Markov Decision Processes

Nayana Saji
CS-7641
nsaji6@gatech.edu
903967749

***Abstract*—As part of this assignment, different reinforcement learning algorithms such as Policy Iteration, Value Iteration and Q-Learning are compared and contrasted using 2 non grid world problems: Forest Management and Hotter Colder.**

I. Problems

A. Forest Management

The Forest Management Problem is a decision making challenge where the goal is to balance two objectives: maintaining an old forest for ecological and wildlife benefits and maximizing profits through harvesting trees. The forest is represented by a set of states ranging from a young forest, denoted as state 0, to the oldest and most mature forest, denoted as state S-1. Each year the forest manager has two possible actions. The first action is Wait where the forest is allowed to grow naturally, progressing to the next state unless it is already in the oldest state.However, there is a probability p that a fire will occur, resetting the forest to the youngest state regardless of the chosen action. The second action is Cut" where the forest is harvested. The reward for cutting depends on the maturity of the forest, with older forests giving higher rewards.

Hiive mdptoolboxs' implementation of forest management was used for this assignment. The total number of states for forest management was chosen to be 625.

This problem is interesting because it highlights critical trade-offs between short-term gains and long-term sustainability, reflecting realworld challenges in resource management. The forest manager must decide if to harvest immediately for profit or to wait and allow the forest to mature, but with risk of losing everything to a fire. The inclusion of fire probability introduces uncertainty, adding a layer of complexity to the decision-making process. Managers must balance risk mitigation with strategic long term planning, making the problem an excellent test case for reinforcement learning algorithms.

The problem's structure is particularly suited for reinforcement learning because it involves sequential decision-making with delayed rewards. Reinforcement learning algorithms must balance exploration, such as experimenting with waiting strategies to learn long-term impacts, and exploitation, such as harvesting for immediate rewards. This dynamic makes the problem ideal for testing approaches like Q-learning , value iteration ,policy iteration.The stochastic nature of the problem, driven by the risk of fire, creates a challenging fitness landscape where short-term local optima, such as cutting too early, can prevent finding globally optimal solutions. T The dual objectives, stochastic transitions, and inherent trade-offs make the Forest Management Problem interesting.

B. Hotter-Colder

The Hotter-Colder Problem is a decision-making probelm where the goal is to guess a target number within a range while balancing the accuracy of guesses and the number of attempts. The target number is randomly selected within a range of [1500,1500] and the agent must make guesses by taking actions within a continuous space - bounded by [3000,3000]. The environment provides feedback in the form of observations that indicate the closeness of the guess to the target number. Observations are discrete states ranging from Very Far to Very Close, based on the distance between the guess and the target.

This problem was modelled directly from openAi gym - v0.20.0.

This is an interesting problem because it combines a small state space with a large action space, requiring the agent to navigate a continuous action space while interpreting feedback from discrete states. This creates a balance of exploration and exploitation since as the agent must search the action space effectively while using the feedback to refine its guesses. And, the reward function is non-linear, scaling quadratically with proximity, which is like real-world problems where rewards have diminishing returns or sharp increases based on precision. This complexity tests an algorithm's ability to find and optimize non-linear relationships between actions and rewards.

The problem also features mixed dynamics, with a discrete state space and continuous action space, making it challenging for algorithms that are designed primarily for one type of space. This hybrid nature allows for testing the adaptability of various algorithms in handling environments with both discrete and continuous elements. Furthermore, the problem is highly deceptive - focusing onlt on improving rewards for closer guesses may lead an agent to overlook strategies for narrowing the action range.

II. Hypothesis

A. Forest Management

Policy Iteration will be efficient for the forest management problem due to its state space - 625 states, enabling quick policy evaluation and stable convergence. While it requires fewer iterations than Value Iteration, the time per iteration is higher due to solving policy evaluation equations.

Value Iteration will converges faster computationally as it directly updates state values using the Bellman equation. Though it may require more iterations to stabilize compared to Policy Iteration, the structured state space ensures rapid convergence, making it effective for problems with simple transitions.

Q-Learning might take the longest to converge because it is model-free and relies on sampling through exploration. While the state space aids learning, the need for extensive exploration and balancing exploitation slows convergence. Despite this, Q-Learning will be able to achieves good performance over time, albeit slower than policy-based methods.

B. Hotter Colder

Policy Iteration is expected to performs well on the Hotter-Colder problem due to the small state space (4 states), enabling quick and stable convergence. While the large continuous action space increases computational effort, the deterministic transitions allow efficient policy refinement in a small number of iterations.

Value Iteration is expected to converge faster than Policy Iteration in terms of computation by directly updating state values through the Bellman equation. Although it may require more iterations to stabilize, the small state space ensures rapid convergence, and the deterministic nature of the problem simplifies updates.

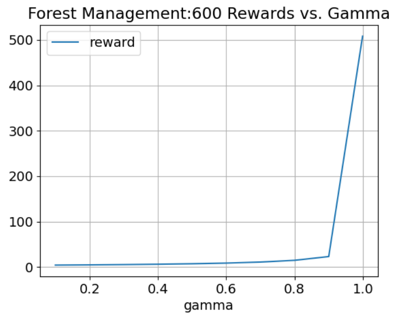
Q-Learning will take longest to converge as it is model-free and relies on exploration to learn optimal actions. The continuous action space slows learning, especially early on, but the small state space aids refinement over time.

III. Forest Management

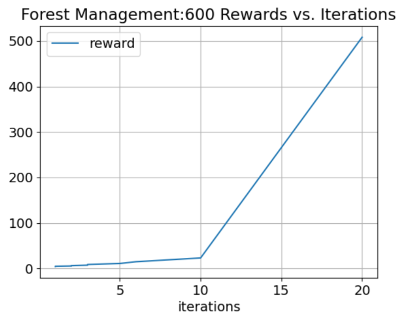
A. Policy Iteration

a) Discount/Gamma vs Rewards: From Figure 1.a, It was found that increasing the discount factor helps increase the cumulative reward in the forest management problem.A higher rvalues future rewards more significantly encouraging the agent to adopt long-term strategies. This allows the agent to wait for the forest to regrow before cutting, leading to higher overall rewards. A low gamma however, causes the agent to prioritize immediate rewards, resulting in suboptimal policies and lower cumulative rewards. The exponential growth of rewards as gamma approaches 1 suggests that the problem heavily rewards sustainable, long-term strategies over short-sighted ones.

b) Rewards vs Iterations: From Fig 1.b It was also found that the number of iterations significantly impacts the cumulative rewards achieved by Policy Iteration. Starting with suboptimal policies, the algorithm gradually improves over iterations by refining the value estimates and learning and finding better policies. The initial flat phase in rewards indicates that early iterations focus on exploring and evaluating policies, which may not immediately yield good gains. However as the iterations progress, the agent converges to an optimal policy, leading to exponential increases in cumulative rewards. This behavior shows the iterative nature of policy iteration, where improvement depends on effective policy evaluation and policy improvement steps



(a) Rewards vs Gamma



(b) Rewards vs Iteration

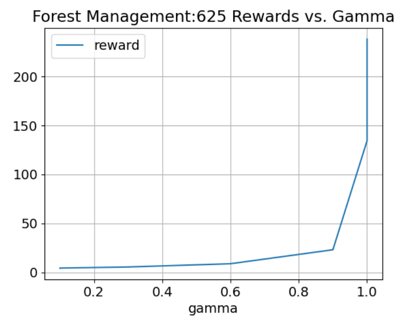
Fig. 1: Policy Iteration on Forest Management with 600 states

*c) **Hypothesis Evaluation**:* The **hypothesis holds true**. Policy Iteration performs well on the forest management problem because the state space (625 states) allowing for quick and stable convergence. Increasing the discount factor (γ) leads to higher rewards as the agent learns to adopt long-term strategies, like waiting for the forest to regrow before cutting. Early iterations show limited improvement while the algorithm refines value estimates, but rewards grow rapidly as the policy converges. Policy Iteration does takes more time per iteration due to solving equations,.

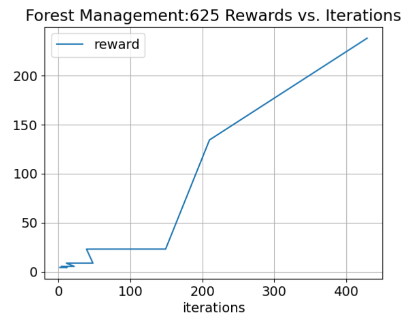
B. Value Iteration

a) Discount/Gamma vs Rewards: From Fig 2.a, It was found that increasing the discount factor gamma in Value Iteration helps the reward grow significantly. At lower gamma values (e.g., gamma = 0.1), rewards remain low because the agent prioritizes immediate rewards by cutting trees early. However, as gamma approaches 1, the rewards grow sharply. This indicates that a higher gamma enables the agent to value future rewards more, encouraging a sustainable strategy of waiting for forest regrowth before cutting. The sharp increase near gamma = 1 suggests that the forest management problem heavily rewards long-term planning, which only becomes fesiable with high gamma.

b) Rewards vs Iterations: From Fig 2.b It was also found that rewards increase over iterations, with a slow start followed by a sharp rise and eventual convergence. In the



(a) Rewards vs Gamma



(b) Rewards vs Iteration

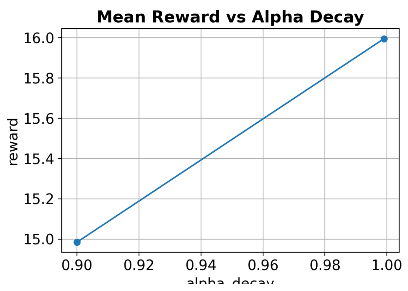
Fig. 2: Value Iteration on Forest Management with 625 states

early iterations, the value estimates are far from optimal, and rewards increase minimal as the algorithm chooses immediate rewards through the state space. Around 200 iterations, the rewards grow significantly, indicating that the agent begins to capture the long-term benefits of waiting for regrowth. After this phase, the growth in rewards becomes gradual as the value function and policy converge to optimality. This behavior show the mechanics of Value Iteration, where the Bellman equation iteratively evaluates the long-term effects of actions.

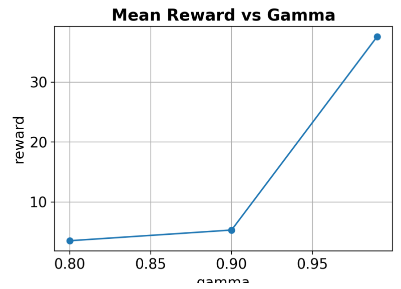
*c) **Hypothesis Evaluation:*** The **hypothesis holds true**. Value Iteration works well on the forest management problem due to the state space (625 states), which allows quick updates using the Bellman equation. Higher discount factors (γ) lead to better rewards, as the agent learns to choose long-term strategies like waiting for regrowth before harvesting. Early iterations show slow progress, but rewards rise sharply around 200 iterations as the algorithm captures the benefits of long-term planning. While Value Iteration requires more iterations than Policy Iteration, it converges quickly and is well-suited to structured problems like this.

C. Q-learning

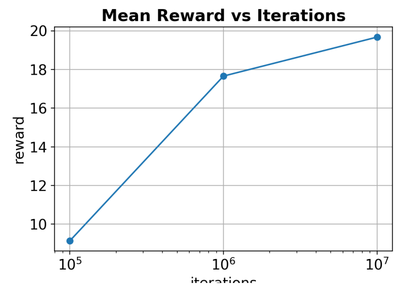
a) Mean Rewards vs Alpha Decay: From Fig 3.a,It was found that increasing the alpha decay in Q-Learning improves the mean reward over iterations. At lower alpha decay (e.g., 0.9), the reward is lower because the agent's learning rate decays quickly, limiting its ability to update its Q-values with new information. However, as the alpha decay approaches 1, the agent continues to learn and incorporate new observations



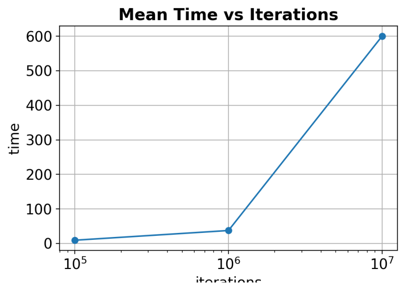
(a) Mean Rewards vs Alpha Decay



(b) Mean Reward vs. Gamma



(c) Mean Reward vs. Iterations



(d) Mean Time vs. Iterations

Fig. 3: Q-Learning on Forest Management(S=625)

over a longer period, leading to higher cumulative rewards. This suggests that in the forest management problem, a slower decay of the learning rate allows the agent to better balance cutting and waiting strategies, resulting in better policies.

