RL Homework 3

Rishabh Verma

November 2024

1 Exercise 1

(a) We are given the dynamical system

$$x_{n+1} = \begin{cases} -x_n + 1 + u_n & \text{if } -2 \leq -x_n + 1 + u_n \leq 2, \\ 2 & \text{if } -x_n + 1 + u_n > 2, \\ -2 & \text{else} \end{cases}$$

where $x_n \in \{-2, -1, 0, 1, 2\}$ and $u_n \in \{-1, 0, 1\}$, and the cost function

$$J = \left(\sum_{k=0}^{2} 2|x_k| + |u_k| \right) + x_3^2$$

The terminal cost at stage 3 would always be the same at $J_3 = x_3^2$
Therefore for $x_3 = -2 \implies J_3 = 4$
$x_3 = 1 \implies J_3 = 1$
$x_3 = 0 \implies J_3 = 0$
$x_3 = 1 \implies J_3 = 1$
$x_3 = 2 \implies J_3 = 4$
For the next iterations our cost would be J = current cost + cost-to-go from the next state
For x = -2,
If u = -1, $x_{new} = 2, \implies J_2(x_0) = 2 * \| -2\| + \| -1\| + \| -2\|^2 = 9$
If $u = 0$, $x_{new} = 2, \implies J_2(x_0) = 2 * \| -2\| + \|0\| + \| -2\|^2 = 8$ (**Minimum**)
If $u = 1$, $x_{new} = 2, \implies J_2(x_0) = 2 * \| -2\| + \|1\| + \| -2\|^2 = 9$
Here we find when moving backwards the minimum cost is at $u = 0$ for stage 2 for the the state x_0.

Similarly we can find for other states and other stages and minimize the cost across the stages for that state and control and get the cost by using Dynamic programming formula as taught in lecture 7,

$$J_k(x_k) = \min_{u_k} g_k(x_k, u_k) + J_{k+1}(f(x_k, u_k))$$

State	**Stage 0**		**Stage 1**		**Stage 2**		**Stage 3**
	$\boldsymbol{J_0}$	$\boldsymbol{u_0}$	$\boldsymbol{J_1}$	$\boldsymbol{u_1}$	$\boldsymbol{J_2}$	$\boldsymbol{u_2}$	$\boldsymbol{u_3}$
-2	10	0	9	0	8	0	4
-1	6	-1	5	-1	4	-1	1
0	3	-1	2	-1	1	-1	0
1	4	0	3	0	2	0	1
2	7	1	6	1	5	0	4

Table 1: Values of State, J_k, and u_k for each stage

(b) Next we need to find the sequence of control actions, states and the optimal cost for the given initializing states. For this we find the optimal control for the state and iterate backwards for the next states and give the minimum cost and optimal control
1. Stagewise Policy for Initial State x0 = 0
Stage 0: x = 0, J = 3, u = -1
Stage 1: x = 0, J = 2, u = -1
Stage 2: x = 0, J = 1, u = -1

Stage 4: x = 0, J = 0

2. Stagewise Policy for Initial State x0 = -2
Stage 0: x = -2, J = 10, u = 0
Stage 1: x = 2, J = 6, u = 1
Stage 2: x = 0, J = 1, u = -1
Stage 4: x = 0, J = 0

3. Stagewise Policy for Initial State x0 = 2
Stage 0: x = 2, J = 7, u = 1
Stage 1: x = 0, J = 2, u = -1
Stage 2: x = 0, J = 1, u = -1
Stage 4: x = 0, J = 0

(c) If the constant term 1 in the previous dynamics is sometimes 0 with a probability then mostly the whole process remains the same but the Cost changes to

$$\mathrm{J} = \mathbf{E}\left[\sum_{k=0}^{2}(2|x_k| + |u_k|) + x_3^2\right]$$

$$\implies \mathrm{J} = \text{Current Cost} + 0.4\text{*Cost-to-go(w} = 0) + 0.6\text{*Cost-to-go(w} = 1)$$

Then we iterate similarly backwards like we did previously

For x_0 = -2
If $u = -1 \implies J_2(x_0) = 5 + 0.4 * 1 + 0.6 * 6 = 7.8$ **(Minimum)**
If $u = 0 \implies J_2(x_0) = 4 + 0.4 * 4 + 0.6 * 4 = 8$
If $u = 1 \implies J_2(x_0) = 5 + 0.4 * 4 + 0.6 * 4 = 9$

Then same as in part(a) we iterate over backwards and recieve our optimal cost and control.

