Markov Decision Processes

Jiaqi Yin (jyin90)

I. Introduction

Markov Decision Processes (MDPs) provide a mathematical framework for modeling decision-making processes that involve interactions between an environment and a decision-maker. An MDP is formally defined by a tuple (S, A, P, R, γ), where S represents the state space, A the action space, P the transition probabilities, R the reward function, and γ the discount factor. This project explores two distinct MDP environments from OpenAI Gym: Blackjack and CartPole. The investigation focuses on discrete state spaces to enable direct comparison of different algorithms: Value Iteration (VI), Policy Iteration (PI), Q-learning, and SARSA. The `bettermdptools` toolkit facilitates the implementation of these algorithms across both environments.

A. Blackjack

Blackjack, a widely popular card game, serves as my first MDP environment. The objective is to beat the dealer by obtaining a higher card value without exceeding 21. The environment features a binary action space (hit or stand) and 290 observation states, with rewards of +1 for winning, 0 for drawing, and -1 for losing. This environment proves particularly interesting due to its highly stochastic nature despite the finite deck of 52 cards. The probability distribution of the next card changes dynamically based on the current game state, creating a complex decision space. Furthermore, the known transition matrix makes it an ideal candidate for evaluating model-based methods like value iteration and policy iteration. I **hypothesize**

- The model-based methods (Value Iteration and Policy Iteration) will converge faster than model-free methods (Q-learning and SARSA) due to their access to complete model information (transition probabilities and rewards).
- PI will have fewer iterations to converge than VI since each PI iteration performs a complete policy evaluation.
- The optimal policies derived from model-based methods will be more consistent than those from model-free methods, given the stochastic nature of the Blackjack environment.
- SARSA will be more conservative than Q-learning, leading to Slightly lower average rewards.
- Higher discount factors ($\gamma > 0.9$) will be crucial for both VI and PI due to the delayed nature of rewards in blackjack
- For Q-learning and SARSA, higher initial learning rates (α) combined with gradual decay will perform better than constant low learning rates due to the need to quickly adapt to the stochastic environment
- Higher exploration rates ($epsilon$) will be more important in this environment due to the large state space and need to explore different card scenarios

B. CartPole

The CartPole environment presents a classical control problem where a pole attached to a cart must be balanced upright through horizontal cart movements. While originally designed with continuous state variables (cart position, cart velocity, pole angle, and pole angular velocity), I discretize these states using `bettermdptools` to enable consistent comparison across all algorithms. This discretization process itself presents an interesting area of study, particularly in understanding how different state space granularities affect learning performance. The environment's immediate reward structure and state-dependent action requirements make it particularly suitable for model-free methods. I **hypothesize** that

- Finer discretization (more bins) will lead to policies achieving higher average rewards, but will require more iterations to converge due to the larger state space. This is because finer discretization allows for more precise control actions at different pole angles and cart positions.
- Model-free methods will demonstrate superior performance due to their ability to learn directly from experience.
- Higher discount factors ($\gamma > 0.9$) will will prove crucial given the environment's failure state characteristics.
- VI and PI will show similar final policy quality but different convergence patterns due to their different update mechanisms
- Q-learning will achieve higher peak performance than SARSA but with more variance in learning.
- The optimal initial learning rate will need to decrease as the number of discretization bins increases to maintain stability

II. Methods

This study explores both model-based (VI, P), and model-free (Q-learning, SARSA) reinforcement learning algorithms for solving MDPs.

Model-based algorithm (VI and PI) require complete knowledge of the environment's dynamics, specifically the transition probabilities $P(s'|s, a)$ and reward function $R(s, a)$. This information is provided through the environment's P matrix in my implementations. These methods can perform systematic sweeps through the state space to compute optimal policies without actual environment interaction.

Model-free approaches like Q-learning and SARSA take a different path - they learn by doing, accumulating wisdom

through direct interaction with their environment. Rather than needing a pre-built map of actions and consequences, these algorithms discover the lay of the land through experimentation, much like an explorer charting unknown territory. While this hands-on learning style offers greater adaptability, it often requires more time to find optimal solutions. This trade-off becomes especially apparent in scenarios such as CartPole, where the smooth, continuous nature of states makes it difficult to create precise mathematical models of the system's behavior.

VI iteratively updates state values by selecting actions that maximize future rewards. The algorithm uses the Bellman optimality equation to update each state's value:

$$V(s) \leftarrow max_a[R(s,a) + \gamma \sum P(s'|s,a)V(s')]$$

where γ is the discount factor.

PI operates in two distinct phases: policy evaluation and policy improvement. During evaluation, it computes state values for the current policy until convergence, then improves the policy by selecting actions that maximize these values. While PI generally requires fewer iterations than VI, each iteration is more computationally intensive due to the complete policy evaluation step. My PI implementation shares the same parameter space as VI for γ and θ, enabling direct comparison of their convergence properties.

Q-learning updates state-action values (Q-values) based on the maximum future Q-value, independent of the policy being followed. The update rule is:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

where α is the learning rate. My implementation includes several key parameters: initial learning rate γ (0.2 to 1.0), learning rate decay ratio, initial exploration rate ϵ (0.2 to 1.0), exploration decay ratio, and discount factor γ.