b) Mean Reward vs. Gamma: Fomr Fig3.b, It was found that increasing the gamma in Q-Learning significantly boosts the mean reward. At lower gamma values (e.g., gamma = 0.8), the agent focuses on immediate rewards by cutting trees early, resulting in lower cumulative rewards. However, as gamma approaches 1, the rewards increase sharply because the agent values future rewards more and learns to wait for forest regrowth before cutting. This indicates that in forest management, long-term strategies yield higher rewards, and the sharp increase near gamma = 1 highlights the importance of sustainable planning. The same is observed in Policy Iteration and Value Iteration.

c) Mean Reward vs. Iterations: It was found that increasing the number of iterations in Q-Learning improves the mean reward. In the early iterations, the reward grows slowly as the agent explores the state space and takes suboptimal actions like cutting too early. However, as iterations increase, the agent transitions from exploration to exploitation, learning better policies that balance cutting and waiting, leading to a steady increase in rewards. The somewhat plateauing growth at higher iterations suggests that the agent has converged to an optimal policy in the forest management problem.

d) Mean Time vs. Iterations: It was found that increasing the number of iterations in Q-Learning also increases the computation time. In the early iterations, the agent updates Q-values quickly while exploring the state space. However, as iterations grow, the updates become more focused on refining the Q-values across the large state space of 625 states, resulting in significantly higher computational costs. This behavior reflects the complexity of the forest management problem, where the large state space demands extensive exploration and computation to learn optimal policies.

e) Hypothesis Evaluation: The **hypothesis holds true**. Q-Learning takes the longest to converge on the forest management problem,as expected, because it relies on exploration and sampling transitions. With proper tuning of hyperparameters like alpha decay and gamma, Q-Learning steadily improves rewards over time, eventually reaching an optimal policy.

Higher alpha decay allowed the agent to learn for longer, and higher gamma encouraged long-term strategies, such as waiting for regrowth before cutting. Early iterations focused on exploration, resulting in suboptimal actions like cutting too early, but rewards increased as the agent shifted to exploitation. The plateau in rewards reflects the eventual convergence of Q-Learning.

Despite its effectiveness, Q-Learning requires more episodes and computation time compared to policy-based methods due to the lack of direct access to transition dynamics. These results affirm that Q-Learning is slower but capable of learning optimal policies in structured problems like forest management.

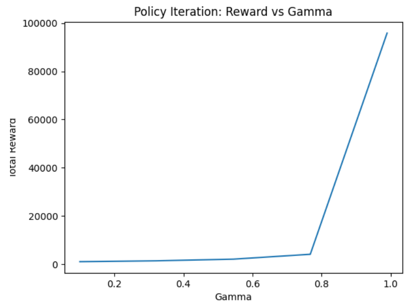
D. Best Policy

It was fouund that policy iteration and value iteration produced same optimal policy showing their effectiveness in optimizing long-tem rewards. The only wait for the youngest tree and the oldest 20 trees all of the rest are cut. In these policies, the action Wai (W) is assigned to younger forests and the oldest, most mature forests, reflecting the benefit of allowing regrowth to maximize rewards. The majority of states are marked Cut (C), indicating the economic incentive to harvest at intermediate stages.

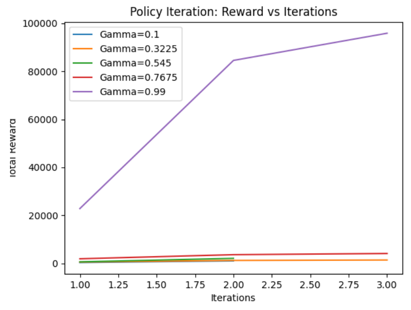
Q learning optimal policy looks very different from that of policy iteration and value iteration, it does not have structured zones for W and C. The scattered actions suggest the algorithm did not fully converge, mostly due to insufficient episodes or suboptimal exploration-exploitation balance. As a model-free method, Q-Learning does not directly leverage the problem's transition dynamics, making it harder to capture the structured reward patterns seen with model-based methods.

IV. Hotter-Colder

A. Policy Iteration



(a) Rewards vs Gamma



(b) Rewards vs Iteration

Fig. 4: Policy Iteration on Hotter Colder

a) Rewards vs Gamma: It can be seen from Fig 4.a, total rewards increase sharply as γ approaches 1.0, while remaining low for γ values between 0.1 and 0.7. With low γ, the agent focuses on immediate rewards, which are small early in the episode due to initial guesses being far from the target. Higher γ allows the agent to prioritize long-term rewards slowlu refining its guesses over multiple steps.