State	**Stage 0**		**Stage 1**		**Stage 2**		**Stage 3**
	J_0	u_0	J_1	u_1	J_2	u_2	J_3
-2	10.600	-1	9.200	-1	7.800	-1	4.000
-1	5.952	-1	4.680	-1	3.600	-1	1.000
0	2.952	0	1.680	0	0.600	0	0.000
1	4.880	0	3.800	0	2.400	0	1.000
2	7.880	1	6.800	1	5.400	1	4.000

Table 2: Values of State, J_k, and u_k for each stage

(d) Below I have tabulated the results from both the methods. We have a random variable which causes a slight variation and so we are calculating the expectation instead of the absolute cost which comes to be slightly different. Because the random variable changes the dynamics of the model the optimal control also changes accordingly to get the minimum cost.

State	**Stage 0**		**Stage 1**		**Stage 2**		**Stage 3**
	J_0	u_0	J_1	u_1	J_2	u_2	u_3
-2	10	0	9	0	8	0	4
-1	6	-1	5	-1	4	-1	1
0	3	-1	2	-1	1	-1	0
1	4	0	3	0	2	0	1
2	7	1	6	1	5	0	4

Table 3: Values of State, J_k, and u_k for each stage

State	Stage 0		Stage 1		Stage 2		Stage 3
	J_0	u_0	J_1	u_1	J_2	u_2	J_3
-2	10.600	-1	9.200	-1	7.800	-1	4.000
-1	5.952	-1	4.680	-1	3.600	-1	1.000
0	2.952	0	1.680	0	0.600	0	0.000
1	4.880	0	3.800	0	2.400	0	1.000
2	7.880	1	6.800	1	5.400	1	4.000

Table 4: Values of State, J_k, and u_k for each stage where ω is a probabilistic variable

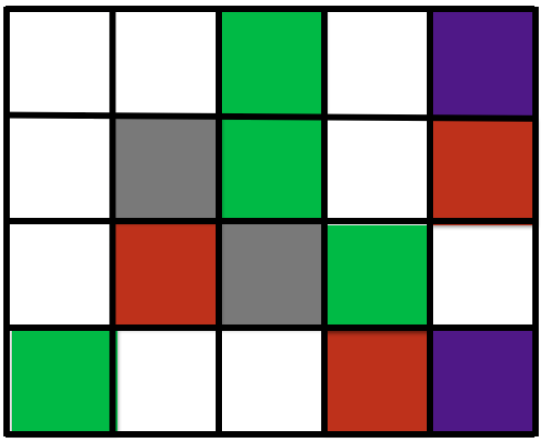


Figure 1: Grid World

2 Exercise 2

Consider the grid-world shown in Figure 1. In each (non-grey) cell, it is possible to perform five actions: move up, down, left, right, or do nothing, as long as the resulting move stays inside the grid world. Grey cells are obstacles and are not allowed. We would like to find the optimal value function and optimal policy that minimize the following cost:

$$\min \sum_{n=0}^{\infty} \alpha^n \, g_n(x_n)$$

with discount factor $\alpha = 0.99$, where the instantaneous cost is defined as

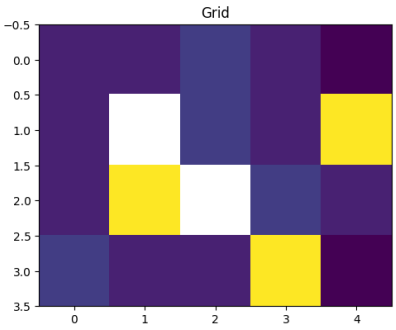
$$g_n(x_n) = \begin{cases} -1 & \text{if } x_n \text{ is a violet cell} \\ 0 & \text{if } x_n \text{ is a white cell} \\ 1 & \text{if } x_n \text{ is a green cell} \\ 10 & \text{if } x_n \text{ is a red cell} \end{cases}$$

(a) The Value Iteration Algorithm2a takes 1520 iterations to converge and takes a time of 0.0003781318664550781 seconds.