SARSA differs from Q-learning by using the actual next action chosen by the policy rather than the maximum Q-value for updates:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R + \gamma Q(s',a') - Q(s,a)]$$

where a' is the action actually taken in state s'. my implementation uses the same parameter ranges as Q-learning, allowing comparison between these on-policy versus off-policy approaches.

A. Experimental Setup

1) Blackjack Environment

The experiments were configured as follows:

- Natural discrete state space with 290 states
- Two actions (hit or stand)
- Value/Policy Iteration parameters:
  - Discount factors γ: [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99, 0.999, 1.0]
  - Convergence thresholds θ: [0.1, 0.01, 0.001, 1e-6, 1e-8, 1e-10, 1e-12, 1e-14, 1e-16]
- Q-learning/SARSA parameters:
  - Discount factors γ: same as VI/PI
  - Initial learning rates α: [0.2, 0.4, 0.6, 0.8, 1.0]
  - Initial exploration rates ϵ: [0.2, 0.4, 0.6, 0.8, 1.0]
  - Number of episodes: 10000
- Testing iterations: 200 episodes for all methods

2) CartPole Environment

The following configurations were used:

- Discretized continuous state space. Position bins: [2, 5, 10, 20, 50]
- Two actions (left or right)
- Value/Policy Iteration parameters:
  - Discount factors γ: [0.2, 0.4, 0.6, 0.8, 0.9, 0.99]
  - Convergence thresholds θ: [0.1, 0.01, 0.001, 1e-6]
  - Number of iterations: 10000
- Q-learning/SARSA parameters:
  - Discount factors γ: same as VI/PI
  - Initial learning rates α: [0.2, 0.4, 0.6, 0.8, 1.0]
  - Initial exploration rates ϵ: [0.2, 0.4, 0.6, 0.8, 1.0]
  - Number of episodes: 10000
- Testing iterations: 200 episodes for all methods

For both experiments, random seed fixed for reproducibility. The best parameters were selected based on the average reward over 200 episodes.

Performance metrics tracked:

- Convergence iteration counts
- Mean and standard deviation of rewards
- Runtime performance
- Value function statistics (mean, max, delta values)

III. Results and Discussion

A. Blackjack

In this experiment, I have compared the performance of VI, PI, Q-learning, and SARSA in the Blackjack environment with various parameters. When compared VI with PI, VI takes more iteration to converge than PI when having the same discount factor and convergence threshold, seen in Fig 1 and Fig. 4a. This supports my hypothesis. The runtime of PI turns to higher than VI, as showed in Fig. 2. PI has less iteration but longer runtime, which can be attributed to PI's requirement to solve a complete system of linear equations during each policy evaluation step, making individual iterations more computationally intensive despite requiring fewer total iterations.

When studying the average reward for PI and VI, it surprises me because the average reward and reward standard deviation are the same when having the same discount factors with any convergence threshold. It also surprises me that the highest mean reward not comes from the higher discount factors, it reaches the highest reward when discount factor between 0.3 and 0.7 in Fig. 3. Blackjack has only 230 states. When one environment has the known transition matrix, the model-based methods will converge to the same policy and value function. The medium discount factors performing better can be explained by the short-term nature of Blackjack decisions - higher discount factors might overemphasize future rewards, leading to overly conservative play, while lower discount factors might make the agent too shortsighted. The optimal

balance appears to be in the middle range, where the agent appropriately weighs both immediate and future consequences of its actions.

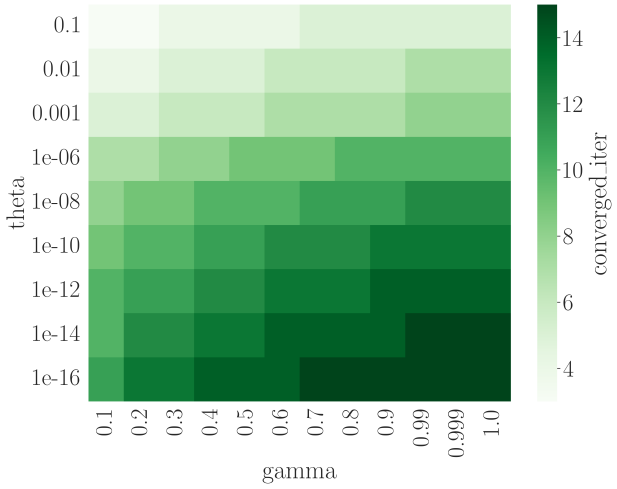
When studying the discount factor (γ) and convergence threshold (θ) impact on the PI and VI converge iterations and runtime, it shows that the higher discount factor and lower convergence rate have higher convergence iteration and runtime (in Fig. 1 and 2). This occurs because higher discount factors extend the temporal horizon over which the algorithm must consider future rewards, requiring more iterations to propagate value changes through the state space. Similarly, lower convergence thresholds demand more precise value estimates, necessitating additional iterations to achieve the required accuracy.

When compared model-free methods metric with VI/PI, both Q-learning and SARSA have higher runtime and rewards, seen in Fig 6 and 5. It is interesting that the reward from model-free methods is higher than model-based methods when transition matrix is known. This unexpected result might be due to the exploration-based nature of model-free methods allowing them to discover profitable state-action pairs that model-based methods might undervalue due to the averaged nature of their probability transitions. It is also possible that the transition matrix provided by the toolkit `bettermdptools` is not entirely accurate, leading to suboptimal performance in model-based methods. When visualizing the optimal policy in Fig. 7, it shows that the optimal policy derived from VI and PI is the same, while Q-learning and SARSA have different optimal policies than VI and PI. It supports the observations that model-based and model-free methods result in different reward outcomes.

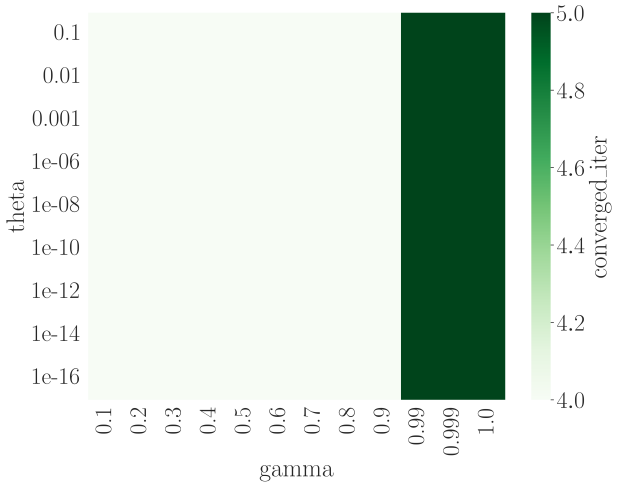
Both Q-learning and SARSA have converged within 10,000 episodes seen in Fig. 4b and **??**, while Q-learning requires more episodes than SARSA. This difference arises because Q-learning's off-policy nature requires more exploration to learn optimal Q-values, while SARSA's on-policy learning more quickly converges to a stable policy. From Fig. 6 and 5, when having the same initial learning rate, it takes more runtime and yields higher reward when having a higher initial exploration rate. This relationship emerges because higher exploration rates allow the agents to discover more diverse state-action pairs, leading to better policy optimization despite the increased computational cost. For both Q-learning and SARSA, higher learning rates combined with high exploration rates initially show faster learning but may lead to less stable policies, while lower learning rates with moderate exploration rates tend to produce more robust policies over time.

The results support my hypothesis that SARSA is more conservative than Q-learning, leading to slightly lower average rewards. This behavior stems from SARSA's on-policy nature, where it learns the value of the actual policy being followed, including exploratory moves, rather than the optimal policy, making it more risk-averse in its final policy.

In summary, this investigation reveals interesting tradeoffs between model-based and model-free approaches in the Blackjack environment. While model-based methods show faster convergence and consistent policies, model-free methods surprisingly achieved higher rewards through their exploration-based learning. The results also highlight the critical role of parameter selection, particularly the discount factor in model-based methods and the exploration-exploitation balance in model-free methods.



(a) VI iterations to convergence

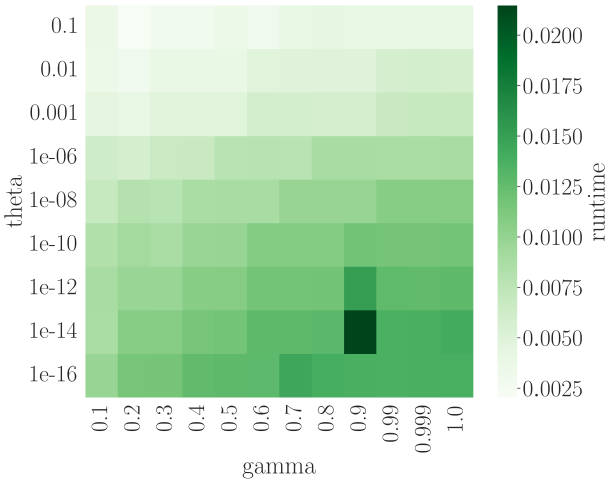


(b) PI iterations to convergence

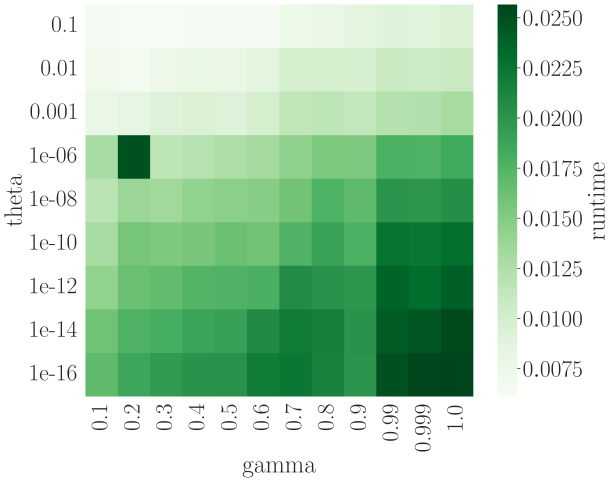
Fig. 1. Blackjack: Convergence iterations for VI and PI

B. CartPole

The CartPole experiments explored both algorithm performance comparisons and the impact of state space discretization. The environment's four continuous state variables (cart position, cart velocity, pole angle, and pole angular velocity) were discretized using `bettermdptools`. While maintaining 10 bins for velocity, pole angle, and angular velocity, we varied position bins (2, 5, 10, 20, 50), resulting in total state spaces of 4600, 11500, 23000, 46000, and 115000 states respectively. All algorithms were evaluated over 10000 iterations/episodes to balance computational constraints with learning requirements.

