The Simple Gridworld Problem

```
import numpy as np

P = np.array([
    # up
    [[1, 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, 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, 1],
     [1, 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, 1, 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, 1, 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, 1, 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, 1, 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, 1, 0, 0, 0, 0],
     [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]],

    # down
    [[0, 0, 0, 0, 1, 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, 1, 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, 1, 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, 1, 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, 1, 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, 1],
     [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, 1, 0],
     [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]],

    # left
    [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
     [1, 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, 1, 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, 1, 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, 1, 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, 1, 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, 1, 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, 1, 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, 1]],

    # right
    [[0, 1, 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, 1, 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, 1, 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, 1, 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, 1, 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, 1, 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, 1, 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, 1],
     [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]]], dtype='float32')

P.shape
```

```
(4, 15, 15)
```

```
P = P.transpose(1, 0, 2)
```

```
P.shape
```

```
(15, 4, 15)
```

```
R = np.array([
    [-1, -1, -1,  1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  0], # up
    [-1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  1, -1, -1,  0], # down
    [ 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  0], # left
    [-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,  1,  0]], #right
    dtype='float32')
```

```
R.shape
```

```
(4, 15)
```

```
R = R.transpose()
```

```
R.shape
```

```
(15, 4)
```

```
pi = np.ones((15, 4), dtype='float32') * 0.25
```

```
pi.shape
```

```
(15, 4)
```

In summary,

```
def policy_eval(P, R, pi, maxiter=30):
    V = np.zeros((maxiter, 15), dtype='float32')

    for i in range(maxiter-1):
        V[i+1] = np.squeeze(
            np.matmul(
                np.expand_dims( pi, 1 ),
                np.expand_dims( R + 0.6 * np.dot(P, V[i]), 2 )))

    return V[maxiter-1]

def policy_upd(P, R, v):
    print(np.squeeze(np.expand_dims( R + 0.6 * np.dot(P, v), 2 )))
    a_idx = np.argmax(np.squeeze(np.expand_dims( R + 0.6 * np.dot(P, v), 2 )), axis=1)
    pi = np.zeros((15, 4), dtype='float32')
    pi[range(15), a_idx] = 1.
    return pi

pi_old = None
pi = np.ones((15, 4), dtype='float32') * 0.25

while not np.all(np.equal(pi_old, pi)):
    pi_old = pi.copy()
    v = policy_eval(P, R, pi)
    pi = policy_upd(P, R, v)

print(pi)
```

```
[[-1.8125086 -2.258132   1.        -2.346084 ]
 [-2.346084  -2.3812647 -1.8125086 -2.434036 ]
 [-2.434036  -2.346084  -2.346084  -2.434036 ]
 [ 1.        -2.346084  -1.8125086 -2.258132 ]
 [-1.8125086 -2.3812647 -1.8125086 -2.3812647]
 [-2.346084  -2.258132  -2.258132  -2.346084 ]
 [-2.434036  -1.8125086 -2.3812647 -2.346084 ]
 [-1.8125086 -2.434036  -2.346084  -2.3812647]
 [-2.258132  -2.346084  -2.346084  -2.258132 ]
 [-2.3812647 -1.8125086 -2.3812647 -1.8125086]
 [-2.346084   1.        -2.258132  -1.8125086]
 [-2.346084  -2.434036  -2.434036  -2.346084 ]
 [-2.3812647 -2.346084  -2.434036  -1.8125086]
 [-2.258132  -1.8125086 -2.346084   1.       ]
 [ 0.         0.         0.         0.       ]]
[[-0.39999998 -1.24        1.         -1.24      ]
 [-1.24       -1.744      -0.39999998 -1.744     ]
 [-1.744      -1.24       -1.24       -1.744     ]
 [ 1.         -1.24       -0.39999998 -1.24      ]
 [-0.39999998 -1.744      -0.39999998 -1.744     ]
 [-1.24       -1.24       -1.24       -1.24      ]
 [-1.744      -0.39999998 -1.744      -1.24      ]
```

The Simple Gridworld Problem

In summary,

```
 [-0.39999998 -1.744      -1.24       -1.744     ]
 [-1.24       -1.24       -1.24       -1.24      ]
 [-1.744      -0.39999998 -1.744      -0.39999998]
 [-1.24        1.         -1.24       -0.39999998]
 [-1.24       -1.744      -1.744      -1.24      ]
 [-1.744      -1.24       -1.744      -0.39999998]
 [-1.24       -0.39999998 -1.24        1.        ]
 [ 0.          0.          0.          0.        ]]
[[-0.39999998 -1.24        1.         -1.24      ]
 [-1.24       -1.744      -0.39999998 -1.744     ]
 [-1.744      -1.24       -1.24       -1.744     ]
 [ 1.         -1.24       -0.39999998 -1.24      ]
 [-0.39999998 -1.744      -0.39999998 -1.744     ]
 [-1.24       -1.24       -1.24       -1.24      ]
 [-1.744      -0.39999998 -1.744      -1.24      ]
 [-0.39999998 -1.744      -1.24       -1.744     ]
 [-1.24       -1.24       -1.24       -1.24      ]
 [-1.744      -0.39999998 -1.744      -0.39999998]
 [-1.24        1.         -1.24       -0.39999998]
 [-1.24       -1.744      -1.744      -1.24      ]
 [-1.744      -1.24       -1.744      -0.39999998]
 [-1.24       -0.39999998 -1.24        1.        ]
 [ 0.          0.          0.          0.        ]]
[[0. 0. 1. 0.]
 [0. 0. 1. 0.]
 [0. 1. 0. 0.]
 [1. 0. 0. 0.]
 [1. 0. 0. 0.]
 [1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [1. 0. 0. 0.]
 [1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 1. 0. 0.]
 [1. 0. 0. 0.]
 [0. 0. 0. 1.]
 [0. 0. 0. 1.]
 [1. 0. 0. 0.]]
```