Planning, Learning and Decision Making: Homework 2. Markov decision problems

Mykhaylo Marfeychuk, Roy De Prins

16 October 2018

1 Excercise 1) a

State Space $\mathcal{X}$ =

{TopLeft, TopRight, BottomLeft, BottomRight,
Red, TopLeftRed, TopRightRed, BottomLeftRed, BottomRightRed,
Blue, TopLeftBlue, TopRightBlue, BottomLeftBlue, BottomRightBlue,
Goal }.

Action Space $\mathcal{A}$ =

{Up, Down, Left, Right }.

2 Excercise 1) b

Transition Probability Matrix for action 'right':

	TL	TR	BL	BR	Red	TLR	TRR	BLR	BRR	Blue	TLB	TRB	BLB	BRB	Goal
TL	0.2	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
TR	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
BL	0.0	0.0	0.2	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
BR	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Red	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
TLR	0.0	0.0	0.0	0.0	0.0	0.2	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
TRR	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
BLR	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.8	0.0	0.0	0.0	0.0	0.0	0.0
BRR	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
Blue	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.8	0.0	0.0	0.0
TLB	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.8	0.0	0.0	0.0
TRB	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
BLB	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.8	0.0
BRB	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.8
Goal	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

Cost Function for action 'right':

	cost_right
TL	0.5
TR	1.0
BL	0.5
BR	1.0
Red	1.0
TLR	0.5
TRR	1.0
BLR	0.5
BRR	1.0
Blue	0.0
TLB	0.0
TRB	1.0
BLB	0.0
BRB	0.0
Goal	0.0

3 Excercise 1) c

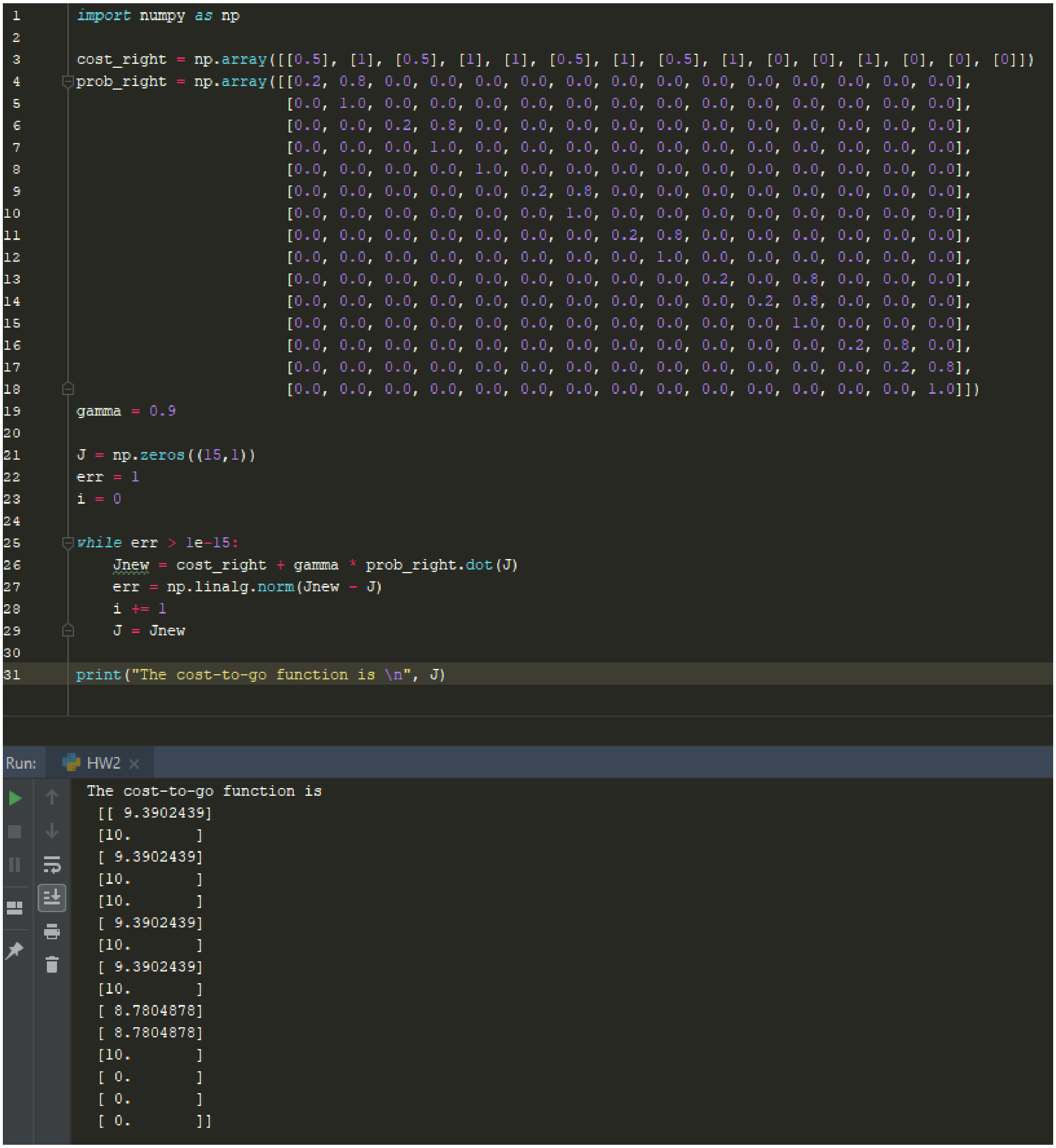
We compute the cost-to-go function with the 'Value Iteration, 2.0' algorithm. So by computing formula 1:

$$\mathbf{J}^t = \mathbf{c}_{right} + \gamma \mathbf{P}_{right} \mathbf{J}^{t-1} \tag{1}$$

The cost-to-go function associated with the policy 'right' is:

cost-to-go
9.39
10.0
9.39
10.0
10.0
9.39
10.0
9.39
10.0
8.78
8.78
10.0
0.0
0.0
0.0

This calculation is done by the program (in Python) in Figure 1.



```python
import numpy as np

cost_right = np.array([[0.5], [1], [0.5], [1], [1], [0.5], [1], [0.5], [1], [0], [0], [1], [0], [0], [0]])
prob_right = np.array([[0.2, 0.8, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.2, 0.8, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.8, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.8, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.0, 0.8, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.8, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.8, 0.0],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.8],
                       [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]])
gamma = 0.9

J = np.zeros((15,1))
err = 1
i = 0

while err > 1e-15:
    Jnew = cost_right + gamma * prob_right.dot(J)
    err = np.linalg.norm(Jnew - J)
    i += 1
    J = Jnew

print("The cost-to-go function is \n", J)
```

```
The cost-to-go function is
 [[ 9.3902439]
 [10.       ]
 [ 9.3902439]
 [10.       ]
 [10.       ]
 [ 9.3902439]
 [10.       ]
 [ 9.3902439]
 [10.       ]
 [ 8.7804878]
 [ 8.7804878]
 [10.       ]
 [ 0.       ]
 [ 0.       ]
 [ 0.       ]]
```

Figure 1: The Python program for calculating to cost-to-go function.