Assignment 2 Report:

1. **Define Problem**

In a binomial model of a single stock with non-zero interest rate, we use Q-Learning and Policy Gradient to train the optimal policy of hedging an ATM American put option with maturity T = 10. For the reward function, we define the negative variance between value of hedge portfolio and value of put value as our rewards.

1. **Q-Learning**

We use Q-Learning to estimate the action-value function and train the optimal policy: the optimal hedging portfolio. For Q-Learning, each update is the Q value that takes action in the current state.

For the ***State*** space, which is the stock price at each node of the binomial tree. We assume it is discrete including all discrete outcomes after each time step. For the ***Action*** space, which includes two types of holding assets at one time step: stock and cash, we discretize them into *action\_stock* from [-1,0] and into *action\_cash* from [0,K] respectively.

1. **Policy Gradient**

We use four different policy gradient methods to learn parameter and probability and train the optimal policy for the hedging portfolio:

1. Policy Gradient - Reinforce with linear preference function
2. Policy Gradient - Reinforce with Neural Network(Categorical)
3. Policy Gradient - Reinforce with Neural Network(Pathwise derivative)
4. Policy Gradient with Baseline by Neural Network

For the ***State*** space, which is the stock price at each node of the binomial tree. We assume it is discrete including all discrete outcomes after each time step. In addition, to know the value of American put for each node, dynamic programming is used and the value as well as early exercise policy is computed backward.

**Monte-Carlo Policy-Gradient**

Input: a differentiable policy parameterization

**Policy Gradient - Reinforce with linear preference function**

For the ***Action*** space, we discretize it ranging from [-1,1]. The action is sampled by the policy function , which is determined by a linear preference function according to the given state and action. Also, we set a minimum value for the probability of a sampled action.

图表, 直方图

描述已自动生成

**Policy Gradient - Reinforce with Neural Network – Categorical**

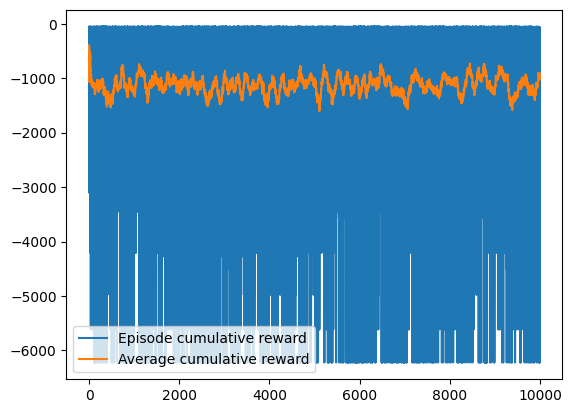
For the ***Action*** space, we try on Neural Network with one hidden layer and activation function as our approximation of the policy function . The ‘Categorical’ here means that the output will be mapped categorically for the probability distribution of each action. As a result, the ***Action*** space is still discretized.

图表

描述已自动生成

**Policy Gradient - Reinforce with Neural Network - Pathwise derivative**

For the ***Action*** space, it is another method to implement these stochastic gradient approaches using the reparameterization from the *rsample()* method, where parameterized random variables can be constructed through deterministic functions of unparameterized random variables. For here, we assume the distribution is from normal family.



**Policy-Gradient with Baseline by Neural Network**

One negative of policy gradient methods is the high variance caused by the empirical returns. A common way to reduce variance is to subtract a baseline from the returns in the policy gradient. Besides the Neural Network proposed in ‘Categorical’ is used, we add another trainable NNs without the softmax function as the approximation of baseline .

Input: a differentiable policy parameterization

Input: a differentiable state-value parameterization

图表, 直方图

描述已自动生成

**Conclusion**

According to the following figure, it is clear to find that the Policy-Gradient with Baseline trained the optimal policy with the largest average cumulative reward to hedge the option.

图表

描述已自动生成

**Reference for Categorical and Pairwise:**

https://blog.csdn.net/weixin\_42018112/article/details/90899559