

Reinforcement Learning: Assignment 2

DSCI-6650

Soroush Baghernezhad
sbaghernezh@mun.ca

July 17, 2024

Abstract

In this assignment, we interact with a grid world and try different dynamic programming approaches and Monte Carlo methods to achieve the optimal policy in the environment. We also compare different methods to see how they perform and what is their final policy.

1 Introduction

The environment is a 5*5 grid with different colors, stepping in blue tile and moving in any direction will results in jumping to red tile and receiving the reward of +5, same for green but there is 50% chance to jump to green tile and 50% chance to jump to yellow tile and receiving +2.5 reward.

In the second part, the grid is a little modified where there are two terminal states added and red tiles relocated.

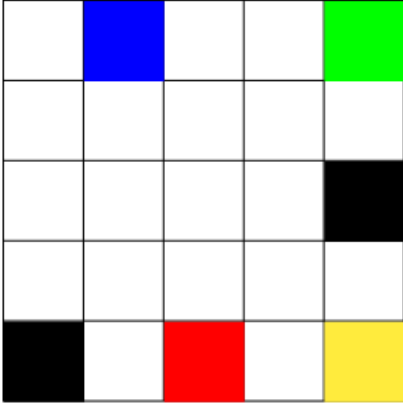


Figure 1: grid for second part

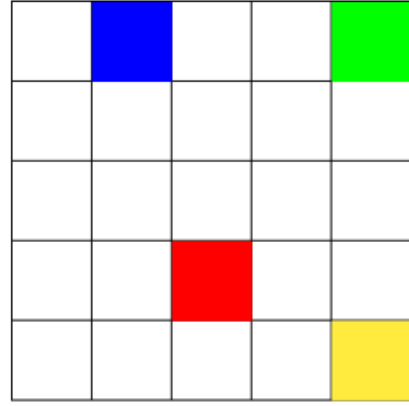


Figure 2: grid for first part

2 Part One

2.1 Question two

In this question we are suppose to estimate the value function of a equiprobable policy.

2.1.1 solving the system of Bellman equations explicitly:

Bellman equation can be expressed concisely using matrices as follow:

$$V = R + \gamma PV$$

$$\begin{bmatrix} v(1) \\ \cdot \\ \cdot \\ v(n) \end{bmatrix} = \begin{bmatrix} R(1) \\ \cdot \\ \cdot \\ R(n) \end{bmatrix} + \gamma \begin{bmatrix} P(11)...P(1n) \\ \cdot \\ \cdot \\ P(11)...P(nn) \end{bmatrix} \begin{bmatrix} v(1) \\ \cdot \\ \cdot \\ v(n) \end{bmatrix}$$

By solving this system of linear equations, we can get the values of each state.

2.1.2 Iterative policy evaluation:

This approach is one of the dynamic programming methods to estimating the value of a policy by iteratively going through all states following the policy under evaluation and then getting the return for that state.

2.1.3 Value iteration.

Evaluating a policy with this method is not feasible since in value iteration policy is changing constantly and what we are evaluating is not the same policy.

2.2 Question two

In this question, we find the optimal policy by following three approaches as below.

2.2.1 explicitly solving the Bellman optimality equation

After solving the equation discussed earlier, we can get an argmax over the actions of value function to get the best action at each state.

2.2.2 using policy iteration with iterative policy evaluation

This approach uses the general idea of Generalized Policy Iteration (GPI) in which we start with an arbitrary policy, we evaluate it and we improve it with its value function, then we again evaluate and improve consequently to converge to a point where no more improvement and evaluation is needed.

2.2.3 policy improvement with value iteration

This method truncate the idea of policy evaluation at each iteration in policy iteration. So we can first iteratively improve the value function of states and then find an optimal policy based on the improved value function after convergence.

