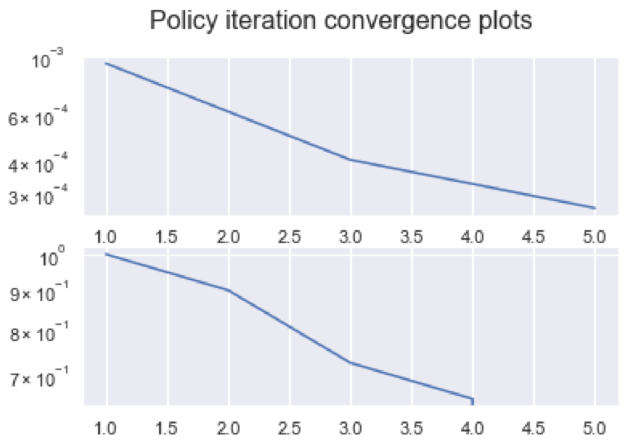
From these four plots, it is apparent that the following values are optimal for this problem: alpha decay = 0.05, alpha = 0.1, epsilon decay = 0.6, and epsilon = 0.2. I utilized these values to plot the convergence plots below.



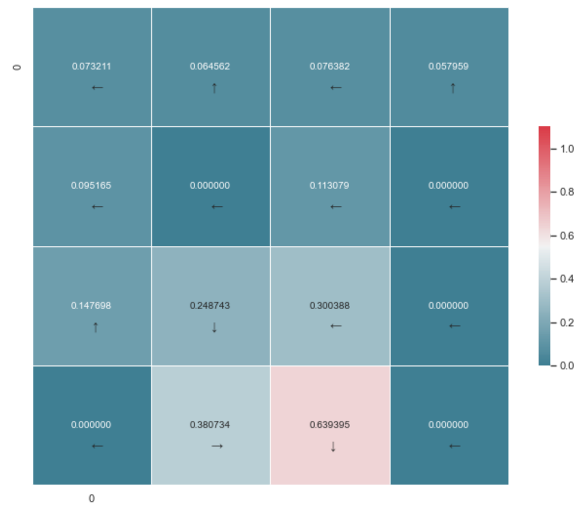
## Frozen Lake

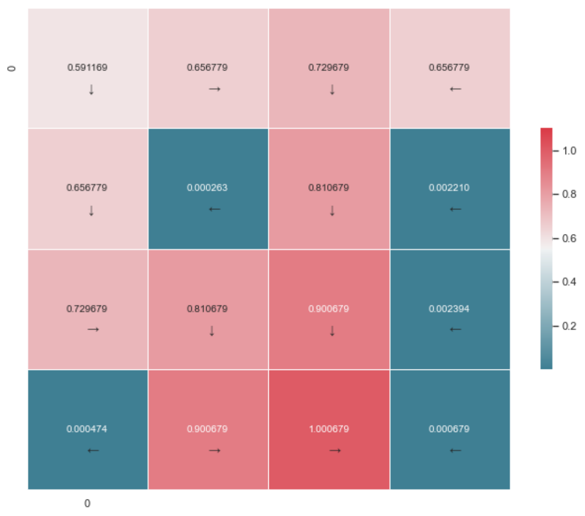
First I constructed a 4x4 grid for the Frozen Lake environment with a gamma (discount factor) value of 0.9. It resulted in the following convergence plots below. The plot on the top row indicates how fast the value function estimates are converging to a stable solution. It is decreasing for both value iteration and policy iteration, so it indicates convergence. The bottom plots reflect the distance between the current value function estimate and the converged optimal value function. Again, these plots are decreasing as the iterations increase, which indicates convergence.





The Frozen Lake problem is considered to have a sparse reward structure, making it difficult for value estimates to converge quickly due to limited feedback. Moreover, the value iteration update mechanism requires all states to have their value functions be iteratively updated regardless of whether the policy has substantially changed. This results in the value iteration requiring a higher number of iterations than policy iteration.





