Part 1 - Dynamic Progamming

November 11, 2018

```
In [9]: %load_ext autoreload
        import numpy as np
        import dMDP as MDP
        %autoreload 2
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

0.1 Defining MDP parameters

```
In [110]: n_state = 3
          n_action = 3

          dyn = np.zeros((n_state, n_state, n_action))
          reward = np.zeros((n_state, n_action))

          gamma = 0.95
          tol = 0.01
```

```
In [111]: # for i in range(n_state):
          #     for j in range(n_state):
          #         for k in range(n_action):
          #             prob = input(f"Probability of transition from state {i} to state {j} f
          #             dyn[i,j,k] = prob

          # for i in range(n_state):
          #         for k in range(n_action):
          #             rew = input(f"Reward of action a{k} from state {i}: ")
          #             reward[i,k] = rew
```

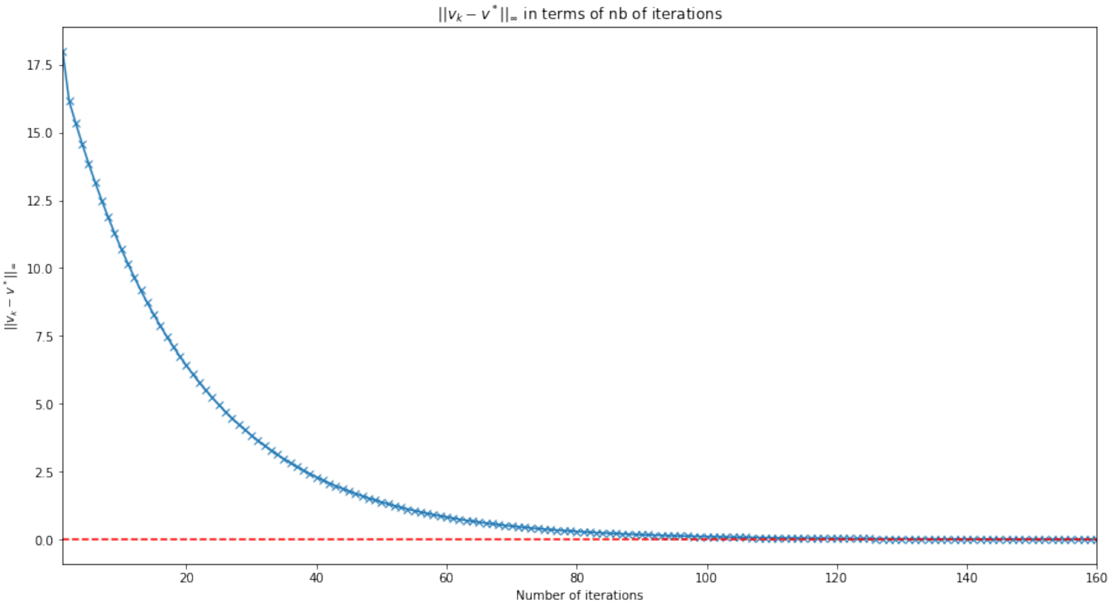
```
In [112]: reward = np.asarray([
                 [ 0.   ,  0.   ,  0.05],
                 [ 0.   ,  0.   ,  0.  ],
                 [ 0.   ,  1.   ,  0.9 ]])
```

```
In [113]: dyn = np.asarray([
                 [[ 0.55,  0.3 ,  1.  ],
                  [ 0.45,  0.7 ,  0.  ],
                  [ 0.  ,  0.  ,  0.  ]],

                 [[ 1.  ,  0.  ,  0.  ],
                  [ 0.  ,  0.4 ,  1.  ],
                  [ 0.  ,  0.6 ,  0.  ]],

                 [[ 0.  ,  0.  ,  0.  ],
                  [ 1.  ,  0.6 ,  0.  ],
                  [ 0.  ,  0.4 ,  1.  ]]])
```

0.2 Value iteration

```
In [116]: model = MDP.dMDP(n_state, n_action, dyn, reward, gamma, tol, solver = 'VI')

          value_init = np.zeros(n_state)
          model.solve(value_init = value_init)
```

```
In [119]: model.plot_error()
```



```
In [121]: model.valuef_opt_
```

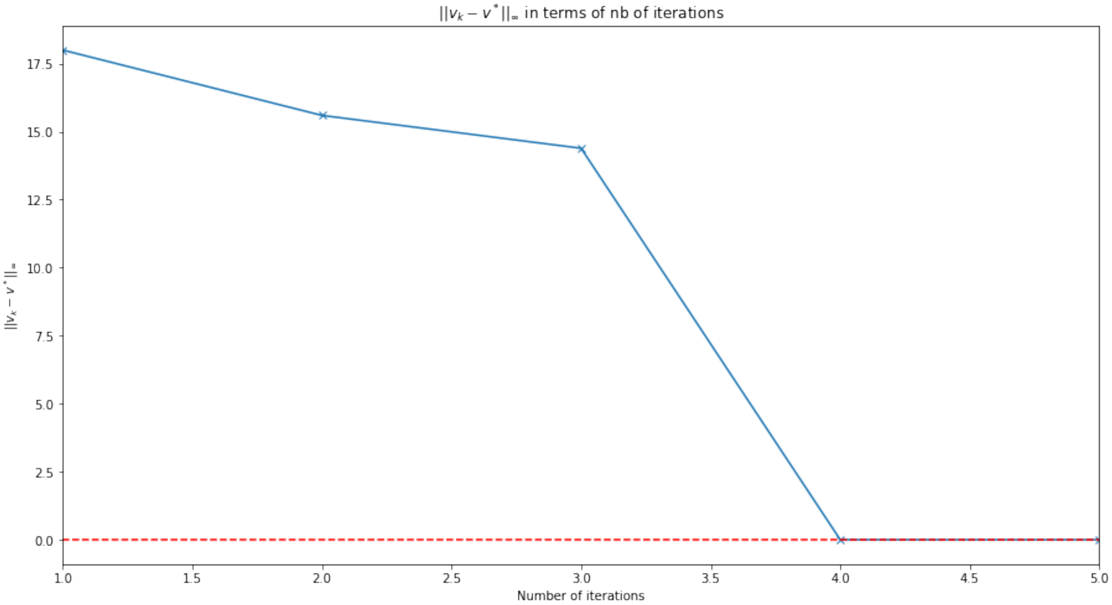
```
Out[121]: array([ 15.39115723,  16.5483871 ,  18.        ])
```

0.3 Policy iteration

```
In [106]: model = MDP.dMDP(n_state, n_action, dyn, reward, gamma, tol, solver = 'PI')
```

```
           policy_init = np.zeros(n_state, dtype = int)
           model.solve(policy_init = policy_init)
```

```
In [107]: model.plot_error()
```



```
In [108]: model.policy_

Out[108]: array([1, 1, 2], dtype=int64)

In [109]: model.valuef_opt_

Out[109]: array([ 15.39115723,  16.5483871 ,  18.        ])
```