Reinforcement Learning

Assignment 02

Spring 2024

By

W.H. Sasinda C. Prabhashana-202395458

&

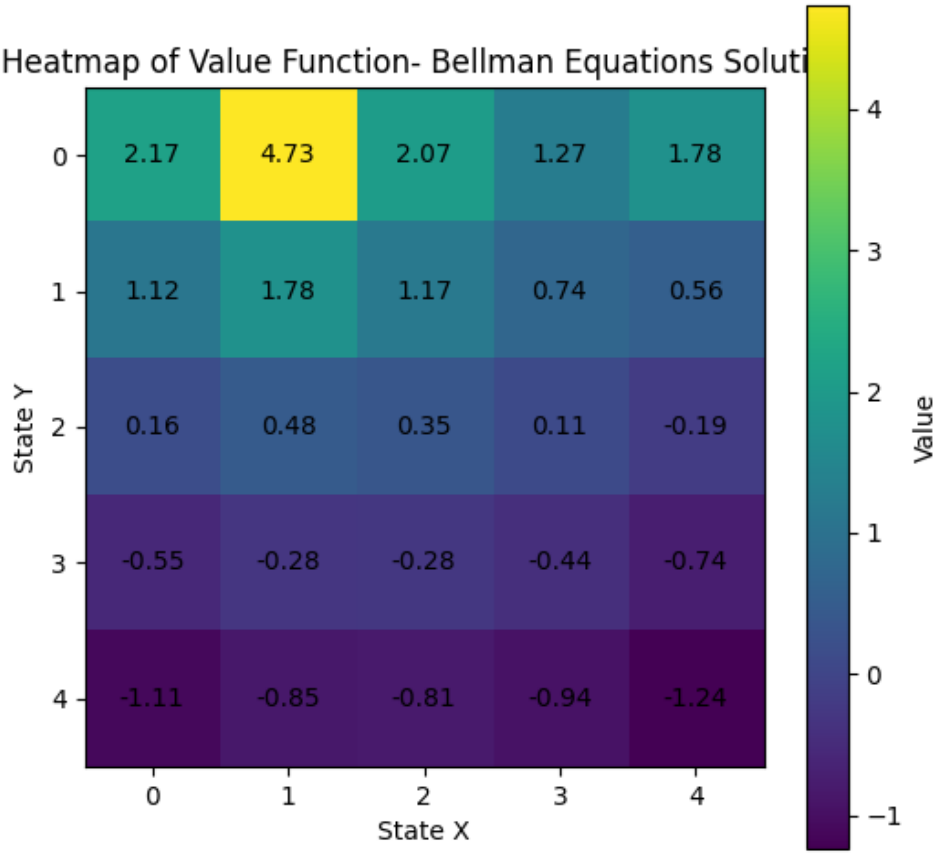
Premisha Premananthan - 202397583

[This document fulfills the requirements for the Reinforcement Learning Module (DSCI-6650-001) at Memorial University in Newfoundland, Canada]

PART 01

1.1) Estimating the value function for each of the states solving the system of Bellman equations explicitly

```
Bellman Equations Solution for Value Function:
[[ 2.17100208  4.7336156   2.07028049  1.26529444  1.77912239]
 [ 1.1180732   1.7821227   1.17409573  0.739174    0.56246548]
 [ 0.16279444  0.47788999  0.35198379  0.11045592 -0.18617038]
 [-0.54699155 -0.28473257 -0.28040463 -0.43990985 -0.7443105 ]
 [-1.10787684 -0.84936779 -0.80799244 -0.93799278 -1.23723244]]
```



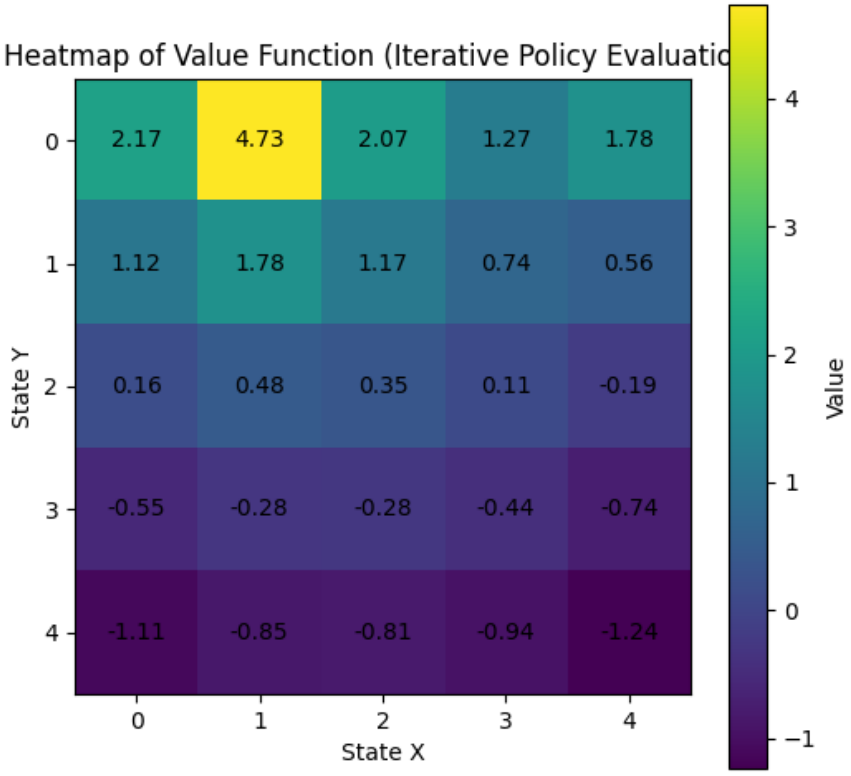
Explanation:

This value function is estimated by solving the system of Bellman equations. By solving this system of equations, we obtain the expected cumulative rewards for each state under the given policy. The policy in this scenario is one where the agent moves to one of the four possible directions (up, down, left, right) with equal probability of 0.25. In this heatmap, each cell's value corresponds to the expected cumulative reward starting from that state. The highest value, approximately 4.73, is observed in the state (0, 1), which

represents the blue square. This high value is primarily due to the immediate reward of 5 that the agent receives upon reaching the blue square. Furthermore, the blue square transitions the agent to the red square (3, 2), allowing the agent to continue accumulating rewards from a relatively advantageous position. The values for other states are comparatively lower, reflecting their lesser immediate rewards and less favorable transition dynamics. The explicit solution of the Bellman equations provides a precise and reliable estimate of the value function. The results highlight the significant impact of the blue square's high reward on the overall value function. The consistency in identifying the blue square as the highest value state confirms the effectiveness of this method in capturing the reward structure and transition dynamics of the grid world environment.

1.2) Estimating the value function for each of the states using iterative policy evaluation

```
Iterative Policy Evaluation for Value Function:
[[ 2.17106167  4.73356968  2.07007691  1.26509157  1.77903674]
 [ 1.11812089  1.78206863  1.17393876  0.73898219  0.56228794]
 [ 0.16288173  0.47788332  0.35188381  0.11029994 -0.18634548]
 [-0.54685011 -0.28468338 -0.28044828 -0.44001921 -0.74445037]
 [-1.10770126 -0.84928557 -0.8080041  -0.93807386 -1.2373476 ]]
```



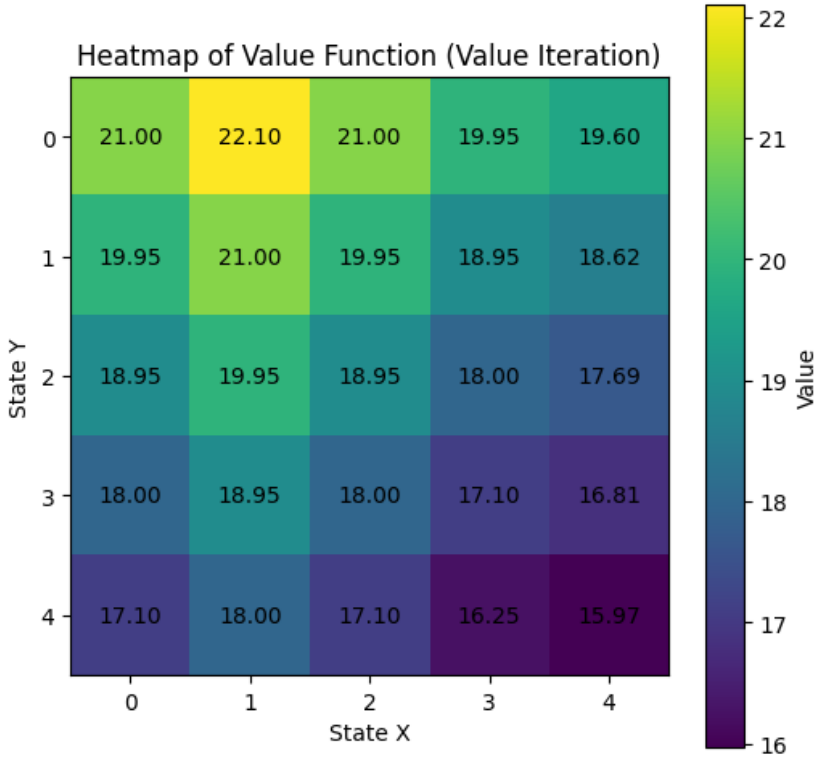
Explanation:

The state (0, 1), which represents the blue square, again stands out with the highest value of approximately 4.73. This consistency with the first method is expected, as the blue square provides

an immediate reward of 5 and transitions the agent to the red square (3, 2), facilitating further accumulation of rewards. Other states have lower values, which reflect their respective reward structures and less advantageous transitions. The iterative policy evaluation method effectively captures the value function by gradually refining the estimates through repeated updates. The high value observed in the blue square (0, 1) underscores its significant immediate reward and strategic transition benefits. This method's results closely match those obtained from explicitly solving the Bellman equations, demonstrating its reliability and accuracy in estimating the value function under the given policy. The iterative nature of this method allows for a more dynamic adjustment of values, ensuring that the estimates reflect the long-term expected rewards accurately.

1.3) Estimating the value function for each of the states using value iteration

```
Value Iteration Solution for Value Function:
[[20.99700533 22.10219208 20.99708247 19.94722835 19.60219208]
 [19.94715507 20.99708247 19.94722835 18.94986693 18.62208247]
 [18.94979731 19.94722835 18.94986693 18.00237358 17.69097835]
 [18.00230745 18.94986693 18.00237358 17.10225491 16.80642943]
 [17.10219208 18.00237358 17.10225491 16.24714216 15.96610796]]
```



Explanation:

In this heatmap, the state (0, 1), corresponding to the blue square, again emerges as the state with the highest value, approximately 22.10. This value is significantly higher than the values observed in the first two methods. This substantial difference is due to the value iteration algorithm's tendency

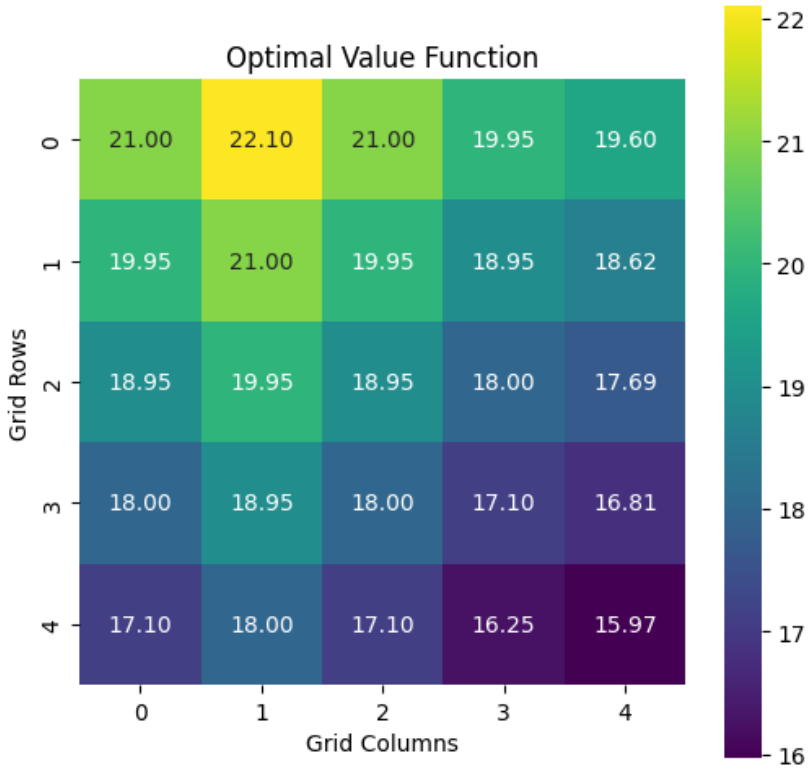
to propagate the highest possible values backward through the states, aggressively optimizing the value function. The blue square's immediate reward of 5 and its favorable transition to the red square (3, 2) significantly boost its value. Other states show higher values compared to the previous methods, reflecting the algorithm's optimization process. The value iteration method provides a highly optimized estimate of the value function by continuously updating it to reflect the maximum expected rewards. The remarkably high value for the blue square (0, 1) highlights the impact of its high immediate reward and strategic transition, confirming its critical role in the grid world environment. The overall higher values across the grid indicate that the value iteration method effectively identifies and propagates optimal value estimates, ensuring that each state's value represents the best possible cumulative reward under an optimal policy.

Does This Surprise?

The consistent identification of the state (0, 1) (the blue square) as the highest value state across all three methods is not surprising. This result is expected due to the grid world's reward structure and transition dynamics. The blue square offers an immediate reward of 5 and transitions the agent to the red square (3, 2), which allows for further accumulation of rewards. This combination makes the blue square particularly valuable. What is more interesting is the magnitude of the values observed in the value iteration method compared to the other two methods. The value iteration method yields a significantly higher value for the blue square (0, 1), approximately 22.10, compared to 4.73 in the other methods. This substantial difference is due to value iteration's optimization approach, which aggressively propagates the maximum possible values backward through the states. By continuously considering the maximum expected reward over all possible actions, value iteration amplifies the value estimates, highlighting the blue square's strategic importance even more.