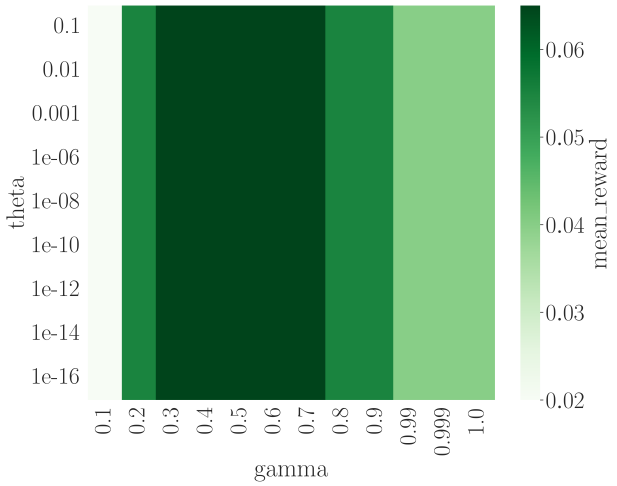


(a) VI runtime

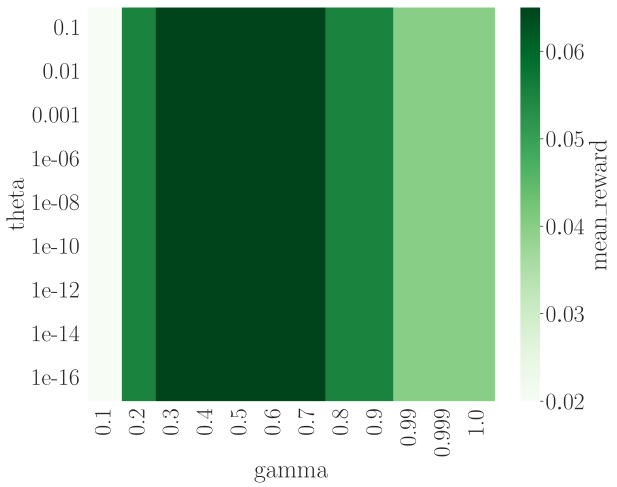


(b) PI runtime

Fig. 2. Blackjack: Runtime performance for VI and PI



(a) VI average reward



(b) PI average reward

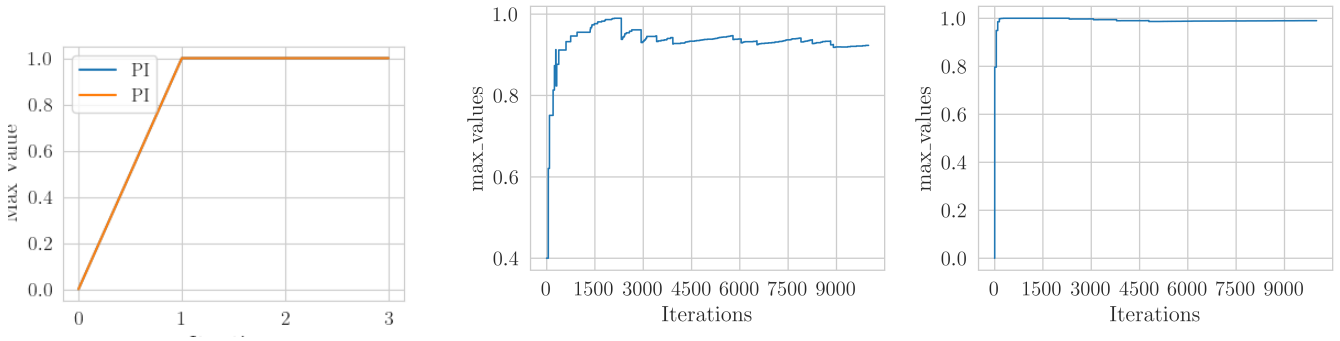
Fig. 3. Blackjack: Average reward for VI and PI

Model-free methods (Q-learning and SARSA) exhibited significantly longer runtime compared to model-based methods (VI and PI) across all state space sizes. Runtime increased with discount factor, correlating positively with mean reward (Fig. 8b). This relationship emerges from the deeper backup of future rewards with higher discount factors, requiring more computational effort but enabling better long-term decision making. The superior performance of model-free methods over VI/PI supports my hypothesis, likely due to their ability to learn actual environment dynamics rather than relying on approximate transition models.

Contrary to my initial hypothesis, finer discretization did not consistently lead to higher rewards. Smaller state spaces (4600 and 11500 states) achieved maximum rewards ($\sim$ 500) with γ ¿ 0.9, while larger state spaces showed incomplete convergence within the episode limit. Fig. 9 demonstrates this through increasing maximum Q-values as episode increases, suggesting potential for improved performance with extended training. This reveals a fundamental tradeoff between state space granularity and learning efficiency - while finer discretization theoretically allows more precise control, it also increases the exploration burden on learning algorithms.

