MDP Policy_iteration

August 6, 2020

0.0.1 MDP with Policy Iteration

```
In [92]: import numpy as np
         import itertools
         import matplotlib.pyplot as plt
```

```
In [93]: def get_rewards_and_state(policy,P_tr,R_tr,states,gamma):
             P=P_tr[policy,states,:]
             R=R_tr[policy,states,:]

             s=np.matmul(np.matrix(np.identity(len(states))-gamma*P).I,R).sum()
             return policy,s
```

```
In [94]: def generate_policies(actions,states):
             policies=list(itertools.product(actions,repeat=len(states)))
             return policies
```

```
In [95]: def get_results(policies,P_tr,R_tr,states,gamma):
             results=[]
             for policy in policies:
                 results.append(get_rewards_and_state(policy,P_tr,R_tr,states,gamma))
             #print(results)
             #best_pol=1
             best_pol=sorted(results,key=lambda e:e[1])[-1]
             fig=plt.figure(figsize=(15,5))
             ax=fig.add_subplot(111)
             plt.plot([i for i,e in enumerate(results)],[e[1] for e in results])
             return best_pol
```

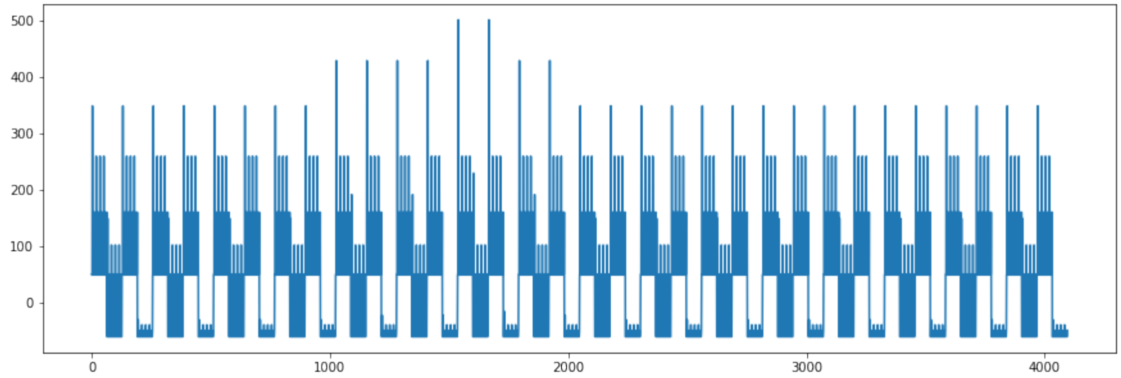
```
In [96]: def caller(actions,states,P_tr,R_tr,gamma):
             policies=generate_policies(actions,states)
             best_pol=get_results(policies,P_tr,R_tr,states,gamma)

             return best_pol
```

```
In [97]: P_tr=np.array([
             [
                 [0,1,0,0,0,0],
                 [0,1,0,0,0,0],
                 [0,0,1,0,0,0],
                 [0,0,0,0,1,0],
                 [0,0,0,0,0,1],
                 [0,0,0,0,0,1],
             ],
             [
                 [0,0,0,1,0,0],
                 [0,1,0,0,0,0],
                 [0,0,0,0,0,1],
                 [0,0,0,1,0,0],
                 [0,0,0,0,1,0],
                 [0,0,0,0,0,1],
             ],
             [
                 [1,0,0,0,0,0],
                 [1,0,0,0,0,0],
                 [0,0,1,0,0,0],
                 [0,0,0,1,0,0],
                 [0,0,0,1,0,0],
                 [0,0,0,0,1,0],
             ],
             [
                 [1,0,0,0,0,0],
                 [0,1,0,0,0,0],
                 [0,0,1,0,0,0],
                 [1,0,0,0,0,0],
                 [0,0,0,0,1,0],
                 [0,0,1,0,0,0],
             ],

         ])

In [98]: R_tr=np.array([
             [
                 [-1],
                 [-1],
                 [10],
                 [-1],
                 [-1],
                 [-1]
             ],
             [
                 [-1],
                 [-1],
                 [-1],
                 [-1],
            [-1],
            [-1]
        ],
        [
            [-1],
            [-1],
            [10],
            [-1],
            [-1],
            [-1]
        ],
        [
            [-1],
            [-1],
            [-1],
            [-1],
            [-1],
            [10]
        ],
    ])
```

```
In [99]: gamma=0.9
```

```
In [100]: actions=[0,1,2,3]
          states=[0,1,2,3,4,5]
```

```
In [101]: policy=caller(actions,states,P_tr,R_tr,gamma)
```



```
In [102]: policy
```

```
Out[102]: ((1, 2, 2, 0, 0, 3), 500.46100000000007)
```

0.0.2 Found Route

```
In [ ]:
```