The sharp increase in rewards near $\gamma = 0.8$ to $\gamma = 1.0$ reflects the agent's ability to use feedback from distance-based observations ("Very Far," "Far," etc.) to adjust actions . At $\gamma = 1.0$, the agent fully values future rewards, maximizing total rewards through strategic planning. This steep increase highlights the importance of long-term planning in optimizing performance in the Hotter-Colder problem.

b) Rewards vs Iterations: The graph shows that rewards increase with iterations for higher γ values (e.g., $\gamma = 0.99$) but remain low for smaller γ values (e.g., $\gamma = 0.1$). For lower γ, the agent focus on immediate rewards resulting in consistently low performance as it fails to refine guesses effectively over multiple iterations.

For higher γ, rewards improve significantly as the agent prioritizes long term planning, refining its guesses and approaching the target. The sharp increase in rewards for $\gamma = 0.99$ reflects effective learning of a policy that use the problem's structure where proximity-based rewards grow quadratically.

Rewards stabilize after 2-3 iterations as the policy converges to optimality in the small state space (V̈ery Far," "Far," etc Once converged, further iterations provide no significant improvement, resulting in a plateau. This behavior aligns with the mechanics of policy iteration in small, structured environments like Hotter-Colder.

*c) **Hypothesis Evaluation**:* The **hypothesis holds true** because Policy Iteration performs efficiently on the Hotter-Colder problem. With higher discount factors (γ), it achieves higher rewards by prioritizing long-term planning. At $\gamma = 0.99$, the algorithm leverages structured feedback from the small state space to refine policies and converge quickly.

Policy Iteration alternates between evaluation and improvement, ensuring stable convergence even with a large continuous action space. The sharp rise in rewards near $\gamma = 1.0$ highlights the importance of long-term planning, aligning with the quadratic reward function that rewards proximity to the target.

The fast stabilization of rewards after 2-3 iterations confirms the efficiency of Policy Iteration in small, structured problems like Hotter-Colder. T

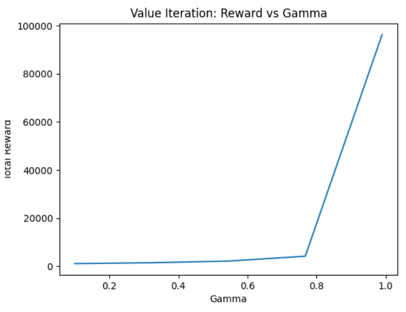
B. Value Iteration

a) Discount/Gamma vs Rewards: As shown in Fig 5.a, total rewards increase sharply as γ approaches 1.0, while remaining low and flat for smaller γ values (e.g., $\gamma < 0.7$). At low γ, the agent focuses on immediate reward, failing to refine guesses over multiple steps which leading to suboptimal policies and low cumulative rewards.

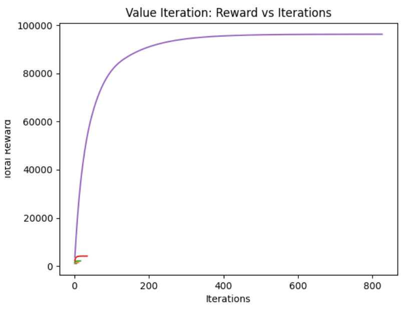
For higher γ values (e.g., $\gamma = 0.7675, 0.99$), rewards increase significantly as the agent prioritizes long-term planning. The agent effectively uses proximity-based feedback to refine its guesses , with a sharp rise in rewards for $\gamma = 0.99$ reflecting the benefits of long-term strategies.

Rewards stabilize after 2-3 iterations due to the small state space, where the value iteration algorithm quickly converges to the optimal policy. The plateau indicates no further improvements after convergence, showing the efficiency of long-term planning with high γ values. the flat curve for low γ demonstrates the limitations of short-sighted policies.

b) Rewards vs Iterations: It can be seen Fig 5.b, rewards increase rapidly in the early iterations for all γ values and plateau after convergence, especially for higher γ values (e.g., $\gamma = 0.99$). This happens because value iteration propagates rewards through the state space, quickly improving policies



(a) Rewards vs Gamma



(b) Rewards vs Iteration

Fig. 5: Value Iteration on Hotter Colder

in the initial iterations. For higher γ, the agent incorporates long-term rewards, enabling it to refine guesses and achieve significantly higher rewards before stabilizing.

