**Vivekanand Education Society’s Institute of Technology**

**Department of AIDS Engineering**



**Subject: Reinforcement Learning**

**Class: D16AD**

| **ROLL NO: 30** | **NAME: SUHANEE KANDALKAR** | | |
| --- | --- | --- | --- |
| **EXP NO: 06** | T**ITLE: Policy Evaluation and Iteration** | | |
| **DOP:** |  | **DOS: 22/03/25** |  |
| **GRADES:** | **LOs MAPPED:** | | **SIGNATURE:** |

## 

## Aim**:**

Implementing a basic grid-world environment as an MDP and applying policy evaluation and policy iteration on it.

Theory**:**A **Markov Decision Process (MDP)** provides a mathematical framework for modeling sequential decision-making problems where outcomes are partly random and partly under the control of an agent. It is defined by:

1. **States (S):** The set of all possible situations in which an agent can be.
2. **Actions (A):** The set of possible actions an agent can take.
3. **Transition Probabilities (P):** The probability of reaching a new state given the current state and action.
4. **Rewards (R):** The immediate reward received after transitioning from one state to another.
5. **Policy (π):** A strategy that defines the action to take in each state.

A common application of MDPs is the **grid-world environment**, where an agent navigates within a grid to reach a goal while encountering obstacles and rewards.

[Markov Decision Process - Wikipedia](https://en.wikipedia.org/wiki/Markov_decision_process)

## **Grid-World Environment:**

The **grid-world** is a simplified environment where an agent moves across a 2D grid. Each cell represents a state, and the agent can take actions such as moving **up, down, left, or right**. The objective is to reach a goal while maximizing cumulative rewards.

### Components of Grid-World as an MDP:

* **States (S):** Each cell in the grid.
* **Actions (A):** {Up, Down, Left, Right}.
* **Transition Probabilities (P):** A deterministic or probabilistic movement between states.
* **Rewards (R):** Positive rewards for reaching the goal, negative rewards for obstacles or certain states.

Example: A **4x4 grid-world** where the agent starts at the bottom-left and tries to reach the top-right goal state.

[REINFORCEjs: Gridworld with Dynamic Programming](https://cs.stanford.edu/people/karpathy/reinforcejs/gridworld_dp.html)

## **Policy Evaluation:**

**Policy Evaluation** is the process of determining the value function for a given policy. The **value function (Vπ(s))** represents the expected cumulative reward an agent receives when starting from state sss and following policy π\piπ thereafter.

### Bellman Equation for Policy Evaluation:

Vπ(s)=∑s′P(s′∣s,π(s))[R(s,π(s),s′)+γVπ(s′)]V\_{\pi}(s) = \sum\_{s'} P(s'|s, \pi(s)) [R(s, \pi(s), s') + \gamma V\_{\pi}(s')]Vπ​(s)=s′∑​P(s′∣s,π(s))[R(s,π(s),s′)+γVπ​(s′)]

where:

* Vπ(s)V\_{\pi}(s)Vπ​(s) is the value function for policy π\piπ.
* P(s′∣s,π(s))P(s'|s, \pi(s))P(s′∣s,π(s)) is the transition probability.
* R(s,π(s),s′)R(s, \pi(s), s')R(s,π(s),s′) is the reward.
* γ\gammaγ is the discount factor (0 ≤ γ\gammaγ ≤ 1).

Policy Evaluation involves iteratively updating Vπ(s)V\_{\pi}(s)Vπ​(s) until convergence.

Iterative Policy Evaluation - Medium

## **Policy Iteration:**

**Policy Iteration** is a method used to determine the optimal policy by alternating between two steps:

1. **Policy Evaluation:** Compute Vπ(s)V\_{\pi}(s)Vπ​(s) for a given policy.
2. **Policy Improvement:** Improve the policy by choosing actions that maximize rewards.

### Steps in Policy Iteration:

1. Start with an initial random policy π\piπ.
2. Perform **policy evaluation** to compute Vπ(s)V\_{\pi}(s)Vπ​(s).
3. Update the policy by selecting the best action based on Vπ(s)V\_{\pi}(s)Vπ​(s):  
    π′(s)=arg⁡max⁡a∑s′P(s′∣s,a)[R(s,a,s′)+γVπ(s′)]\pi'(s) = \arg\max\_a \sum\_{s'} P(s'|s,a) [R(s,a,s') + \gamma V\_{\pi}(s')]π′(s)=argamax​s′∑​P(s′∣s,a)[R(s,a,s′)+γVπ​(s′)]
4. Repeat until the policy converges to the optimal policy π∗\pi^\*π∗.

[Policy Iteration - RL Notes](https://gibberblot.github.io/rl-notes/single-agent/policy-iteration.html)

## **Practical Implementation:**

For hands-on experience, implementing these concepts in Python or other programming languages is beneficial. The following guide provides a step-by-step explanation of **policy and value iteration** in a grid-world environment.

[Navigating in Gridworld using Policy and Value Iteration](https://www.datascienceblog.net/post/reinforcement-learning/mdps_dynamic_programming/)

Code:

[RL\_Exp6\_30.ipynb](https://colab.research.google.com/drive/19mZlLyEeEah2wf6R-xgkJZUbsqTf5s6r?authuser=0#scrollTo=wkAQ1Tenc7Tm)

# Conclusion :

By implementing **policy evaluation** and **policy iteration** in a **grid-world environment**, we can determine an optimal policy that maximizes cumulative rewards. These methods are fundamental in reinforcement learning and serve as a foundation for more advanced techniques such as **value iteration** and **Q-learning**.