Deep Q learning method in Binomial in Stock Hedging Problem under Geometric Brownian Motion

Jin Tian & Zheng Zihao

2022 April

Abstract

In this paper, we used the deep Q learning method to solve the optimal dynamic hedging problem for derivatives when there are transaction costs. We used stock price as the continuous state and assume that the asset price followed a geometric Brownian motion. The paper illustrates the approach by showing the difference between delta hedging and optimal hedging for a short position in a call option when the objective is to minimise a function equal to mean-variance return with transaction costs. Our results show that the methods we use can reduce our costs and consider more targets.

1 Introduction

Hedging is an essential problem for traders. Traders have traditionally hedged the risks associated with derivatives transactions by monitoring "Greek letters." Traders typically try to maintain delta-neutral (or close to delta-neutral) positions. Because the delta of an option changes during its life, the trader's position must be rebalanced periodically. If there are no transaction costs or other frictions, it is in theory optimal to rebalance continuously. In practice, transaction costs and other frictions (which we will collectively refer to as "trading costs") mean that the theoretically optimal strategy must be modified. Also, the position in the underlying asset is, in practice, rebalanced periodically rather than continuously. Several papers have considered the effect of this on option pricing and the efficiency of hedging. These include Leland (1985), Figlewski (1989) and Martinelli (2000).

This paper examines how the reinforcement learning approach can be used to take trading costs into account in hedging decisions. When delta hedging would require shares to be purchased, it tends to be optimal for a trader to be under-hedged relative to the delta. Similarly, when delta hedging would require shares to be sold, it tends to be optimal for a trader to be over-hedged relative to the delta. We use an objective function of the expected cost of hedging plus a constant times the standard deviation of the cost.

In this paper, we will introduce our model using the deep Q learning method in hedging problems. We first illustrate the deep hedging approach by assuming that the underlying asset follows the Geometric Brownian Motion. In this case, our reinforcement learning approach outperforms delta hedging when there are trading costs.

2 Models

First, we introduce the geometric Brownian motion of the stock price. This model assumes that stock prices obey the following movements

$$dS = \mu S dt + \sigma S dz \tag{1}$$

where μ and σ are the stock's mean return and volatility (assumed constant), and dz is a Wiener process. The work of Black and Scholes (1973) and Merton (1973) show that the price of a European

call option with maturity T and strike price K is

$$S_0 e^{-qT} N(d_1) - K e^{-rT} N(d_2) \tag{2}$$

where r is the risk-free rate, q is the dividend yield (both assumed constant), S_0 is the initial value of S and

$$d_1 = \frac{\ln(S_0/K) + (r - q + \sigma^2/2)T}{\sigma\sqrt{T}} \tag{3}$$

$$d_2 = d_1 - \sigma\sqrt{T} \tag{4}$$

We choose to solve the reinforcement learning problem using the Actor-Critic method as the method allows the hedging position to be continuous. Unlike the Q-learning method, it does not require a discrete set of hedging positions in the underlying asset to be specified. The algorithm in geometric Brownian motion depends on (1) interest rate: r (2) volatility: σ.

We use as an example the situation where a trader is hedging a short position in a call option. We assume that the trader rebalances her position at time intervals of Δt and is subject to trading costs. The life of the option is $n\Delta t$. The cost of a trade in the underlying asset in our formulation is proportional to the value of what is being bought or sold. Still, the analysis can easily be adjusted to accommodate other assumptions. The state at time $i\Delta t$ is defined by three parameters:

1. The holding of the asset during the previous period; i.e., from time $(i-1)\Delta t$ to time $i\Delta t$
2. The asset price at time $i\Delta t$
3. The time to maturity

The action at time $i\Delta t$ is the amount of the asset to be held for the next period; i.e., from time $i\Delta t$ to time $(i+1)\Delta t$.

Our rewards (negative costs) are given in the form of accounting P&L formulation, which is

$$\delta w_{t+1} = V_{i+1} - V_i + H_i(S_{i+1} - S_i) \tag{5}$$

for $0 \leqslant i < n$ where S_i is the asset price at the beginning of period i, H_i is the holding between time $i\Delta t$ and $(i+1)\Delta t$, κ is the trading cost as a proportion of the value of what is bought or sold, and V_i is the value of the derivative position at the beginning of period i. (V_i is negative in the case of a short call option position.) In addition, there is an initial reward associated with setting up the hedge equal to $-\kappa|S_0 H_0|$ and a final reward associated with liquidating the hedge at the end equal to $-\kappa|S_0 H_0|$).