2.1). Optimal policy for the grid world problem by explicitly solving the Bellman optimality equation

```
Optimal Value Function by Solving Bellman Optimality Equation:
[[20.99734295 22.10246707 20.99734371 19.94747653 19.60246707]
 [19.9474758  20.99734371 19.94747653 18.9501027  18.62234371]
 [18.95010201 19.94747653 18.9501027  18.00259757 17.69122653]
 [18.00259691 18.9501027  18.00259757 17.10246769 16.8066652 ]
 [17.10246707 18.00259757 17.10246769 16.2473443  15.96633194]]
Optimal Policy:
→ ↑ ← ← ↓
↑ ↑ ↑ ↑ ↑
↑ ↑ ↑ ↑ ↑
↑ ↑ ↑ ↑ ↑
↑ ↑ ↑ ↑ ↑
```



Explanation:

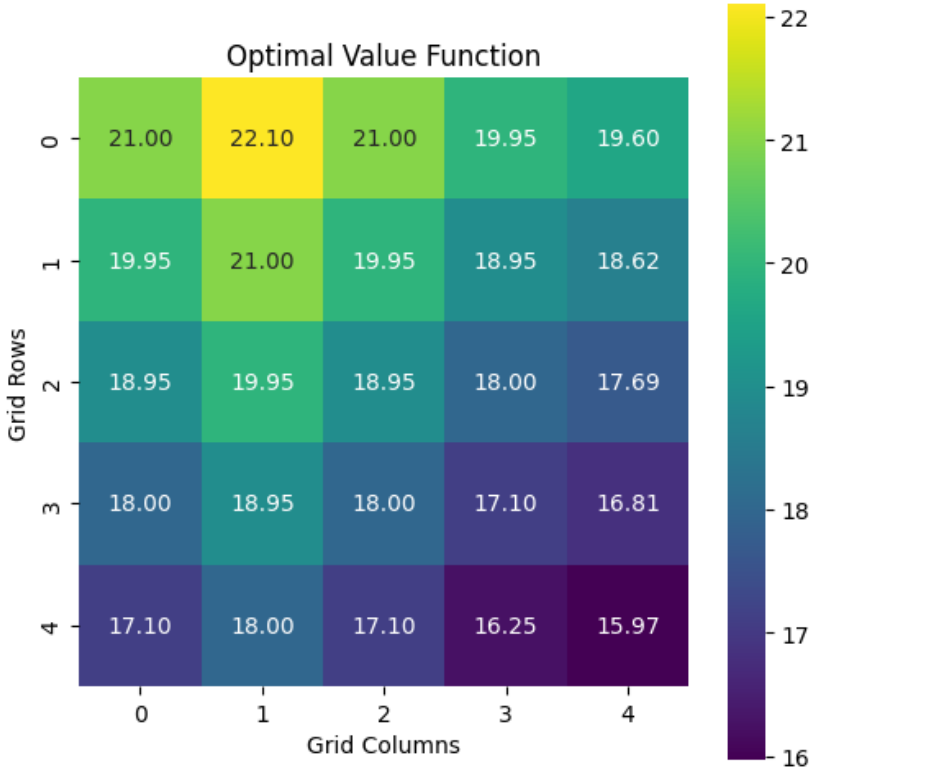
By solving the Bellman optimality equation, we observe a grid where the highest values are concentrated in the top left, particularly in the first two rows. This is expected, given the significant rewards for the blue square at (0,1) and the green square at (0,4), which propagate through the grid. As we move towards the bottom right, the values gradually decline, indicating that states further from these high-reward squares have lower expected cumulative rewards, influenced by the discount

factor. The heatmap clearly illustrates this gradient, with brighter colors highlighting the higher values near the rewarding states and darker shades representing lower values further away.

Initially, the policy recommends moving right from (0,0) towards the blue square at (0,1), then upward, aligning with the transition to (3,2). The first row indicates movements to the left and eventually downward from (0,4). In the subsequent rows, the policy consistently suggests moving upwards, reflecting a strategy where the agent aims to reach higher-value states above. This strategy demonstrates how the value function influences the policy, driving actions that maximize cumulative rewards.

