Reinforcement learning - Q-learning algorithm

Objectif

- Implémenter l'algorithme Q-learning en utilisant la bibliothèque Gym (pour la mise en place de l'environnement).

Le problème

Voir la vidéo : https://youtu.be/XiigTGKZfks

Gym implementation :

A pole is attached by an un-actuated joint to a cart, which moves along a frictionless track. The system is controlled by applying a force of +1 or -1 to the cart. The pendulum starts upright, and the goal is to prevent it from falling over. A reward of +1 is provided for every timestep that the pole remains upright. The episode ends when the pole is more than 15 degrees from vertical, or the cart moves more than 2.4 units from the center.

More info: https://github.com/openai/gym/blob/master/gym/envs/classic_control/cartpole.py

Getting started - Requirements

- Python 3.5+
- numpy
    - pip3 install numpy
- Gym
    - pip3 install gym

Steps

1. Prenez le temps pour comprendre le code fourni. En particulier, la méthode *discretize*.
2. Complétez les TODOs dans le code :
    1. Définition du tableau Q
    2. Implémentation de la fonction epsilon-greedy
    3. Implémentation de l'Equation de Bellman
3. Exécutez le code. Combien d'épisodes sont nécessaires pour converger ?
4. (**Avancé**) Appliquez l'algorithme Q-learning à l'environnement gym 'MountainCar-v0'. Voir :

- https://gym.openai.com/envs/MountainCar-v0/
- https://github.com/openai/gym/wiki/MountainCar-v0

Ou l'environnement LunarLander-v2 (harder !!) Voir :

- http://gym.openai.com/envs/LunarLander-v2/
- https://github.com/openai/gym/blob/master/gym/envs/box2d/lunar_lander.py

Comment : Créez un autre script et adaptez les différentes méthodes. En particulier, la méthode *discretize* et la taille du tableau Q.