For model-free methods, higher learning rates and exploration rates generally led to increased runtime and rewards (Fig. 10-12). This correlation strengthens with larger state spaces, as more aggressive learning and exploration become necessary to effectively cover the expanded state space. However, an interesting pattern emerged where larger state spaces required higher initial learning rates but lower exploration rates. This counterintuitive relationship suggests that with larger state spaces, focused learning in discovered valuable states becomes more critical than broad exploration.

SARSA unexpectedly outperformed Q-learning in average rewards under identical parameters, contrary to my hypothesis. This might be attributed to SARSA's on-policy nature providing more stable learning in this continuous control task, where consistent behavior could be more valuable than the potentially aggressive optimization of Q-learning.

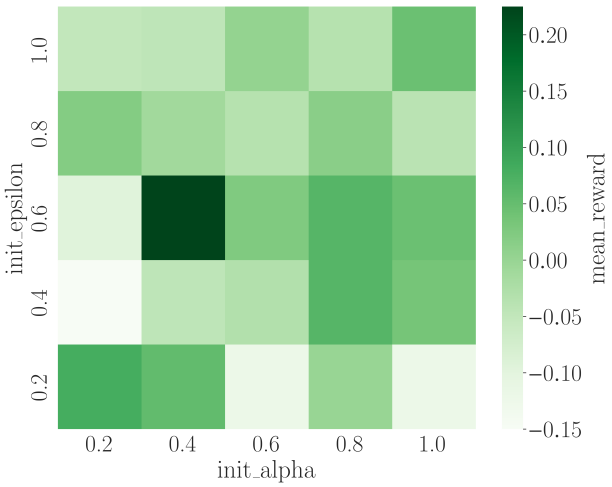


(a) Blackjack: Maximum value iteration comparison between VI and PI with $\gamma = 0.3$ and θ=0.1

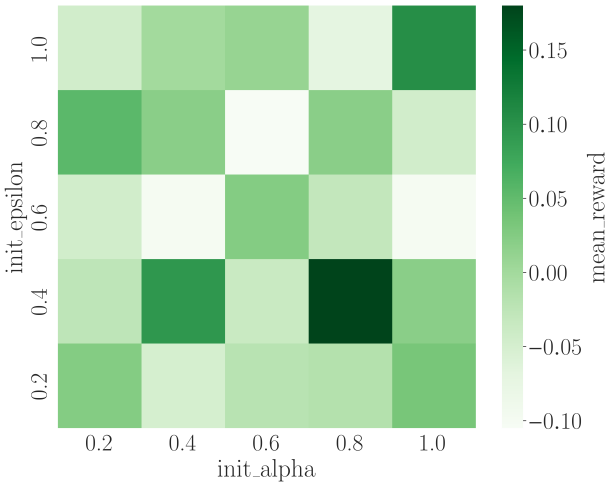
(b) Blackjack: Maximum Q-value comparison changes over episodes for Q-learning with $\gamma = 0.99$, $\alpha = 0.4$, $\epsilon = 0.6$

(c) Blackjack: Maximum Q-value comparison changes over episodes for SARSA with $\gamma = 0.7$, $\alpha = 0.8$, $\epsilon = 0.4$

Fig. 4. Blackjack: Maximum value/Q-value comparison among VI, PI, Q-learning, and SARSA

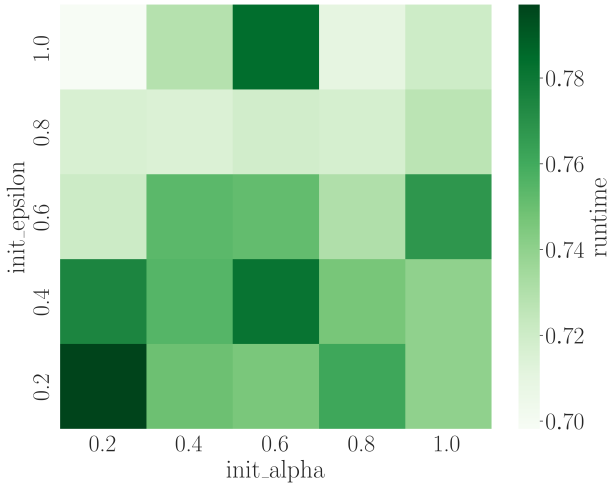


(a) Q-learning average reward with respect to initial α and initial ϵ when setting $\gamma = 0.99$

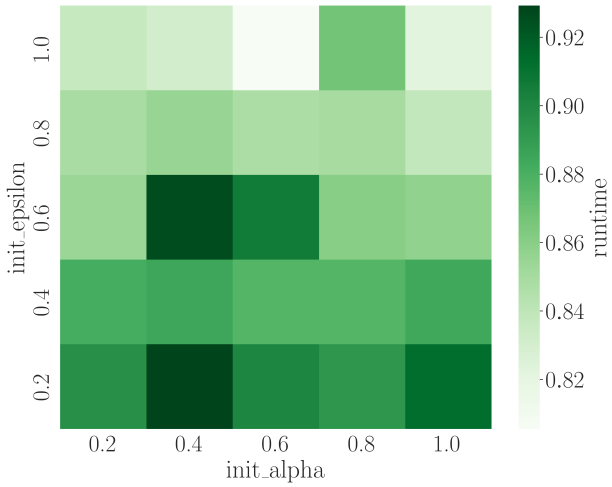


(b) SARSA average reward with respect to initial α and initial ϵ when setting $\gamma = 0.7$