2.2). Optimal policy for the grid world problem by explicitly using policy iteration with iterative policy evaluation

```
Optimal Value Function by Policy Iteration:
[[20.99734629 22.10246979 20.9973463  19.94747898 19.60246979]
 [19.94747897 20.9973463  19.94747898 18.95010503 18.6223463 ]
 [18.95010503 19.94747898 18.95010503 18.00259978 17.69122898]
 [18.00259977 18.95010503 18.00259978 17.10246979 16.80666753]
 [17.10246979 18.00259978 17.10246979 16.2473463  15.96633416]]
Optimal Policy:
→ ↑ ← ← ↑
→ ↑ ↑ ↑ ↑
→ ↑ ↑ ↑ ↑
→ ↑ ↑ ↑ ↑
→ ↑ ↑ ↑ ↑
```



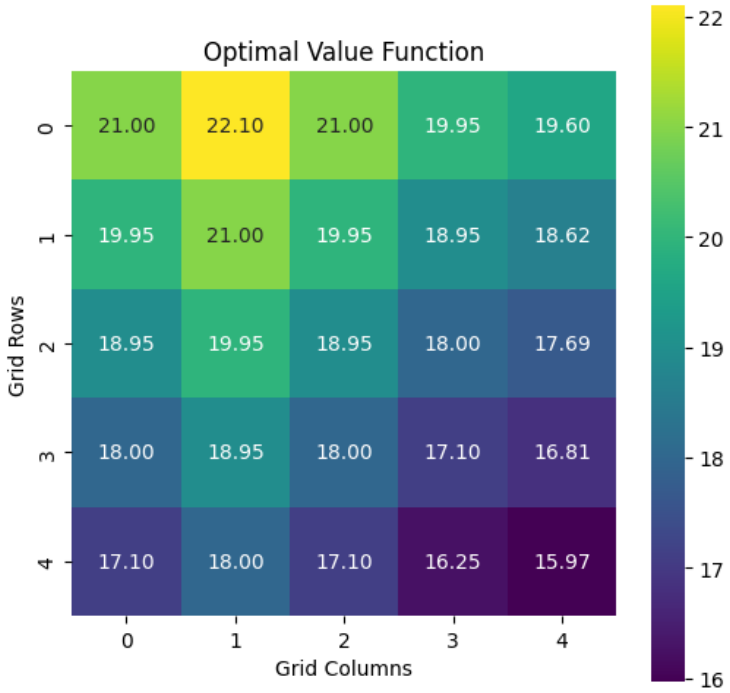
Explanation:

When we use policy iteration with iterative policy evaluation, we can see that the highest values are again concentrated near the top left, especially in the first two rows. This pattern is consistent with the rewards for the blue square at (0,1) and the green square at (0,4), which propagate through the grid. As we observe the values moving towards the bottom right, there is a gradual decrease, indicating lower expected cumulative rewards for states further from these high-reward squares due to the discount factor. The heatmap visualization vividly displays this gradient, with brighter colors near the rewarding states and darker colors as one moves away.

The policy starts by suggesting a rightward move from (0,0) towards the blue square at (0,1), followed by an upward movement to (3,2). Next, the policy indicates leftward movements from (0,2) to (0,4), and finally, an upward move from (0,4). The second row begins with moving right and then consistently suggests upward movements in the subsequent rows. This consistent upward movement strategy in the later rows reflects a simplified decision-making process, aiming to reach higher-value states above. The visual representation of the policy effectively demonstrates this strategy, guiding the agent towards high-reward states and optimizing its path.

2.3). Optimal policy for the grid world problem by explicitly using policy improvement with value iteration

```
Optimal Value Function by Value Iteration:
[[20.99734371 22.10246769 20.9973443  19.94747709 19.60246769]
 [19.94747653 20.9973443  19.94747709 18.95010323 18.6223443 ]
 [18.9501027  19.94747709 18.95010323 18.00259807 17.69122709]
 [18.00259757 18.95010323 18.00259807 17.10246817 16.80666573]
 [17.10246769 18.00259807 17.10246817 16.24734476 15.96633245]]
Optimal Policy:
→ ↑ ← ← ↑
→ ↑ ↑ ↑ ↑
→ ↑ ↑ ↑ ↑
→ ↑ ↑ ↑ ↑
→ ↑ ↑ ↑ ↑
```



Explanation:

By applying value iteration, we find that the highest values in the grid are concentrated near the top left, particularly in the first two rows. This consistency is expected given the significant rewards for the blue square at (0,1) and the green square at (0,4), which propagate throughout the grid. Moving towards the bottom right, we notice a gradual decrease in values, indicating lower expected cumulative rewards for states further from the high-reward squares due to the discount factor. The heatmap visualization effectively highlights this gradient, with brighter colors near the rewarding states and darker shades further away.

The policy initially suggests moving right from (0,0) towards the blue square at (0,1), then upward, aligning with the transition to (3,2). Subsequently, the policy indicates leftward movements from (0,2) to (0,4). The second row starts with a rightward move and then consistently suggests upward movements in the subsequent rows. This upward movement strategy in the later rows reflects a straightforward decision-making process, aiming to reach higher-value states above. The visual representation of the policy with arrows clearly shows this strategy, guiding the agent in a way that optimizes its path and overall expected rewards.