3 Results: Part One

3.1 Question one

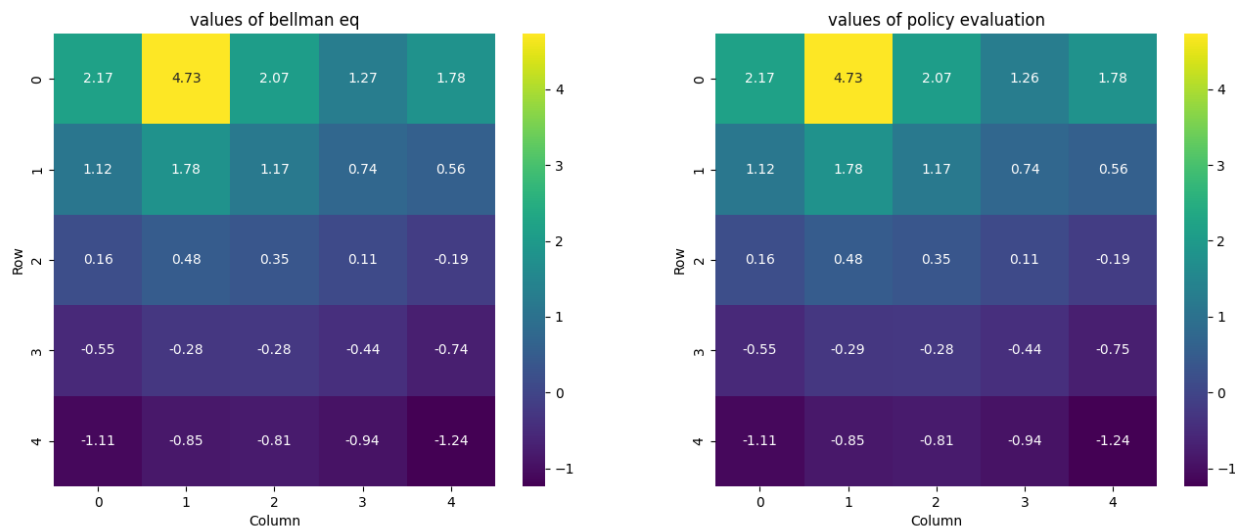


Figure 3: State values of a policy with equiprobable moves, comparison between two methods.

State that contain Blue color is the most valuable state based on this estimation, and states around it are the second most valuable states because they lead to the blue state which finally return a +5 rewards for any move.

The fact that values achieved by policy evaluation algorithm is very similar to values obtained by solving bellman equation is not surprising; by the rule of large numbers we were expecting that policy evaluation estimation should be similar to the true values after large numbers of iteration over states.

3.1.1 Parameter tuning

To see the effect of different input parameters on our methods (Figure 4), we studied 4 different discount factors to see how they change the value functions. The values we used are $\gamma = 0.95, 0.70, 0.50, 0$. It can be seen that when gamma goes to zero, state-values are tend to be more like the immediate reward of that state.

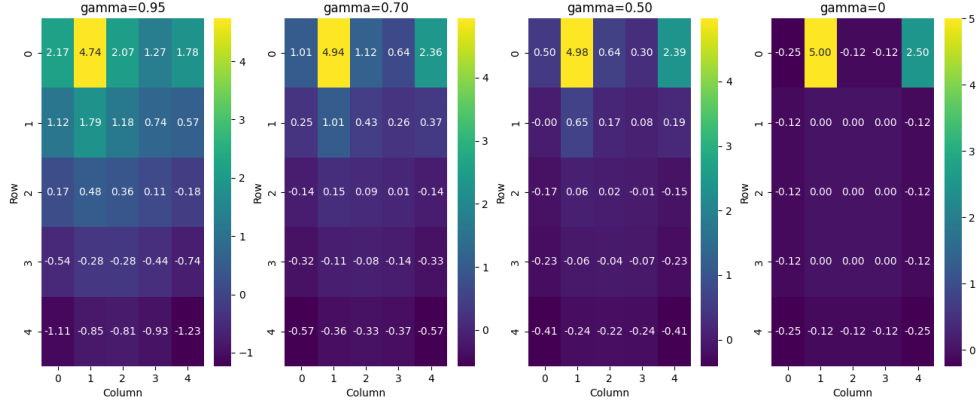


Figure 4: Different gammas for policy evaluation.

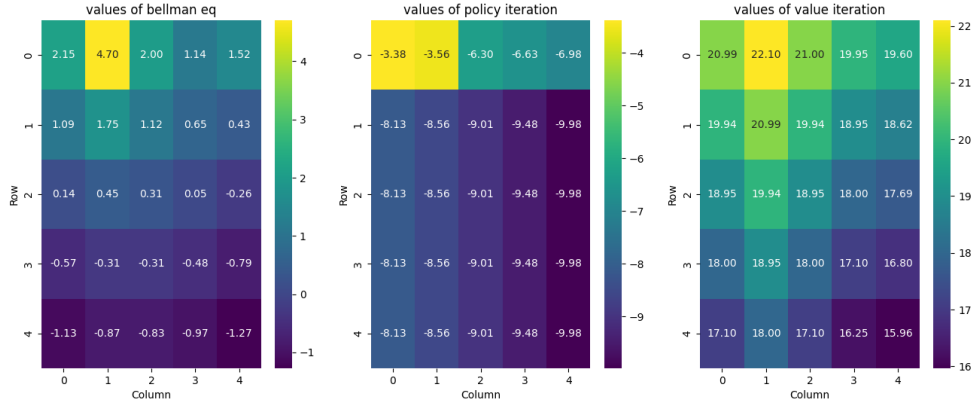


Figure 5: Final value estimation of solving bellman equation.

