# Reinforcement

July 22, 2020

### Probability Mass Function Generator

```
In [1]: def randomizer(a, probabilities):
            import numpy as np
            cumulative=[]
            num_classes=len(a)
            i=0
            while i<num_classes:
                sum=0
                for j in range(i+1):
                    sum+=probabilities[j]
                cumulative.append(sum)
                i+=1
            if cumulative[num_classes-1]>1 or cumulative[num_classes-1]<1:
                print("error")
                return -1
            cumulative.insert(0,0.)
            number=np.random.uniform(0.,1.)
            i=1
            while i<=(num_classes):
                if number>=cumulative[i-1] and number<cumulative[i]:
                    return(a[i-1])
                i+=1
```

#### 0.0.1 Markov Process

```
In [2]: def markov_process(S_vec, states, Ptr, states_req):
            import numpy as np
            import tensorflow_core as tf
            for i in range(states_req):
                print("State at Time t={}: {}".format(i,np.argmax(S_vec)))
                Q_vec=np.matmul(S_vec,Ptr)
                S_vec=tf.keras.utils.to_categorical(randomizer(states, Q_vec),len(S_vec))
```

**For 3 states**

```
In [3]: import numpy as np
        Ptr=np.array([
            [0.1,0.4,0.5],
            [0.3,0.2,0.5],
            [0.5,0.3,0.2]
        ])
        S_vec=np.array([1.,0.,0.])
        states=[0,1,2]
        states_req=10

        markov_process(S_vec, states, Ptr, states_req)
```

```
State at Time t=0: 0
State at Time t=1: 2
State at Time t=2: 2
State at Time t=3: 0
State at Time t=4: 1
State at Time t=5: 0
State at Time t=6: 2
State at Time t=7: 0
State at Time t=8: 2
State at Time t=9: 0
```

**For 4 states**

```
In [4]: import numpy as np
        Ptr=np.array([
            [0.1,0.3,0.2,0.4],
            [0.3,0.2,0.2,0.3],
            [0.2,0.3,0.2,0.3],
            [0.1,0.2,0.4,0.3]
        ])
        S_vec=np.array([1.,0.,0.,0.])
        states=[0,1,2,3]
        states_req=10

        markov_process(S_vec, states, Ptr, states_req)
```

```
State at Time t=0: 0
State at Time t=1: 3
State at Time t=2: 2
State at Time t=3: 1
State at Time t=4: 0
State at Time t=5: 1
State at Time t=6: 2
State at Time t=7: 1
```

```
State at Time t=8: 0
State at Time t=9: 3
```

### 0.0.2 Markov Reward Process

```
In [5]: def MRP(S_vec, reward, states, Ptr, states_req):
            import numpy as np
            import tensorflow_core as tf
            for i in range(states_req):
                print("State at Time t={}: {}, Reward gained= {}".format(i,np.argmax(S_vec),rew
                Q_vec=np.matmul(S_vec,Ptr)
                S_vec=tf.keras.utils.to_categorical(randomizer(states, Q_vec),len(S_vec))
```

**For 3 states**

```
In [6]: import numpy as np
        Ptr=np.array([
            [0.1,0.4,0.5],
            [0.3,0.2,0.5],
            [0.5,0.3,0.2]
        ])
        R=np.array([2,3,5])
        S_vec=np.array([1.,0.,0.])
        states=[0,1,2]
        states_req=10

        MRP(S_vec, R, states, Ptr, states_req)
```

```
State at Time t=0: 0, Reward gained= 2
State at Time t=1: 0, Reward gained= 2
State at Time t=2: 2, Reward gained= 5
State at Time t=3: 0, Reward gained= 2
State at Time t=4: 1, Reward gained= 3
State at Time t=5: 0, Reward gained= 2
State at Time t=6: 2, Reward gained= 5
State at Time t=7: 2, Reward gained= 5
State at Time t=8: 2, Reward gained= 5
State at Time t=9: 0, Reward gained= 2
```

**For 4 States**

```
In [7]: import numpy as np
        Ptr=np.array([
            [0.1,0.3,0.2,0.4],
            [0.3,0.2,0.2,0.3],
```

```
            [0.2,0.3,0.2,0.3],
            [0.1,0.2,0.4,0.3]
        ])
        R=np.array([2,3,5,4])
        S_vec=np.array([1.,0.,0.,0.])
        states=[0,1,2,3]
        states_req=10

        MRP(S_vec, R, states, Ptr, states_req)
```

```
State at Time t=0: 0, Reward gained= 2
State at Time t=1: 2, Reward gained= 5
State at Time t=2: 1, Reward gained= 3
State at Time t=3: 0, Reward gained= 2
State at Time t=4: 2, Reward gained= 5
State at Time t=5: 3, Reward gained= 4
State at Time t=6: 0, Reward gained= 2
State at Time t=7: 1, Reward gained= 3
State at Time t=8: 1, Reward gained= 3
State at Time t=9: 3, Reward gained= 4
```

In [ ]: