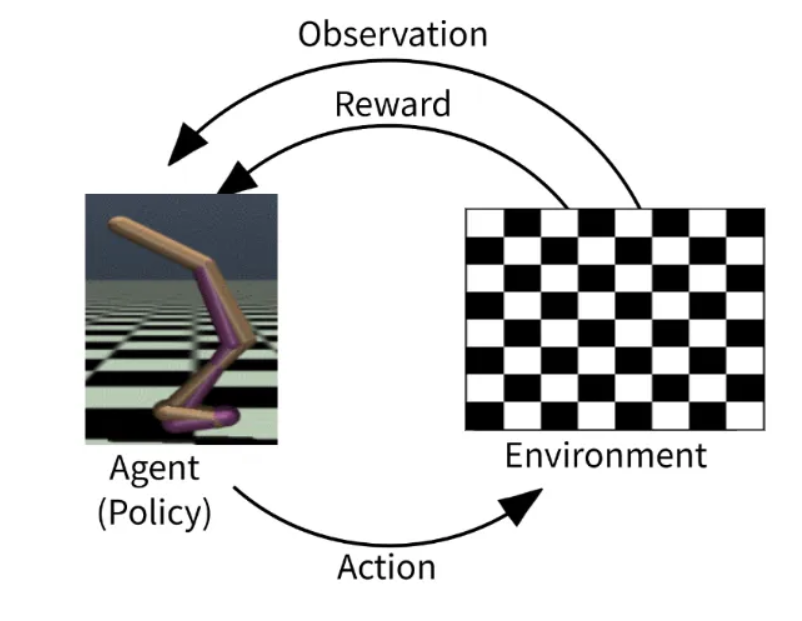
**Policy Gradient (Reinforce).**

**Action-value Reinforcement Learning method.**

**“Policy gradient methods are a type of reinforcement learning techniques that rely upon optimizing parametrized policies with respect to the expected return (long-term cumulative reward) by gradient descent.”**

A simulated robot must explore the unknown environment with an infinitely large number of actions it can take to find which actions yield the best results to keep moving forward.



The goal of the simulated robot is to find **optimal actions** for a given state in an **environment** to **maximize the** **long-term reward of moving forward**. The goal can be achieved using an optimized policy that maps the robot's optimal actions for the different environmental states.

**The challenge for the robot is to choose actions from an infinitely large action space in an infinitely large state space**

## One way to solve the problem is to use Value-based or Action-value methods.

The Value-based or Action-value RL method aims to find an optimal strategy(policy) for the robot to maximize its reward in an environment by repeatedly evaluating the expected reward of each action and choosing the one that leads to the highest reward in the long run.

Value-based methods use a value function to estimate each state or state-action pair's expected return or cumulative reward. A policy is then derived from the value function by choosing the action with the highest estimated value for each state.

***Value-based RL like***[***Q-Learning***](https://towardsdev.com/reinforcement-learning-q-learning-38146880ca49)***,***[***SARSA***](https://arshren.medium.com/reinforcement-learning-sarsa-and-q-learning-e11ebe87dca9)***, or***[***DQN***](https://arshren.medium.com/deep-q-learning-a-deep-reinforcement-learning-algorithm-f1366cf1b53d)***are best suited when the action space is discrete and not too large.***

## what happens in the continuous control domain, where actions are continuous and often high-dimensional, as in the case of the simulated Robot?

***Value-based RL algorithms operate only in discrete state and action spaces***

Making a simulated robot move forward with continuous states and actions can be challenging using value-based algorithms as they typically require discretizing the state and action spaces to estimate the value function for each state or state-action pair.

***Discretization of high dimensional continuous state and action spaces can be diffucult, as it requires defining a finite set of states and actions, which can lead to information loss and decreased accuracy.***

[Deep Q-Learning (DQL)](https://arshren.medium.com/deep-q-learning-a-deep-reinforcement-learning-algorithm-f1366cf1b53d) can solve problems with high-dimensional observation spaces and only handle discrete and low-dimensional action spaces. Large action spaces are difficult to explore efficiently, and thus successfully training using DQN is likely intractable.

DQL has a deterministic policy, as the policy is implicitly defined by selecting the action with the highest estimated value (maximum Q-value) for a given state.

A deterministic policy will always produce the same action. for a given state. This means that the behavior of the agent is fully determined by the policy.

***Value-based methods are also sample-inefficient in continuous state and action spaces requiring many experiences to accurately estimate the value function for each state or state-action pair making it difficult to converge towards an optimal policy in a reasonable amount of time.***

## so, can we learn a policy that can select actions without consulting a value function and is sample efficient?

**Policy based methods learn a parameterized policy that can select actions without consulting a value function**

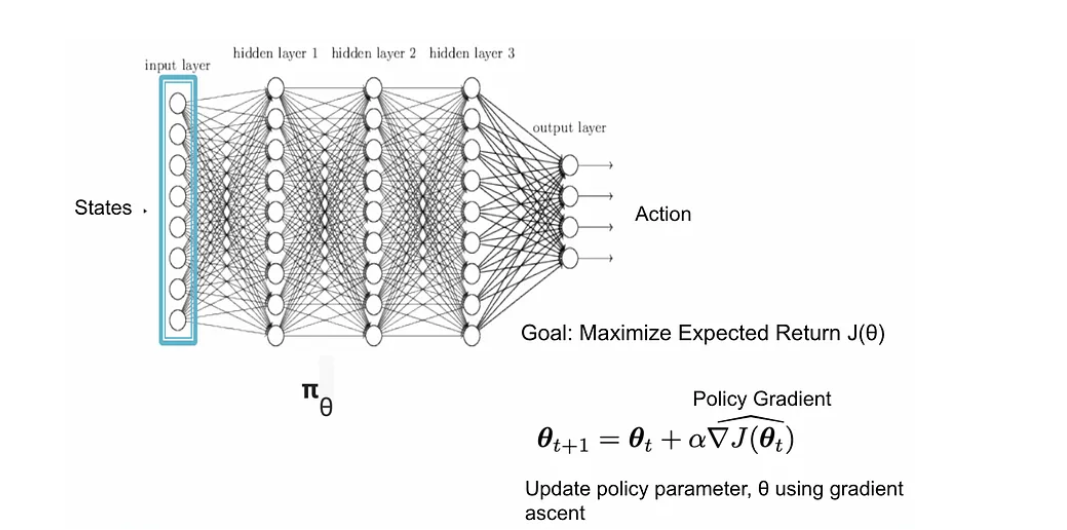
Policy-based methods define a **Policy Network** that implements the agent, which takes the state representation of the game as inputs and outputs the probabilities of selecting each action.

Policy-based methods are more sample efficient than Value-based methods for continuous control tasks.

***Policy-based methods learn a direct mapping from states to actions allowing the agent to make better use of the experience it collects***

***Policy-based methods do not need to estimate a value function that predicts the expected reward for each action.***

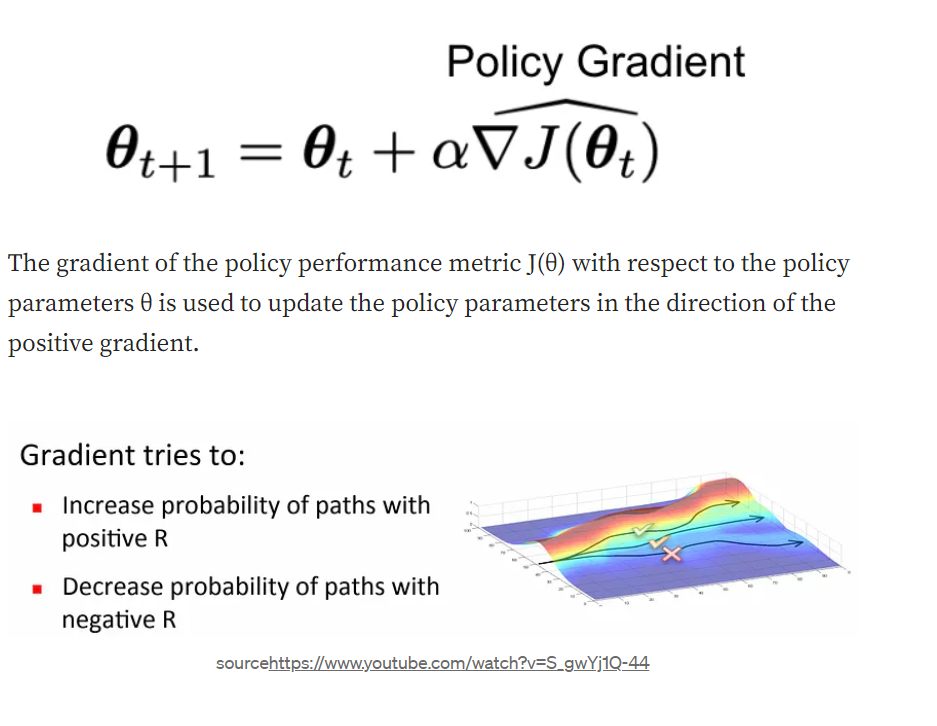
***The policy network is trained using Gradient Ascent, a gradient-based optimization techniques that outputs a probability distribution over the possible actions based on a stochastic policy, rather than a deterministic action.***



Goal of stochastic gradient ascent(SGD) used in Policy gradient RL is to increase the expected reward by adjusting the policy network parameters, which determines the action to be taken in a given state

SGD is called stochastic because the gradient is estimated using a random sample of environmental experiences.

The gradient of the policy is computed with respect to the policy parameters and is used to update the policy in the direction that increases the expected reward.



Goal of policy gradient RL is to learn a stochastic policy that maximizes the expected cumulative reward.

Data sample efficiency refers to the amount of experience or data that is needed to learn a good policy. The policy gradient learns a stochastic policy that balances the exploitation and exploration of the environment leading to more efficient data samples.

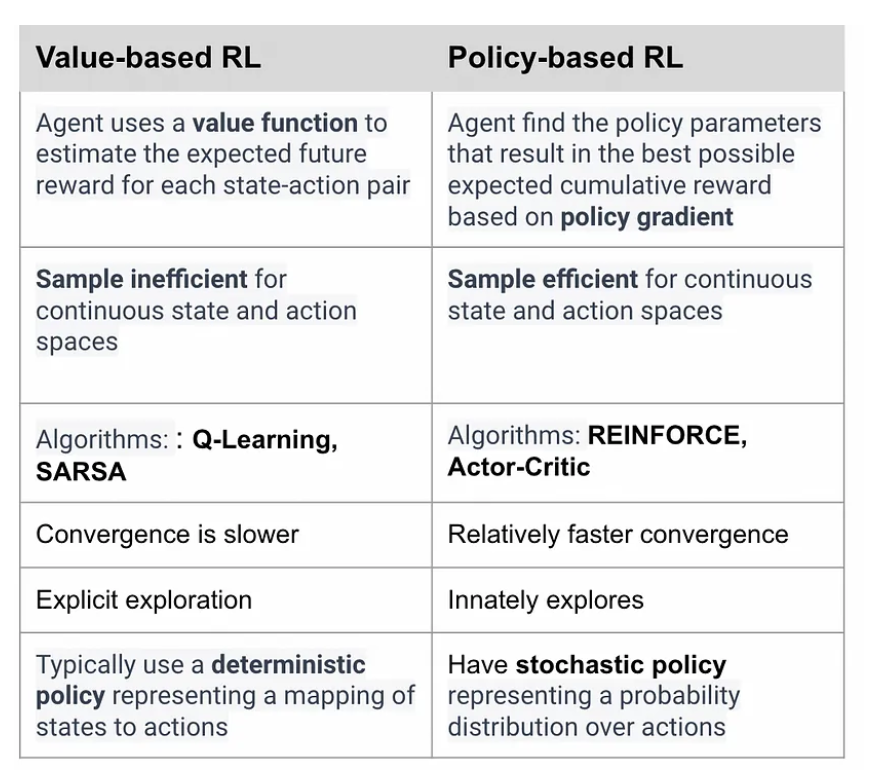
The stochastic policy maps each state to a probability distribution over the set of possible actions; this makes the Agent’s behavior probabilistic, allowing it to explore different actions in a given state, leading to better performance in complex environments.

policy gradient RL is a method for training the agent to make decisions by adjusting the parameters of its policy, which maps states to actions.

## How does the Policy Gradient adjust the policy to maximize the expected cumulative reward?

The policy gradient uses the policy gradient theorem to adjust the policy parameters to increase the expected cumulative reward.

The Policy Gradient Theorem guides how to adjust the policy parameters to increase the likelihood of taking the optimal action in each situation, in order to maximize the reward.

 The robot using a stochastic policy will move randomly based on the probabilities in the distribution. This means that even if the robot encounters the same obstacle, it may behave differently each time, depending on the probabilities generated by the policy.

**Pros**

* Better convergence properties
* **Effective in high-dimensional or continuous action spaces**
* Can learn stochastic policies

## **Conclusion:**

The robotic arm trained using policy-based methods based on the policy gradient theorem will generate a direct mapping from states to actions. This mapping, called the policy, is updated by adjusting its parameters based on the results of the actions taken by the arm. The policy gradient theorem calculates the gradient, or slope, of the policy’s performance with respect to its parameters. The policy is then updated in the direction of this gradient to increase the expected reward. **This continuous improvement allows the robotic arm to smoothly and efficiently navigate its environment, gradually discovering the actions that lead to the highest expected reward.**