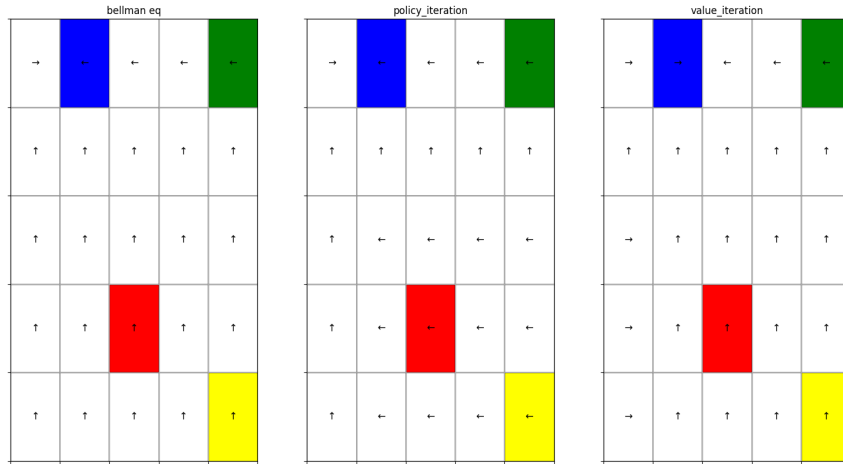


Figure 6: Optimal policies obtained by three methods with respect to θ value.

3.2 Question two

Figure 5 suggest that all three methods think that the blue tile with tiles around are the best states to be in, the difference between their values roots in looping for many times in order to converge.(or satisfying the terminating condition)

Figure 6 shows the direction in which moving leads to maximum reward. these directions were obtained by getting the maximum argument from the value functions of above.

Since there is no negative reward for taking actions or moving to white tiles in this part, the intuitive approach would be avoiding out-of-grid movement while we are on the edges and getting to the blue tile which leads to the maximum reward in this environment.

Given that there is no terminal state and no time limit, we can assume that a policy can be optimal in this grid if:

1. there is no cycle with the return of zero that agent stuck on it
2. no actions lead to negative reward

It is interesting that all three policies tend to go toward blue tile even if green tile is in their neighbourhood.

Gradient method with the use of action preferences chooses the most optimal actions between other methods.

3.2.1 Parameter tuning

To see the effect of different input parameters to our final policies of this three methods, we interact with grid environment 100 times and each time we limited the environment to only 100 steps, to avoid running forever, then we averaged the accumulated rewards over those 100 trials. In table below you can see the results of this parameter tuning.

	$\gamma = 0.95$	$\gamma = 0.70$	$\gamma = 0.50$	$\gamma = 0$
Policy Iteration	100.0	100.0	100.0	100.0
Policy Evaluation	75.0	75.0	75.0	75.0

Table 1: Comparison of Policy Iteration and Policy Evaluation for different γ values

4 Part Two

In this part, we used a modified version of the grid in which two terminal states with black color are present in (4,0) and (2,4) positions. This terminal states will end an episode and return the reward accumulated. s

There is also a negative reward added for moving to white tiles.

4.1 Question one

4.1.1 Monte Carlo with exploring starts

In this part we arbitrarily initialize the S_0 and A_0 .

4.1.2 Monte Carlo with ϵ -soft approach

In this part we use an $\epsilon - greedy$ policy to ensure that all states can be visited.

4.2 Question two

4.2.1 Monte Carlo with off policy approach

In this question, we implemented off policy MC in which there are two policies:

1. behaviour policy: the policy that is responsible for generating state, action and reward sequences. This policy is usually $\epsilon - soft$ so it can explore whole state space.
2. target policy: the policy that is targeted for improvement.

4.3 Question three

In this question, we permute the locations of the green and blue squares with probability 0.1 and we apply policy iteration from part1 to it to see how can it performs.

