**Tune alpha, gamma, and/or epsilon using a decay over episodes**

**-Jemin Kachhadiya**

* **alpha** is the **learning rate,**
* **gamma** is the **discount factor**. It quantifies how much importance we give to future rewards. It’s also handy to approximate the noise in future rewards. Gamma varies from 0 to 1. If Gamma is closer to zero, the agent will tend to consider only immediate rewards. If Gamma is closer to one, the agent will consider future rewards with greater weight, willing to delay the reward.
* **Epsilon** is **exploration and exploitation.** As the agent begins the learning, we would want it to take random actions to explore more paths. But as the agent gets better, the Q-function converges to more consistent Q-values. Now we would like our agent to exploit paths with the highest Q-value i.e, take greedy actions. This is where epsilon comes in.

The Agent experiments with various "state-action" pairings until it either succeeds or falls into the pit. Each of these explorations will be referred to as an episode. Every time an agent completes a goal or is terminated, the next episode begins.

**Learning rate decay**

Learning rate is how big you take a leap in finding an optimal policy. In the terms of simple Q-Learning, it's how much we are updating the Q value with each step.

[enter image description here](https://i.stack.imgur.com/un5Wj.png)

Higher **alpha** means we are updating your Q values in big steps. When the agent is learning we should decay this to stabilize our model output which eventually converges into an optimal policy.

**Epsilon Decay**

Epsilon is used when we are selecting specific actions based on the Q values we already have. As an example, if we select the pure greedy method (epsilon = 0) then we are always selecting the highest q value among all the q values for a specific state. This causes an issue in exploration as we can get stuck easily at a local optimum.

Therefore, we introduce randomness using epsilon. As an example, if epsilon = 0.3 then we are selecting random actions with 0.3 probability regardless of the actual q value.

In conclusion, the learning rate is associated with how big we take a leap, and epsilon is associated with how randomly we take an action. As the learning goes on both should decay to stabilize and exploit the learned policy which converges into an optimal one.

Code in google Colab : <https://colab.research.google.com/drive/1e6ZHwnUAe65tETE6kPu5BnG7Dh7gNVGJ?usp=sharing>