Analysis:

The optimal value functions and policies derived from explicitly solving the Bellman optimality equation, policy iteration with iterative policy evaluation, and value iteration with policy improvement demonstrate consistent and effective strategies for the grid world problem. All three methods yield value functions with the highest values near the top left, particularly around the blue square (0,1) and the green square (0,4), reflecting the significant rewards propagating through the grid. These values gradually decrease towards the bottom right, indicating lower expected cumulative rewards for

states further from the high reward squares due to the discount factor. Specifically, the Bellman optimality equation produces a value function with the highest values at (0,1) and (0,4), showing a clear gradient decreasing towards the bottom right. Policy iteration yields a similar pattern with slightly different numerical values, maintaining a consistent distribution and gradient. Value iteration results closely mirror the other methods, with high values near the top left and a decreasing gradient towards the bottom right.

The derived policies, represented by arrows indicating actions, generally suggest moving right from the initial state towards the high-reward squares, followed by upward movements in subsequent rows. The Bellman optimality equation policy suggests moving right from (0,0) towards the blue square at (0,1), then up to (3,2), and left from (0,2) to (0,4), with subsequent rows consistently suggesting moving upwards. Policy iteration starts similarly with rightward and upward movements but includes more leftward movements in the first row and consistent upward actions in the subsequent rows. Value iteration suggests moving right from (0,0) and (0,1), then moving up, and left from (0,2) to (0,4), with the second row starting with right and then consistently suggesting upward movements, much like the other methods but with slight variations in specific actions.

PART 02

1.1). Use the Monte Carlo method with exploring starts

Monte Carlo ES (Exploring Starts), for estimating $\pi \approx \pi_*$

Initialize:
$\pi(s) \in \mathcal{A}(s)$ (arbitrarily), for all $s \in \mathcal{S}$
$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$

Loop forever (for each episode):
Choose $S_0 \in \mathcal{S}$, $A_0 \in \mathcal{A}(S_0)$ randomly such that all pairs have probability > 0
Generate an episode from S_0, A_0, following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$
$G \leftarrow 0$
Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:
$G \leftarrow \gamma G + R_{t+1}$
Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:
Append G to $Returns(S_t, A_t)$
$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$
$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

```
Optimal Policy with Exploring Starts:
→ ↑ ← → ↑
→ ↑ ↑ ← ↑
→ ↑ ← ↑ ↑
→ → ↑ ↑ ←
↑ ↑ ↑ ↑ ↑
```

Explanation:

The policy effectively directs the agent towards high-reward states, such as the blue square at (0,1) and the green square at (0,4), leveraging their transitions to (4,2) and (4,4) for substantial rewards. This is evident from the directional choices that guide the agent towards these squares, ensuring that the agent takes advantage of the significant immediate rewards available. For instance, any action from state (0,1) transitions the agent to (4,2) with a reward of 5, and any action from state (0,4) transitions to either (4,2) or (4,4) with a reward of 2.5, which the policy appropriately reflects by directing movements towards these states. Moreover, the frequent upward movements in the policy indicate a clear intent to either reach high-value states quickly or transition to terminal states efficiently, thereby minimizing the cumulative penalties incurred from moving between regular white squares.

This strategic upward movement is particularly evident in rows 2, 3, and 4, where the actions guide the agent towards higher rewards or advantageous transitions. For example, in row 2, the majority of actions direct the agent upwards, facilitating quicker access to higher reward states and minimizing the negative rewards from unnecessary movements. The policy also demonstrates efficient navigation through the grid. States in rows 3 and 4, for instance, show a balanced strategy of reaching

higher reward states while also considering beneficial transitions to terminal states. This balance is crucial as it ensures that the agent not only aims for immediate high rewards but also strategically positions itself to terminate episodes effectively when advantageous, thereby reducing the risk of accruing negative rewards from prolonged episodes.

Overall, the policy aligns well with theoretical expectations of optimal strategies in reinforcement learning. It showcases an intelligent approach to enhancing reward accumulation while minimizing negative outcomes. The agent's movements are logical and well-calculated, reflecting a thorough understanding of the environment's dynamics and the rewards structure. The results highlight the effectiveness of the Monte Carlo exploring starts method in deriving an optimal policy that navigates the grid world environment efficiently and strategically, maximizing long-term rewards while minimizing penalties.