For lower γ values (e.g., $\gamma = 0.1, 0.3225$), rewards remain consistently low as the agent focuses only on immediate rewards. This short-sighted approach prevents effective refinement of guesses over time, leading to suboptimal policies and poor performance.

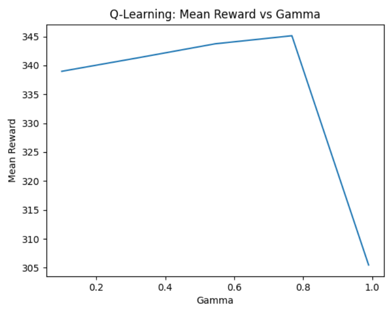
The small state space (4) allows value iteration to converge quickly, while the proximity-based quadratic reward function increases differences between high and low γ values. High γ supports long-term planning, maximizing rewards, while low γ limits the agent's ability to leverage the problem's reward structure.

*c) **Hypothesis Evaluation**:* The **hypothesis holds true** because Value Iteration converges efficiently on the Hotter-Colder problem. Higher discount factors (γ) result in significantly better rewards by enabling long-term planning. At $\gamma = 0.99$, the algorithm uses structured feedback to refine guesses , using the quadratic reward structure where being closer to the target is rewarded more.

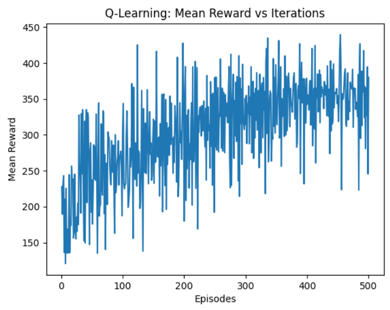
Value Iteration direct updates with the Bellman equation allow quick propagation of values across the small state space , ensuring fast convergence despite the large action space. The sharp rise in rewards near $\gamma = 1.0$ highlights the importance of future rewards in optimizing performance.

Rewards stabilize after a few iterations as the small state space and deterministic transitions ensure quick convergence.

C. Q-learning



(a) Mean Reward vs. Gamma



(b) Mean Reward vs. Iterations

Fig. 6: Q-Learning on Hotter Colder

a) Mean Reward vs. Gamma: It can be seen from Fig 6.a, mean rewards increase as γ rises from low values (e.g., $\gamma = 0.2$) to moderate values (e.g., $\gamma = 0.8$), but drop sharply for higher γ (e.g., $\gamma = 1.0$). This shopws how γ influences Q-learning balance between immediate and future rewards.

For lower γ values ($0.2 \leq \gamma \leq 0.4$), the agent prioritizes immediate rewards, leading to slow improvements as it fails to refine guesses over multiple steps. Moderate γ values ($0.4 \leq \gamma \leq 0.8$) yield the highest rewards, as the agent balances immediate and long-term planning, systematically refining its guesses using feedback from the proximitybased state space.

At high γ values ($\gamma > 0.8$), rewards drop sharply as the agent over prioritizes future rewards, disregarding immediate gains and delaying effective actions. The small state space combined with the continuous action space also limits the agent ability to distinguish differences in action values, resulting in suboptimal policies.

This highlights the importance of selecting a moderate γ to balance immediate and long-term rewards, ensuring optimal Q-learning performance in the Hotter-Colder problem.

b) Mean Reward vs. Iterations: As shown in Fig 6.b, the mean reward fluctuates in early episodes but steadily increases as training progresses. Early variability occurs because the agent explores the environment using strategies like ϵ-greedy, often taking suboptimal actions in the large continuous action space. This results in guesses that are far from the target and low rewards.

As episodes progress, the agent begins exploiting its learned policy while continuing to explore. The small state space provides structured feedback, allowing the agent to refine its value estimates and improve guesses. The proximity-based quadratic reward function further give prefernce better actions, driving a steady increase in rewards.

c) Mean Time vs. Iterations: It was found the mean time (remains constant across all episodes. This is due to the fixed maximum steps per episode in the Hotter-Colder problem, which the agent consistently uses as it has not yet converged to a policy that solves the problem in fewer steps.

Early in training, the ϵ-greedy exploration strategy leads the agent to take suboptimal actions often failing to approach the target efficiently. Even as training progresses the large continuous action space poses challenges for refining guesses, preventing a noticeable reduction in time per episode.