Fig. 5. Blackjack: Average reward for Q-learning and SARSA with respect to initial α and ϵ

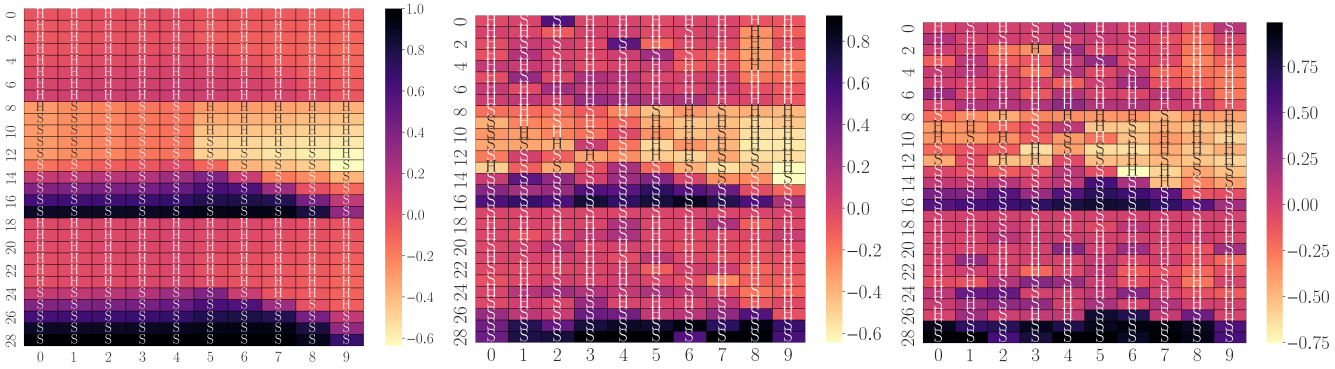


(a) Q-learning runtime with respect to initial α and initial ϵ when setting $\gamma = 0.99$



(b) SARSA runtime with respect to initial α and initial ϵ when setting $\gamma = 0.7$

Fig. 6. Blackjack: Runtime for Q-learning and SARSA with respect to initial α and ϵ



(a) Blackjack: Optimal policy derived from VI/PI with $\gamma = 0.3$ and θ=0.1

(b) Blackjack: Optimal policy derived from Q-learning with $\gamma = 0.99$, $\alpha = 0.4$, $\epsilon = 0.6$

(c) Blackjack: Optimal policy derived from SARSA with $\gamma = 0.7$, $\alpha = 0.8$, $\epsilon = 0.4$

Fig. 7. Blackjack: Optimal policy comparison among VI, PI, Q-learning, and SARSA

A striking observation was the identical average reward (9.4) across all parameter combinations for VI/PI. This uniformity stems from the toolkit's simplified transition model, which assigns probability 1 to deterministic state transitions. While this highlights a limitation in my current implementation, it also emphasizes a crucial insight: in real-world control problems, accurate transition models are extremely difficult to specify without detailed physical modeling. This reinforces the practical advantage of model-free methods in such domains, where learning from experience can capture complex dynamics that might be difficult to model explicitly.

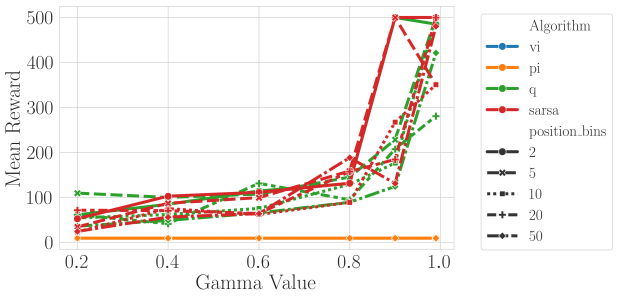
The relationship between learning parameters and state space size revealed interesting patterns. For larger state spaces: Higher initial learning rates became necessary to make meaningful updates given limited episodes; Lower exploration rates proved more effective, suggesting the importance of exploiting discovered good behaviors; Discount factor impact became more pronounced, with higher values (> 0.9) consistently leading to better performance

These results demonstrate the complex interplay between state space discretization, algorithm choice, and parameter selection in continuous control problems. While model-free methods showed superior performance, their effectiveness heavily depends on appropriate parameter tuning and sufficient training time, especially with larger state spaces. The findings highlight the practical challenges of applying model-based methods to continuous control problems and the importance of careful discretization choices in balancing representation precision with learning efficiency.

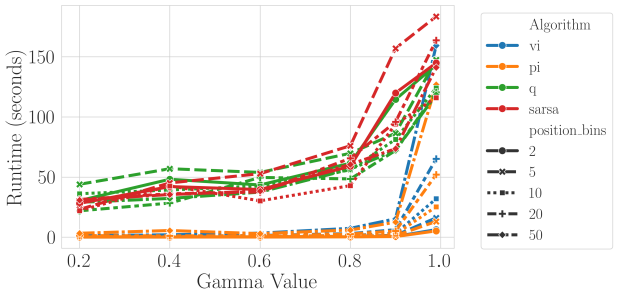
IV. Conclusion

This study's comprehensive investigation of MDPs through both model-based and model-free approaches across two distinct environments - Blackjack and CartPole - has yielded several significant insights, both supporting and challenging my initial hypotheses.