5 Results: Part two

5.1 Question one

As it can be seen in figure 7 exploring start approach tends to perform better as it can choose shorter path to get to the blue tile whereas soft policy approach can even get stuck in a loop (states (3,2) and (3,3)), hence with the same number of episodes exploring start outperform epsilon-greedy approach. Nonetheless, both policies prefer to terminate the episode as soon as they get to the proximity of a terminal state instead of going up and get a +5 reward.



Figure 7: Optimal policies obtained by MC with exploring start and epsilon soft.

5.1.1 Parameter tuning

we tested a set of discount factors for both MC methods: $\{0.95, 0.75, 0.50, 0\}$

we also examined the effect of epsilon on the MC with soft policy. the epsilons we used are: $\{0.01, 0.1, 0.2, 0.3\}$

We also interact with grid world with each of the tuned policies and the results are as follow :

	$\gamma = 0.95$	$\gamma = 0.75$	$\gamma = 0.50$	$\gamma = 0$
MC with exploring start	4.8	4.8	4.8	-4.8
MC with soft policy	4.8	4.8	4.8	4.8

Table 2: Comparison of reward achieved by MC with exploring start and MC with soft policy over 1000 episodes for different γ values

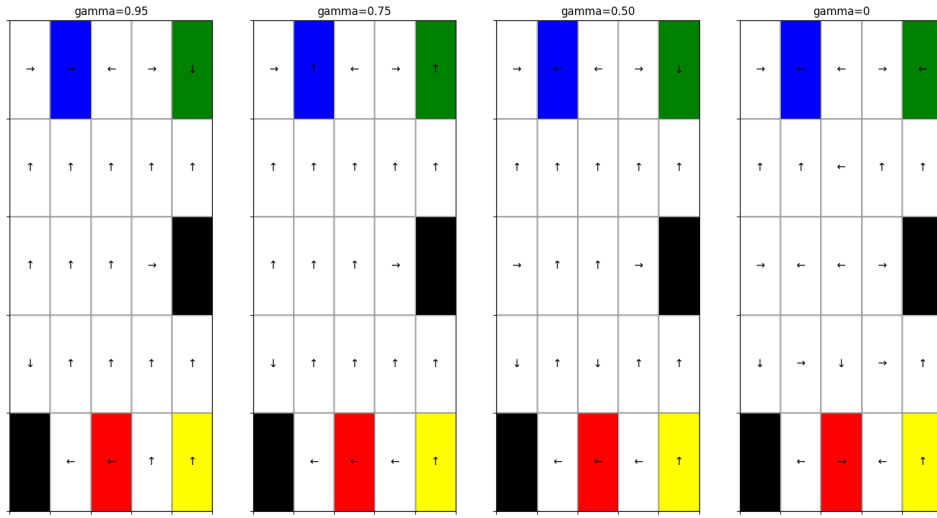


Figure 8: Optimal policies obtained by MC with exploring start over different gammas.

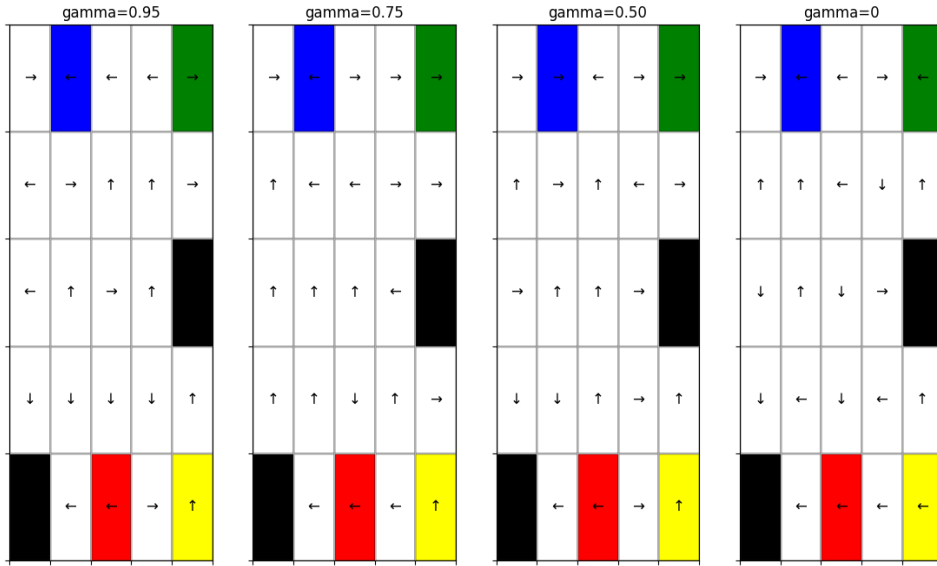


Figure 9: Optimal policies obtained by MC with soft policy over different gammas.