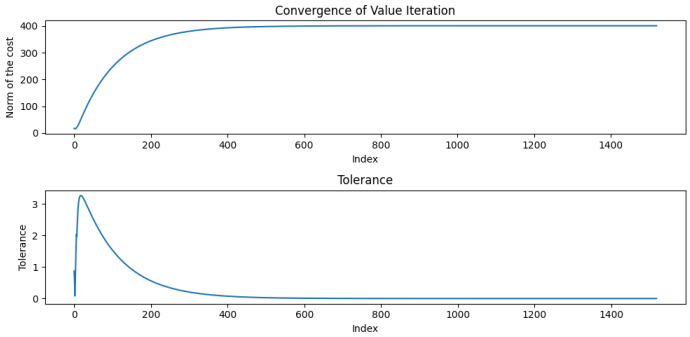
Table 5: Cost of Occupancy Grid

	Column 1	Column 2	Column 3	Column 4	Column 5
Row 1	-95.08	-96.04	-97.01	-99.00	-100.00
Row 2	-94.13	NaN	-96.03	-98.01	-89.00
Row 3	-93.19	-82.26	NaN	-97.01	-99.00
Row 4	-91.26	-90.34	-89.44	-89.00	-100.00

(b) The Policy Iteration Algorithm3a takes 11 iterations to converge and takes a time of 0.00013208389282226562 seconds.



(a) Value Iteration Occupancy Cost

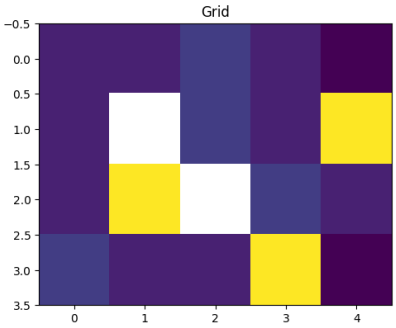


(b) Convergence of Value Iteration Algorithm

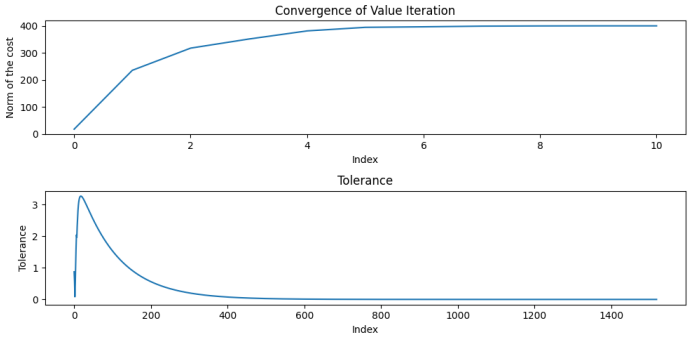
Figure 2: Value Iteration Algorithm

Table 6: Cost of Occupancy Grid

	Column 1	**Column 2**	**Column 3**	**Column 4**	**Column 5**
Row 1	-95.08	-96.04	-97.01	-99.00	-100.00
Row 2	-94.13	NaN	-96.03	-98.01	-89.00
Row 3	-93.19	-82.26	NaN	-97.01	-99.00
Row 4	-91.26	-90.34	-89.44	-89.00	-100.00



(a) Policy Iteration Occupancy Cost



(b) Convergence of Policy Iteration Algorithm

Figure 3: Policy Iteration Algorithm

2.1 Results and Observations

The Policy Iteration Algorithm achieves convergence faster and in fewer iterations. Overall the cost of occupancy of each place in the grid is more or less the same but performance wise Policy Iteration outperforms by a big margin.