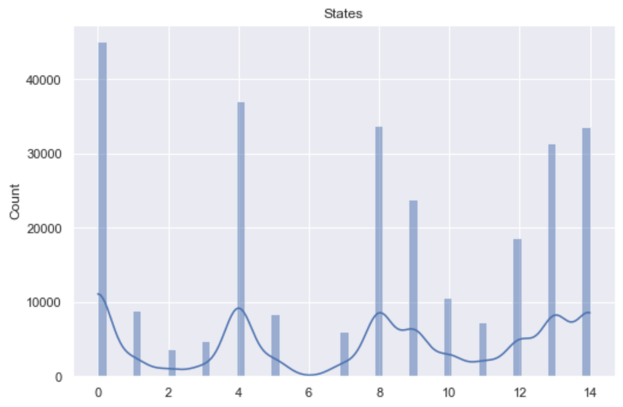
The heatmaps above represent value function estimates for each state and the corresponding action (or policy) that guides the agent on the most optimal path from the start state to the goal state. The heatmap on the left reflects the learned policy of the value iteration, while the heatmap on the right reflects the learned policy of the policy iteration method. The redder/pinker areas correspond to states with higher value estimates, thus indicating more rewarding areas in the grid.The bluer areas refer to the riskier states with lower value estimates. It is apparent from the heatmaps that the value iteration method results in riskier and low-reward states, while the policy iteration method results in safer and high-reward states. The differences in the heatmaps generated may be attributed to the differences in the policy-updating mechanisms. For instance, the value iteration method tends to be risk-averse and conservative compared to the policy iteration method. This may explain why the heatmap for the value iteration method resulted in riskier, low-reward areas as the mechanisms of the value iteration method itself emphasizes these high-risk, low immediate reward states. The policy iteration method tends to explore and exploit rewarding states more efficiently than the value iteration method, thus highlighting areas with higher rewards.

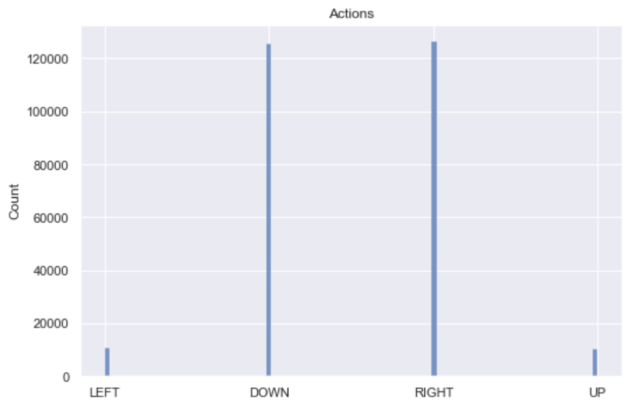
| | **Value Iteration** | | **Policy Iteration** | | |
|---|---|---|---|---|---|
| Gamma Value | Total Time (s) | Steps | Total Time (s) | Improvement Steps | Evaluation Steps |
| 0.1 | 0.23 | 3 | 0.08 | 6 | 3 |
| 0.2 | 0.07 | 5 | 0.07 | 6 | 7 |
| 0.3 | 0.08 | 5 | 0.06 | 6 | 7 |

| | | | | | |
|---|---|---|---|---|---|
| 0.4 | 0.06 | 5 | 0.06 | 6 | 10 |
| 0.5 | 0.06 | 6 | 0.07 | 7 | 10 |
| 0.6 | 0.07 | 6 | 0.07 | 6 | 14 |
| 0.7 | 0.08 | 6 | 0.07 | 7 | 17 |
| 0.8 | 0.08 | 6 | 0.07 | 5 | 27 |
| 0.9 | 0.06 | 5 | 0.07 | 6 | 45 |

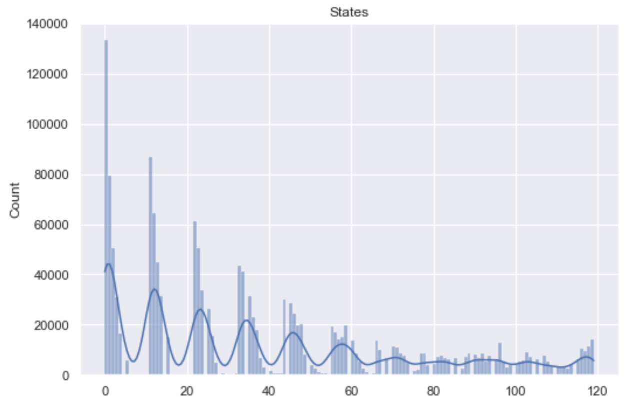
I examined how different gamma values affect the total time and steps for both the value iteration and policy iteration methods as depicted in the table above. Increasing the gamma values increases the number of policy evaluation steps but doesn't have much of an effect on the value iteration steps, policy improvement steps, or the total time for either iteration method. Changing the gamma (discount factor) does not have much of an impact on the total time for each iteration method because other factors like the complexity of the state space have more influence. Generally a higher discount factor means that future rewards are valued more than immediate rewards. I expected the higher discount factor to force the agent to sacrifice immediate rewards and take longer paths, thus resulting in longer time taken for the method. However, that was not the case. It is apparent from the table above that increasing gamma did not affect the total time taken. The policy evaluation steps estimates the value function associated with a particular policy by estimating the expected future rewards from each state-action pair. Changing gamma affects how the agent weighs future rewards so it makes sense that increasing gamma values means that the agent will dedicate more importance to future rewards and result in more iterations for convergence. This might explain why increasing gamma will increase the number of policy evaluation steps.