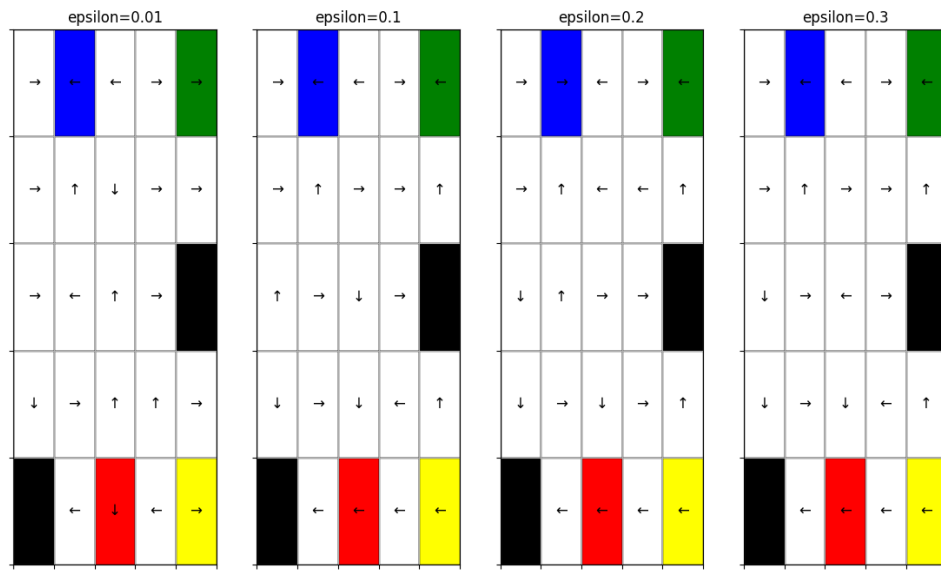


Figure 10: Optimal policies obtained by MC with soft policy over different epsilons.

5.2 Question two

As shown in figure 11 off policy MC could not efficiently converge to a good policy with its equiprobable behavioural policy.

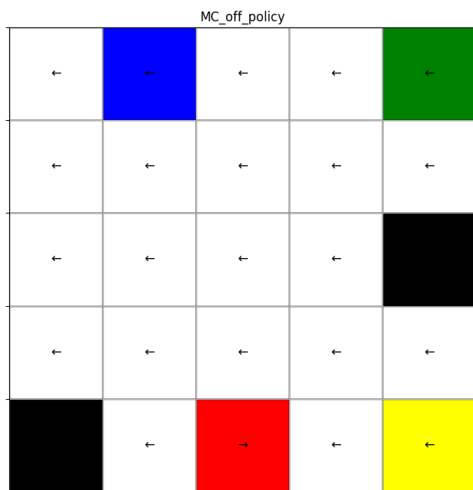


Figure 11: Optimal policies obtained by off policy MC.

5.3 Question Three

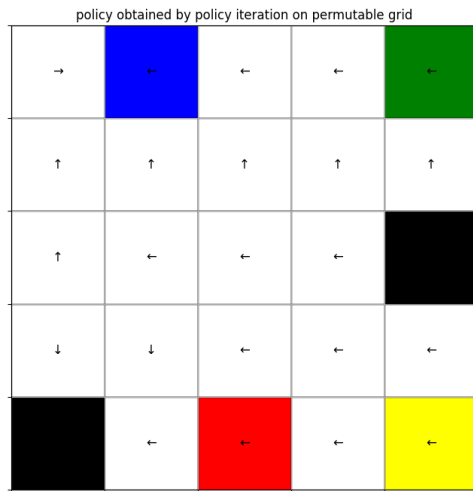


Figure 12: Optimal policies obtained in permutable grid.

References

Richard S. Sutton and Andrew G. Barto. 2018. Reinforcement Learning: An Introduction. A Bradford Book, Cambridge, MA, USA.

RL Course by David Silver - Lecture 2: Markov Decision Process, howpublished = <https://www.youtube.com/watch?v=1fHX2hHRMVQ&t=2349s>,