In the Blackjack environment, my hypothesis about PI requiring fewer iterations than VI was confirmed, though interestingly, this came at the cost of higher computational runtime



(a) CartPole: Average reward for VI, PI, Q-learning, and SARSA with respect to discount factor γ and number of position bins



(b) CartPole: Runtime for VI, PI, Q-learning, and SARSA with respect to discount factor γ and number of position bins

Fig. 8. CartPole: Average reward and runtime performance

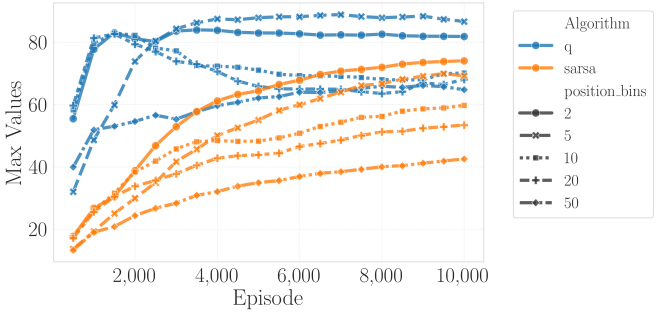
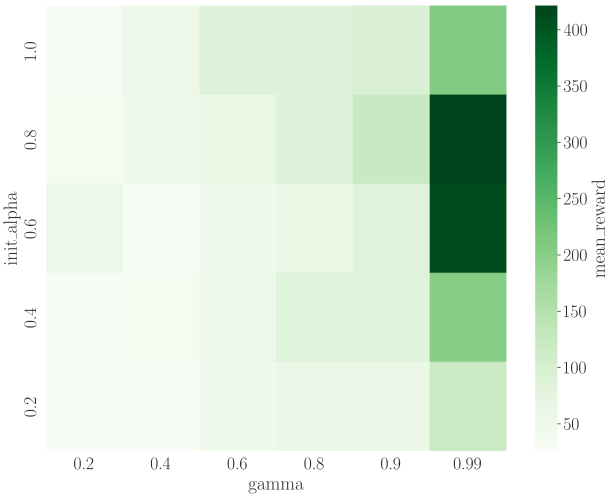
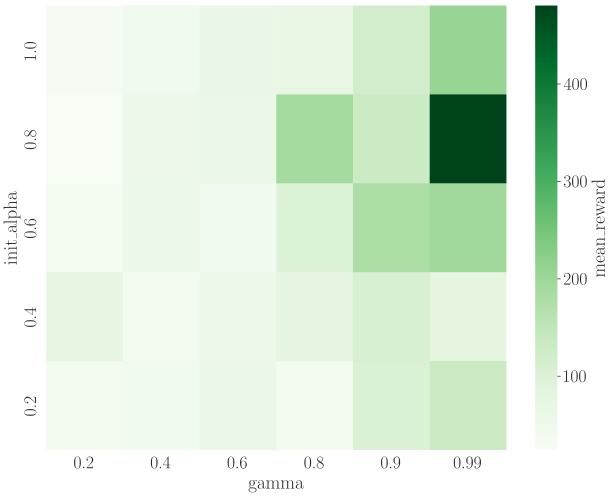


Fig. 9. CartPole: Maximum Q-value comparison changes over episodes for Q-learning and SARSA with $\gamma = 0.99$, $\alpha = 0.6$, $\epsilon = 0.8$

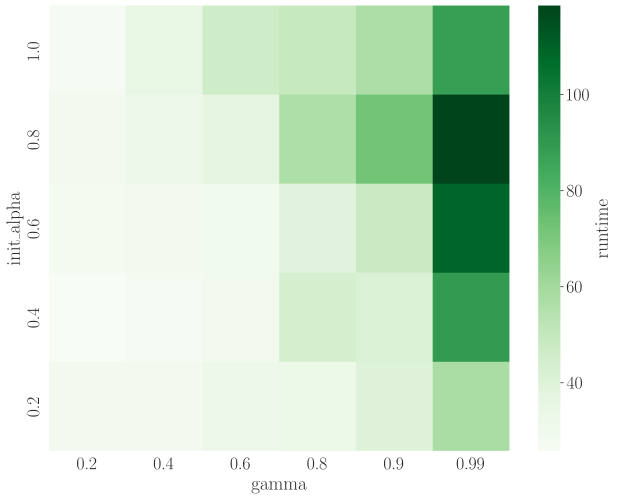


(a) Q-learning average reward with respect to γ, initial α for 50 position bins

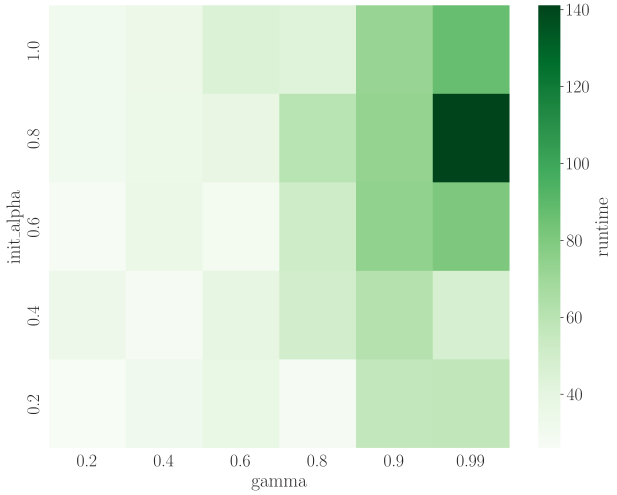


(b) SARSA average reward with respect to γ, initial α for 50 position bins

Fig. 10. CartPole: Average reward for Q-learning and SARSA with respect to γ and initial α