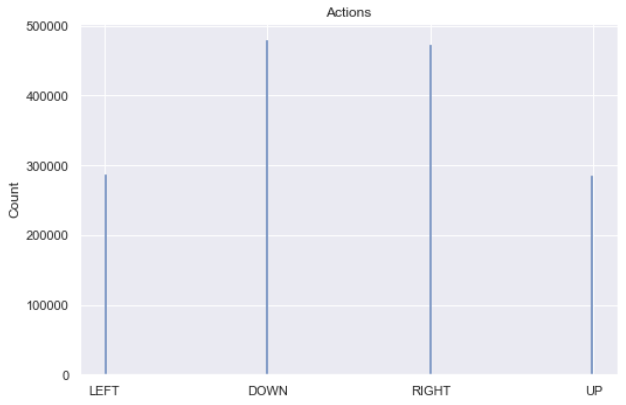
Next, I examined how the agent conducts itself on various map sizes: 4x4, 7x7, 9x9, and 11x11. I plotted the count of states at different iterations. I also plotted the total number of actions for each of the following possible actions: left, down, right, and up. I visualized the last frame of the simulation. The frame will reflect if the agent learned and pursued a policy to solve the task well by showing the agent on the treasure tile (goal state). Lastly, I visualized a heatmap to reflect the learned Q-values and represent the best action for each state. The darker blue areas reflect high-reward, low-risk states.



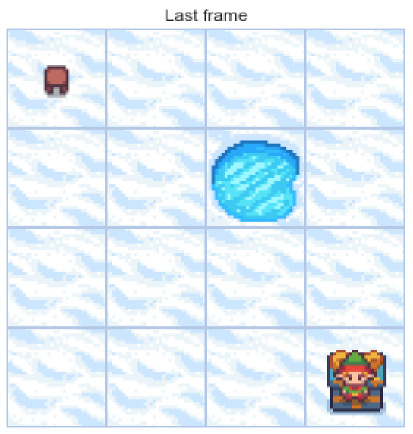


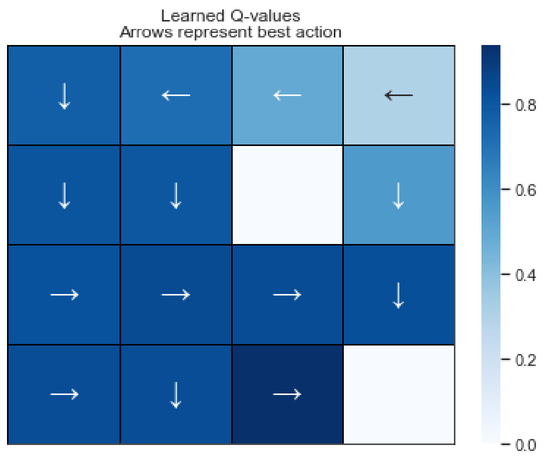
Above are the states' count plot and actions count plot for the 4x4 map size of the Frozen Lake problem.



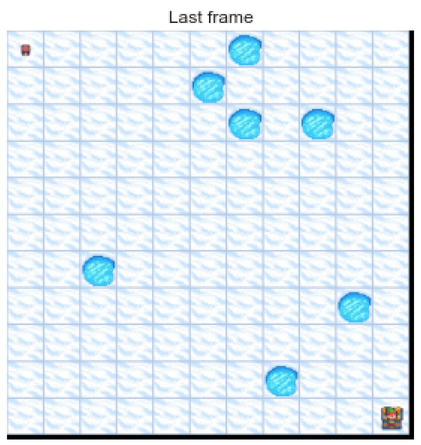


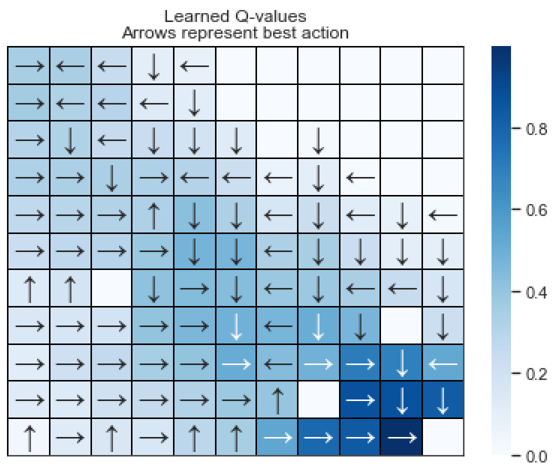
Above are the states' count plot and actions count plot for the 11x11 map size. Increasing the map size resulted in a substantial increase in the number of iterations, number of actions taken, and number of states. This makes sense as a bigger map means that there is more to explore and exploit.