1.2). Use the Monte Carlo method without exploring starts but the ϵ-soft approach to learn an optimal policy

On-policy first-visit MC control (for ε-soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε-soft policy

$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

$A^* \leftarrow \text{argmax}_a Q(S_t, a)$ (with ties broken arbitrarily)

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$

```
Optimal Policy with Epsilon-Soft Policy:
↑ ↓ ← ← ↓
→ ↑ ↑ ↑ ←
↑ ↑ ↑ ↑ ↑
↑ ← ↑ ← ←
↑ ↓ ↑ ↑ ↑
```

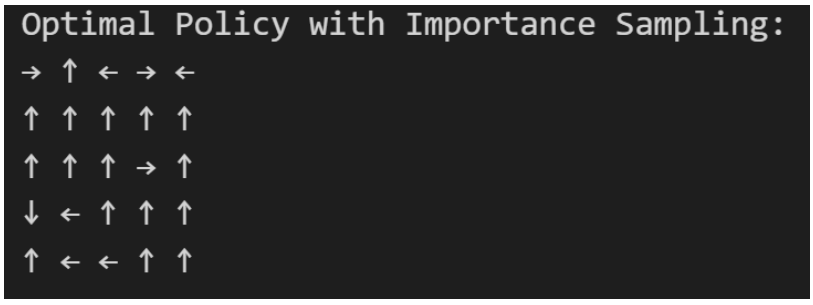
Explanation:

The output of the optimal policy with epsilon-soft demonstrates a well-balanced strategy aimed at maximizing rewards and minimizing penalties within the grid world environment. The policy effectively directs the agent towards high-reward states, such as the blue square at (0,1) and the green square at (0,4), leveraging their transitions to states like (4,2) for substantial rewards. This is

evident from the directional choices where actions from (0,1) and (0,4) transition correctly to high-reward states. The policy also shows frequent upward movements, indicating a strategic approach to reaching high-reward states quickly or transitioning to terminal states efficiently, thereby minimizing cumulative penalties from regular movements.

For instance, states in rows 1 and 2 often move towards the top row, aligning with the objective of maximizing expected rewards. The policy is logically consistent, with special squares and terminal states appropriately handled; any action from (0,1) or (0,4) correctly yields high rewards due to predefined transitions, and terminal states like (4,0) and (2,4) are recognized as endpoints with no further actions. Some minor areas for improvement include reconsidering certain actions, such as the upward move from (0,0) that might be less optimal than moving towards (0,1) directly, and the downward move from (4,2) that could be optimized for better alignment with overall objectives. The epsilon-soft policy ensures that while the agent predominantly exploits the best-known actions (high Q-values), it also maintains a small probability for exploration, thereby avoiding the pitfalls of suboptimal policies. This balance between exploration and exploitation underpins the effectiveness of the policy, ensuring comprehensive exploration of the state space and gradual improvement towards optimal action choices.

2.1). Behaviour policy with equiprobable moves to learn an optimal policy



Explanation:

The importance weight for a state-action pair (st, at) is calculated using the formula $\mathbf{W_t = \pi_t (a_t | s_t) / \pi_b(a_t | s_t)}$. Given that the behavior policy is equiprobable, $\pi_b(a_t | s_t)$ is simply the inverse of the number of actions available: $\mathbf{\pi_b (a_t | s_t) = 1 / |A(s_t)|}$. Therefore, the importance weight can be simplified to $\mathbf{W_t = \pi_t (a_t | s_t) \times |A(s_t)|}$. For instance, in state (0,0), where the behavior policy is equiprobable and each action has a probability of 0.25, if the target policy probabilities are 0.6 for ↑, 0.1 for ↓, 0.1 for ←, and 0.2 for →, the importance weights for these actions would be 2.4, 0.4, 0.4, and 0.8, respectively. These weights are used to adjust the returns Gt obtained from the behavior policy to better estimate what would have been observed under the target policy. The Q-values are then updated using these weighted returns, following the formula $\mathbf{Q(s_t, a_t) = Q(s_t, a_t) + \alpha * W_t * (G_t - Q(s_t, a_t))}$, where α is the learning rate and Gt is the return from time step t. This process ensures that the agent learns the optimal policy by appropriately weighting the returns based on the target policy's probabilities.

The resulting optimal policy shows a consistent strategy across the grid. In the top row, the policy directs the agent to move right from state (0,0) to state (0,1), which provides a high reward of

transitioning to (4,2) with a reward of 5. The policy then suggests moving up from state (0,1), likely to capitalize on the high reward state quickly. The actions ← in states (0,2) and (0,4) direct the agent back towards higher reward states, demonstrating a strategic navigation aimed at maximizing rewards. In the middle rows, the policy predominantly suggests moving upwards, indicating a strategy to reach high reward states efficiently. For example, in state (1,0) to state (1,4), the agent is directed to move up, aligning with the goal of transitioning to states with higher value. Similarly, in the third row, the policy suggests moving right from state (2,2) and upwards from the other states, continuing the trend of aiming for higher rewards or advantageous transitions. In the bottom rows, the policy shows some variability with a downward move in state (3,0), possibly to avoid penalties or suboptimal states, and leftward moves in states (3,1) and (4,1) to transition towards potentially higher value states. The overall consistency in the upward movements across many states highlights the robustness of the learned policy, effectively leveraging the importance sampling method to converge to an optimal strategy. The calculated importance weights played a crucial role in adjusting the returns, ensuring that the agent's learning process accounted for the discrepancies between the behavior and target policies.