Additionally, the small state space simplifies learning but limits distinctions between states, further slowing the agent's ability to improve efficiency. The constant mean time reflects the environment's constraints and the challenges of balancing exploration and exploitation in Q-learning.

d) ***Hypothesis Evaluation****:* The **hypothesis holds true** because Q-Learning takes the longest to converge on the Hotter-Colder problem, as expected. With moderate γ values ($0.4 \leq \gamma \leq 0.8$), the agent balances immediate and long-term planning, achieving higher rewards by useign the proximity-based feedback of the small state space

Q-Learning's reliance on exploration slows convergence, especially early on, as it spends time sampling the large continuous action space, leading to high reward variability. Over time, the agent transitions to exploitation, refining guesses based on structured feedback, which improves performance despite the complexity of the action space.

The drop in rewards for high γ values ($\gamma > 0.8$) highlights the challenges of overvaluing future rewards, leading to very conservative strategies that neglect immediate improvements..

D. Best Policy

In policy iteration the same action (14) is used consistently across all states. This shows the deterministic nature of policy iteration, which evaluates and improves the policy until convergence. In value iterations also the same action (16) is used across all states. Similar toPI , the uniformity in the actions demonstrates that the algorithm efficiently propagates value updates across the small state space using the Bellman equation. The results from Policy Iteration and Value Iteration are nearly identical, highlighting their shared reliance on the transition dynamics of the problem. However The best policy derived from Q-Learning for theproblem contains only a few non-zero values, showiung that the algorithm has identified a limited set of meaningful actions. Specifically, the policy has non-zero values for the 2nd, 3rd, 4th, and 5th iterations, with corresponding actions of 25, 20, 22, and 20,. These non-zero actions suggest that Q-Learning was able to identify optimal or near-optimal responses for these specific states. The rest is just zeros, indicating the absence of learning actions for those states or a failure to explore. This shows a limitation in Q Learning: while Q-Learning was able to refine a small subset of actions, it struggled to converge properluy, leaving most of the policy unrefined or defaulting to zero, this could be because for not enough iterations, poor tuning or because of the continuous action space of hotter colder.

V. WallClock Time

In Forest Management, Value Iteration is the fastest - completing in just 0.36 seconds,this is because of to its direct updates using the Bellman equation, which allows quick propagation of values across the state space. Policy Iteration takes significantly longer at 1.57 seconds, as it involves solving systems of equations for policy evaluation, making each iteration computationally heavier. Q-Learning, as expected, is the slowest due to its reliance on sampling transitions and extensive exploration, with runtimes exceeding 4,735 seconds for 10 million iterations. This huge difference arises because Policy Iteration and Value Iteration use the structured transition dynamics of the problem, while Q-Learning being model-free, requires far more episodes and computational resources to approximate optimal policies.

In Hotter Colder, Policy Iteration is the fastest completing in 0.04 seconds this is due to its ability to leverage the small state space (4 states) and structured nature of the problem for rapid convergence. Value Iteration takes around 0.70 seconds it slower because it requires iterative updates to approximate the value function until convergence, which is probably taking more iterations than Policy Iteration. QLearning is the slowest, requiring 5 minutes, as it is a model-free algorithm that relies on extensive exploration and sampling to estimate Q-values. This involves more computational effort, particularly in the presence of a large continuous action space and the need for extensive iterations to refine the policy.

Conclusion

For the Forest Management problem, VI was the fastest and achieved optimal rewards, followed closely by PI, which took slightly longer per iteration due to solving equations. Q-Learning was the slowest, requiring significantly more time to converge.

In the Hotter-Colder problem, Policy Iteration was the fastest, leveraging the small state space for rapid convergence. VI performed well but required more iterations, while Q-Learning struggled with the large continuous action space and took the longest to converge.

References

[1] *Library Reference.* Hiive Mdptoolbox. https://github.com/hiive/hiivemdptoolbox.

[2] *GPT 4o* Code Generation and understand topics better

[3] T. M. Mitchell, Machine Learning, Indian Edition. New York, NY, USA: McGraw-Hill, 1997.

[4] OpenAi Gym https://github.com/openai/gym/blob/v0.20.0/gym/envs/toy_text/hotter_colder.py

Overleaf Link: https://www.overleaf.com/read/kyjtpkttzkzpb98d3d