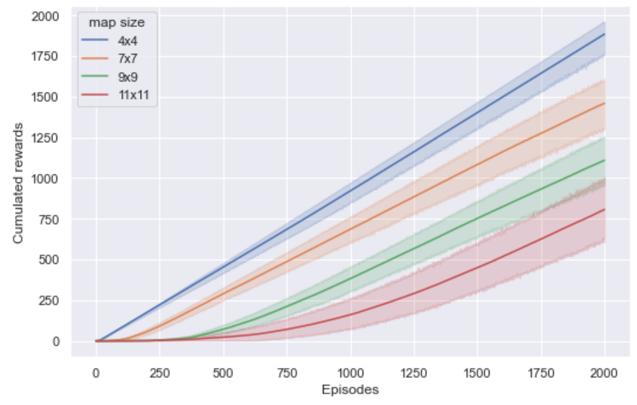


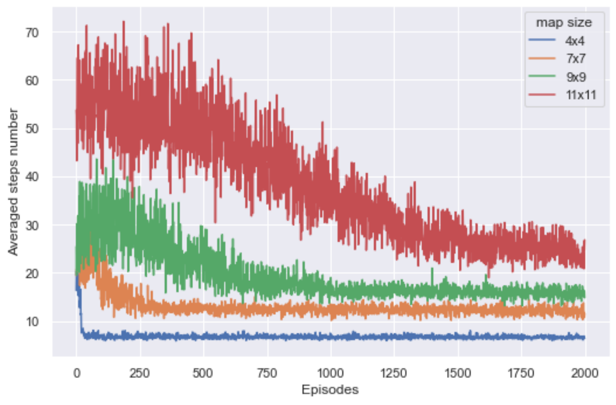
The visualizations above are of the last frame and the heatmap of learned policy for the 4x4 size. It is clear from the heatmap that there are lots of high-reward areas. Interestingly, the goal state has a learned Q-value of zero. I believe this is to signify that there is no further expected reward from that state onwards as it is the goal state.





The last frame and heatmap above corresponds to the 11x11 map size. It is apparent from the last frame that there are more "holes" that the agent risks falling into. Moreover, the heatmap shows that the agent will neglect the tiles in the upper right and lower left corners in comparison to the rest of the grid. Interestingly, the agent appears to choose more "down" and "right" actions for the 11x11 grid compared to the smaller map sizes.





The plot above on the left reflects the cumulative sum of rewards and the plot on the right reflects the number of steps needed per episode. The cumulative sum of rewards increase as the number of episodes increase for all four map sizes, which indicates that the agent is learning in each environment. The plot on the right reflects that the 4x4 map converges the quickest compared to the other map sizes. This just points to how a small state space can minimize the amount of exploration/exploitation needed thus achieving fast convergence rates.

**Conclusion**

With the Forest Management problem, I was wrong about my hypothesis regarding the impact of the alpha (learning rate) and epsilon values on the convergence rates. It is apparent that the low alpha and epsilon values resulted in greater reward and better convergence. With the Frozen Lake problem, I was wrong in that increasing the gamma value had no substantial impact on the total time taken to iterate through the state-action pairs. I was right in that the smallest map size (4x4) resulted in the fastest convergence compared to the other three map sizes.

This project was particularly exciting to explore and compare the different reinforcement learning algorithms. Value iteration appears to focus on exploiting known values. Policy iteration utilizes policy improvement and policy evaluation to balance both exploration and exploitation. Q-learning focuses on exploration by continuously updating the Q-values based on interactions with the environment.

**Works Cited**

- https://pymdptoolbox.readthedocs.io/en/latest/api/mdp.html
- https://medium.com/@m.alzantot/deep-reinforcement-learning-demysitifed-episode-2-policy-iteration-value-iteration-and-q-978f9e89ddaa#:~:text=Value%20iteration%20computes%20the%20optimal,converge%20to%20the%20optimal%20values
- https://gymnasium.farama.org/tutorials/training_agents/FrozenLake_tuto/
- https://medium.com/sequential-learning/optimistic-q-learning-b9304d079e11
- https://www.baeldung.com/cs/epsilon-greedy-q-learning