D1

Tae Hyeon Kwon, undergrad(ITM)

2021-01-13

#Page 10

```
import numpy as np

def soda_simul(this_state):
    n=np.random.random()

    if this_state=='c':
        if n<=0.7:
            next_state='c'
        else:
            next_state='p'
    else:
        if n<=0.5:
            next_state='c'
        else:
            next_state='p'

    return next_state

def cost_eval(path):
    cost_one_path=path.count('c')*1.5+path.count('p')*1
    return cost_one_path

MC_N=10000
spending_records=np.zeros((MC_N,))

for i in range(MC_N):
    path='c'

    for t in range(9):
        this_state=path[-1]
        next_state=soda_simul(this_state)
        path+=next_state

    spending_records[i]=cost_eval(path)

print(spending_records)
```

```
## [14.5 13.  13.  ... 14.5 13.  13.5]
```

#Page 11

```
episode_i = 0
cum_sum_G_i = 0
num_episode = 10000

while episode_i < num_episode:
  path = 's'
  for t in range(9):
    this_state = path[-1]
    next_state = soda_simul(this_state)
    path = path+next_state

  G_i = cost_eval(path)

  cum_sum_G_i = cum_sum_G_i + G_i

  episode_i +=1

V_t = cum_sum_G_i / num_episode

print(V_t)
```

```
## 11.73585
```

#page 20

```
import numpy as np

P = np.array([0.7,0.3,0.5,0.5]).reshape(2,2)
R = np.array([1.5,1.0]).reshape(2,1)
H = 10
v_t1 = np.array([0,0]).reshape(2,1)

t = H-1
while (t>=0):
    v_t = R+ np.dot(P,v_t1)
    t = t-1
    v_t1 = v_t

print('simple method',v_t)
```

```
## simple method [[13.35937498]
##  [12.73437504]]
```

```
while (t>=0):
    v_t = R+P*v_t1
    t = t-1
print('backward induction',v_t)
```

```
## backward induction [[13.35937498]
##  [12.73437504]]
```