Assignment 5
GridWorld

Jonas Kulhanek

Abstract—This paper is the result of my assignment concerned with dynamic programming. In the assigment I worked with python to implement some algorithms used in dynamic programming and did some tests on them.

Figure 1. Plot showing how values of selected states of game 3x4 changes with change in probability.

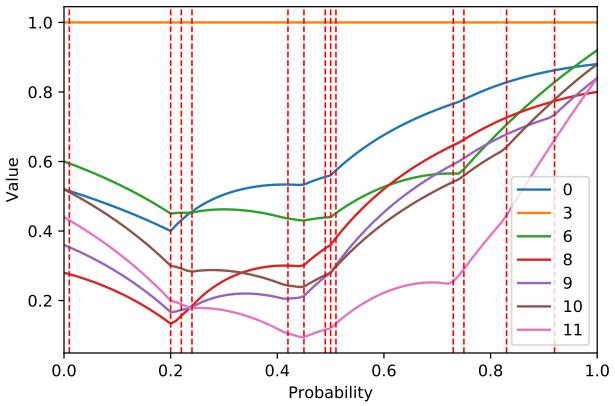
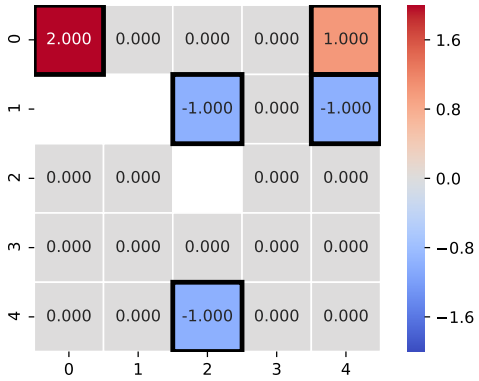


Figure 2. The 5x5 board



1 Introduction

The objective of this assignment was to experiment with reinforcement learning techniques. Specifically with dynamic programming. In the implementation part, I was supposed to implement value iteration and policy iteration techniques used in dynamic programming. In the experiment part I work with them to get some insight and to learn effectively, how they work. The experiments presented here are implemented in python jupyter notebooks. They are included in the homework. In the source code I used python with numpy library.

Experiment 1

In this experiment I analyzed how the valuation of selected states changed with a change in probability. The experiment was analyzed on the 3x4 board. I calculated the value for probabilities taken uniformly from the interval [0,1]. The result can be seen in figure 1. A vertical line represents the probability, where a policy change occurs. Furthermore, those probabilities are given in the following table.

0.01	0.2	0.22	0.24	0.42	0.45	0.49
0.5	0.51	0.73	0.75	0.83	0.92	

It can be seen, that on the places, where the policy was changed, the value of the board started increasing again.

Experiment 2

In the second experiment, I analyzed, how the policy changed with a change in transition cost. The experiment was done on 5x5 board. Because, I needed to test it on unbounded interval - [0, ∞), I took the [0, 1) interval and transformed it using a function, that stretched it to the desired interval. For the details on the used function, please refer to Appendix 1
Transform function The change in policy occured on those values:

0.02	0.27	0.34	0.38	0.46	0.64	0.96
1.04	2.00	2.92	3.24			

The 5x5 board can be seen in the figure 2

I present you with some of the policies here. The optimal policy for transition cost 2.92 can be seen in figure 3. The policy for 0.46 in figure 4. And for 0.27 in figure 5. The arrows indicate the direction to destination state taken by the policy and the color of each block is the value of that state.

We can see, that the willingness of taking an action decreases with higher cost. It can be seen in figure 4 and figure 3, that it is better to take a -1 action instead of going for higher valued state.

Figure 3. Optimal policy for transition cost 2.92 on board 5x5

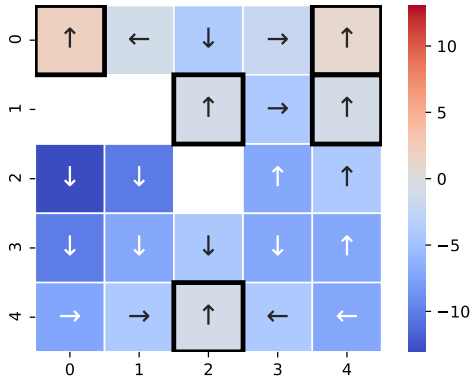


Figure 4. Optimal policy for transition cost 0.46 on board 5x5

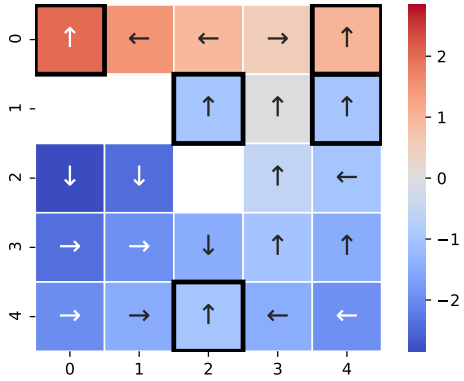


Figure 5. Optimal policy for transition cost 0.27 on board 5x5

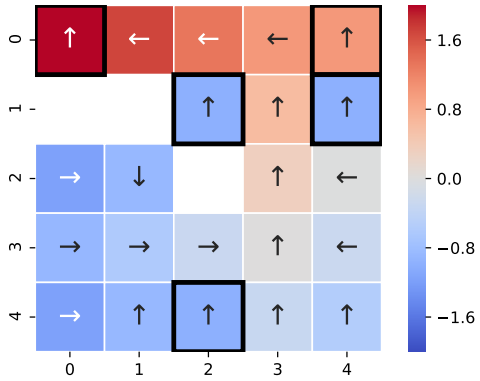


Figure 6. Heatmap showing influence of probability and transition on the transition cost of value of state 10 of board 3x4