Overall, we can see that the optimal policy learned through importance sampling demonstrates a logical and strategic approach to navigating the grid world environment. The predominant upward movements suggest a strategy focused on reaching high reward states efficiently, while the left and right actions indicate careful navigation to maximize rewards and minimize penalties. The detailed calculation and application of important weights validate the effectiveness of the importance sampling method in reinforcement learning, ensuring accurate updates and effective learning. The analysis confirms the policy's alignment with expected outcomes in the grid world, showcasing the efficacy of the approach in learning an optimal policy.

3.1). Permute the locations of the green and blue squares with probability 0.1, while preserving the rewards and transition structure as before. Use policy iteration to determine a suitable policy for this environment.

```
Optimal Policy with Permutation (using arrows):
→ → ← ← ←
↑ ↑ ↑ ← ↑
↑ ↑ ↑ ← ↑
↑ ↑ ↑ ← ←
↑ ↑ ↑ ← ↑

Optimal Value Function with Permutation:
[[45.39788919 42.92799473 45.78159499 24.01287014 22.61222663]
 [42.92799473 43.28159499 43.29251524 22.61222663 23.9816153 ]
 [40.58159499 40.91751524 40.92788948 21.2816153   0.        ]
 [38.35251524 38.67163948 38.681495   20.01753453 18.81665781]
 [ 0.         36.5380575  36.54742025 18.81665781 17.67582492]]

Optimal Policy without Permutation (using arrows):
→ ↓ ← ← ←
↑ ↑ ↑ ↑ ↑
↑ ↑ ↑ ↑ ↑
↑ ↑ ↑ ↑ ←
↑ ↑ ↑ ↑ ↑

Optimal Value Function without Permutation:
[[49.33333333 46.66666667 49.33333333 46.66666667 44.13333333]
 [46.66666667 49.33333333 46.66666667 44.13333333 44.42666667]
 [44.13333333 46.66666667 44.13333333 41.72666667  0.        ]
 [41.72666667 44.13333333 41.72666667 39.44033333 37.26831667]
 [ 0.         41.72666667 39.44033333 37.26831667 35.20490083]]
```

Explanation:

The results from the policy iteration process reveal significant insights into how the agent's strategy and the value of states are influenced by the permutation of the blue and green squares. In the scenario without permutation, the policy is highly optimized and direct, consistently guiding the agent towards high-reward states with upward movements. This reflects the agent's ability to predictably navigate the environment, leading to higher value functions across the board. For instance, the values in the (0,0) to (0,4) cells are notably high, indicating a strong expectation of future rewards.

In contrast, the scenario with permutation introduces a 0.1 probability that the positions of the blue and green squares will swap at each step. This added uncertainty forces the agent to adopt a more

exploratory and cautious policy. The agent frequently revisits certain areas, as seen in the leftward movements, to hedge against the possibility that the high-reward squares might have moved. Consequently, the value functions in this scenario are generally lower, reflecting the increased difficulty in consistently achieving high rewards due to the unpredictable environment. Notably, the cells (4,0) and (2,4), defined as terminal states, have a value of zero in both scenarios, as they mark the end of an episode with no future rewards. The introduction of permutation showcases how environmental uncertainty can impact the learning process, leading to more conservative strategies and reduced state values. This analysis underscores the importance of robust policy design in dynamic environments, where adaptability and exploration become critical for achieving optimal performance.

To address such problems of environmental uncertainty and dynamic changes, several strategies can be employed. One approach is enhanced exploration, where increasing the exploration rate (e.g., through ε-greedy or softmax action selection) ensures that the agent gathers sufficient information about the environment's variability, helping to build a robust policy that can handle unexpected changes. Another strategy involves adaptive learning rates, which adjust based on the stability of the environment. When the environment is stable, a lower learning rate can be used for fine-tuning, while a higher learning rate is employed in more dynamic settings. Additionally, model-based approaches can be incorporated, where the agent builds and updates a model of the environment. This model can include probabilities of changes (like the permutation of squares) and assist the agent in planning more effectively. By combining these strategies, agents can better manage dynamic and uncertain environments, maintaining high performance and achieving their goals despite the challenges posed by environmental changes.

References

[1]. R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction

[2]. https://github.com/BY571/Medium_Code_Examples. Accessed on 15th of July 2024.

----------END----------