(a) Q-learning runtime with respect to γ, initial α for 20 position bins



(b) SARSA runtime with respect to γ, initial α for 50 position bins

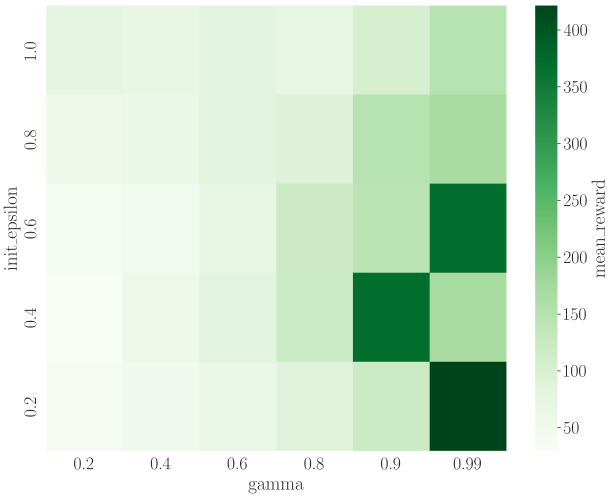
Fig. 11. CartPole: Runtime for Q-learning and SARSA with respect to γ and initial α

due to PI's intensive policy evaluation step. Contrary to my expectations about discount factors, optimal performance was achieved with moderate values ($\gamma = 0.3 - 0.7$) rather than higher ones ($\gamma > 0.9$), suggesting that in Blackjack, balancing immediate and future rewards is more crucial than long-term planning. Perhaps most surprisingly, model-free methods achieved higher rewards than model-based approaches despite the availability of transition probabilities, challenging my initial assumptions about the superiority of model-based methods in environments with known dynamics.

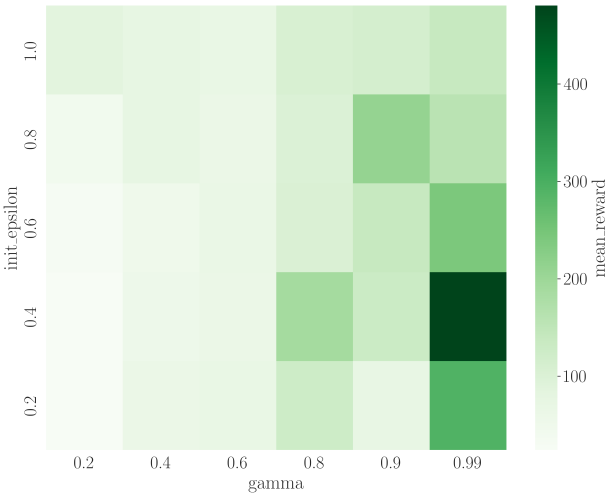
The CartPole experiments revealed different patterns. While my hypothesis about model-free methods demonstrating superior performance was confirmed, my expectation about finer discretization leading to better performance was only partially supported. Smaller state spaces (4,600-11,500 states) achieved optimal performance more consistently than larger ones, highlighting the critical balance between representation precision and learning efficiency. SARSA's unexpected outperformance of Q-learning in this environment challenges conventional wisdom about off-policy methods' superiority in continuous control tasks.

Several significant challenges emerged during this investigation. A primary concern was the potential inaccuracy of transition matrices provided by the `bettermdptools` library, particularly evident in the identical rewards observed across different parameter combinations in CartPole's model-based methods. This limitation underscores the broader challenge of accurately modeling transition dynamics in real-world applications. Additionally, the computational constraints with larger state spaces limited my ability to fully explore convergence properties with finer discretization levels.

Given additional time, several promising avenues for future research emerge. A deeper investigation into dynamic discretization methods could optimize the state-space representation



(a) Q-learning average reward with respect to γ, initial ϵ for 50 position bins



(b) SARSA average reward with respect to γ, initial ϵ for 50 position bins

Fig. 12. CartPole: Average reward for Q-learning and SARSA with respect to γ and initial ϵ

during learning. Implementing more sophisticated exploration strategies, particularly for larger state spaces, could improve learning efficiency. Additionally, examining the impact of different reward structures and incorporating function approximation methods could provide valuable insights for scaling these approaches to more complex environments.

This study contributes to my understanding of the practical considerations in applying different RL algorithms to distinct problem domains, highlighting the importance of careful parameter selection and the sometimes counterintuitive relationship between model knowledge and performance. The findings underscore the need for empirical validation of theoretical assumptions and the value of comparing diverse approaches when tackling real-world reinforcement learning problems.