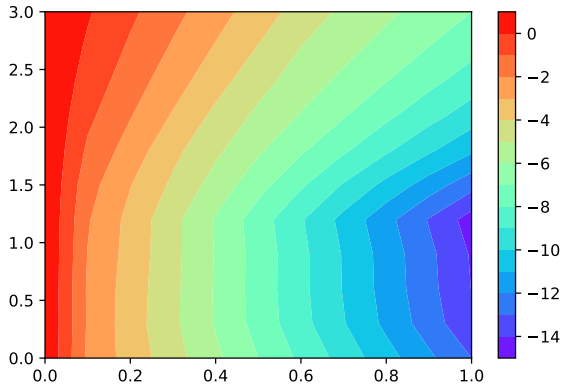
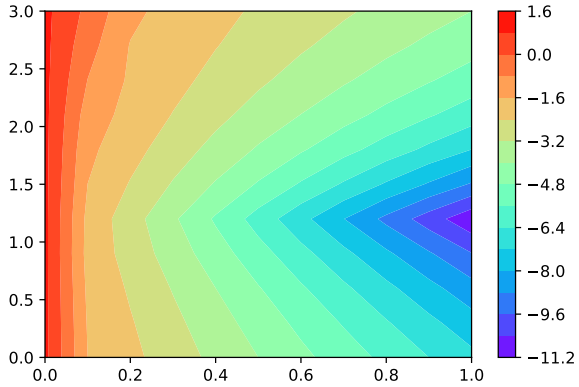


Figure 7. Heatmap showing influence of probability and transition on the transition cost of value of state 11 of board 3x4



Experiment 3

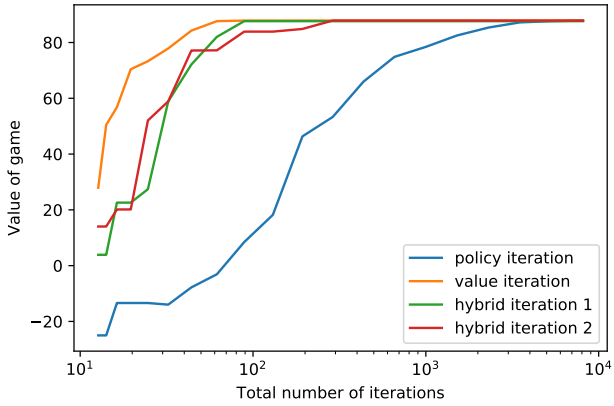
In this experiment I have decided to look at the correlation of probability and the transition cost. Specifically how they influence the value of some of the states together. The result is a heatmap, where the color is given by the value of the state with the probability on the x axis and the transition cost on the y axis. The test was done on 3x4 grid. Few chosen states are presented in figure 6 for state 10 and in figure 7 for state 11.

In the heatmap, it can be seen, that the willingness of of taking an action decreases with lower probability and decreases with higher transition cost as expected.

Experiment 4

In my final experiment I compared the the value vs. the policy iteration. Furthermore I modified the policy iteration to the "hybrid" iteration and compared it with the policy and the value iterations. First I will present you with my modification of the policy iteration which I call "hybrid" iteration. In the policy iteration, a policy is chosen at the beginning and then the value of the board is calculated for this policy. The value of the board is calculated in several iterations. Then the optimal policy for that value is chosen

Figure 8. Comparison of speed of convergence of several algorithm for best policy estimation.



and the process repeats. My modification was to persist the value between iterations and reduce the number of iterations in each policy iteration needed to get the value of that policy. The objective of this modification is, that at the beginning the value iteration is too unstable because the policy changes too often. After it gets stable there is no need to make so much iterations for value calculation in each policy iteration, because the value gets more stable. My approach is somewhere in the middle between value and policy iterations. First it acts as the policy iteration and with more iterations it becames more like the value iteration.

All approaches are compared in figure 8. The plot has the joint number of iterations on the x axis and total value of the board on the y axis. For the policy iteration the number of iterations is distributed equally for the inner iterations and policy iterations. Their product is the same as for value iteration. For the "hybrid" approach, I had chosen two strategies. One of them should act first as the policy iteration and then as the value iteration. The second algorithm is the same as the policy iteration, but is persists the value of the board between iterations.

The experiment was conducted on 12x6 board. The value estimation appeared to be the best algorithm for this problem, the "hybrid" approach 1 was second, closely followed by the second "hybrid" approach. The policy iteration was the worst for this scenerio.

Appendix A
Transform function

In this section, the transform function used in the second experiment is described. This function takes values in range [0, 1) and transforms them to the interval [0, ∞). In my experiment, I have chosen this function:

$$(1/(1-X)) - 1 \tag{1}$$

The plot of this function can be seen in the figure 9.

Figure 9. The transform function [0, 1) -> [0, ∞)