In the problem we are considering, it is natural to work with costs (negative rewards). This is what we will do from now on. We assume agents have quadratic utility, which implies that their optimal portfolios are given as solutions of the mean–variance optimization problem. In this case, we define the one-period reward function as

$$R_t := \delta w_t - \frac{\kappa}{2}(\delta w_t)^2 - c|S_t(H_t - H_{t-1})| \tag{6}$$

where κ denoting an agent's risk aversion, δw_t is the individual wealth increments in period t, S_t is the asset price at the beginning of period t, H_t is the holding between time $t-1$ and t and tc is the transaction cost ratio.

In deep Q-learning, we use a neural network to approximate the Q-value function. The state is given as the input and the Q-value of all possible actions is generated as the output. DQN used the following semi-gradient form of Q-learning to update the network's weights:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha\left[R_{t+1} + \gamma \max_a \hat{q}(S_{t+1}, a, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t)\right]\nabla\hat{q}(S_t, A_t, \mathbf{w}_t) \tag{7}$$

where $\mathbf{w}_t$ is the vector of the network's weights, A_t is the action selected at time step t, and S_t and S_{t+1} are respectively the preprocessed image stacks input to the network at time steps t and $t+1$. The gradient in above equation was computed by backpropagation.

The pseudocode for deep Q-learning is as follows:

Algorithm 1 Deep Q-Learning algorithm

Initialize replay memory $\mathcal{D}$ to capacity N
Initialize action-value function Q with random weights
for episode = 1, M do
  Initialise state s_t
  for t = 1, T do
    With probability ϵ select a random action a_t, otherwise select $a_t = \max_a Q^*(s_t, a; \theta)$
    Execute action a_t and observe reward r_t and state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
    Set $s_{t+1} = s_t$
    Sample a random minibatch of transitions (s_i, a_i, r_i, s_{i+1}) from $\mathcal{D}$
    Set $y_j = \begin{cases} r_j & \text{for terminal } s_{t+1} \\ r_j + \gamma \max_{a'} Q(s_{t+1}, a'; \theta) & \text{for non-terminal } s_{t+1} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(s_t, a_j; \theta))^2$
  end for
end for

3 Empirical Results

Tables 1 compare the results from using reinforcement learning with delta hedging for short positions in at-the-money ($S_0 = K$) call options on a stock lasting one month and three months when $\mu = 5\%$, $r = 0$, $q = 0$. We set $c = 0.001$ in equation (6) so that the hedger's objective is to minimize the mean cost of hedging plus c times the transaction of the cost of hedging. The risk parameter, κ, is 10%.

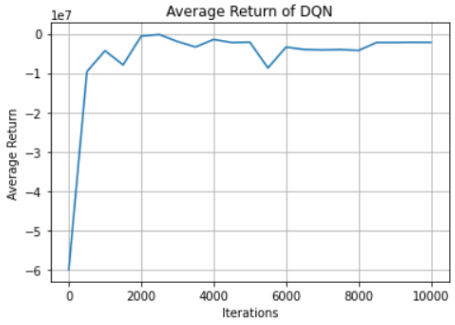

Figure 1: Convergence

	Delta	Deep Q
Weekly	838.22	704.12
4 Days	669.65	615.23
2 Days	355.69	304.52
Daily	211.17	199.81

Figure 2: Cost

The table shows that using RL optimal hedging instead of delta hedging has a positive effect on the overall cost of hedging. As we can see, it converges quickly. So, it seems very effective.

4 Conclusions

From this paper, we developed our model to use the Q-learning method to solve the problem of optimal hedging of derivatives in the presence of transaction costs. Our results show that using RL optimal hedging rather than delta hedging has a positive effect on the cost of hedging in the situations we consider and markedly reduces the mean cost of hedging.

5 Reference and Appendix

Please see https://github.com/JinTian0717/hm3