Reinforcement Learning for Optimal Execution:

Training Q-Learning Agents in a Simulated Environment

Alycia (Yingyue) Qiu

Abstract

In this project, we trained a Q-learning agent in a simulated environment that is based on the Almgren-Chriss Model. The goal of this project is train the agent to find the optimal policy to liquidate 10 shares of stock over a horizon T=5 in an unknown environment. Comparing to the two naive strategies: (1) executing with a constant trading speed, and (2) executing everything at time 0, the Q-learning agent show betters performance especially when encounters a more complicated market.

Introduction. Liquidation is the process of selling a large number of shares of one stock sequentially within a given time frame, taking into consideration the costs arising from market impact and a trader's risk-aversion[2]. This project focuses on applying Q-learning, a form of reinforcement learning, to develop a stock trading algorithm. The primary objective is to train this algorithm within a simulated trading environment, with an emphasis on demonstrating convergence over time. A key component of our analysis is the use of the Sharpe Ratio, a measure of risk-adjusted return, to evaluate the performance of the Q-learning algorithm. This performance is compared against two simpler trading strategies: one involving constant trading speed and another involving immediate liquidation of all holdings at the beginning of the trading period. Additionally, we plan to design a new simulation environment to test these strategies and the Q-learning algorithm, allowing us to assess the adaptability and effectiveness of the Q-learning approach in varied market conditions. The results of these experiments will provide insights into the capabilities and limitations of using Q-learning for stock trading simulations.

Problem Set-up. The task is to liquidate 10 shares of a hypothetical stock A over a finite time horizon T=5. Price of Stock A ranges from 100 to 110 with interval 0.1. To train the agent, we will need to provide it an environment for the learning process. Since we are not directly interacting with the real market, we define a market simulator that not only reflects the market dynamics, but also give us convenience to control the parameter and see how different strategies work in with different parameters.

Model Employed. In this project, we have employed two pivotal models to create a robust framework for our stock trading simulation: the Q-learning Algorithm and the Almgren-Chriss Model. This combination of the Q-learning Algorithm's decision-making

prowess and the market realism provided by the Almgren-Chriss Model creates a comprehensive and sophisticated approach to exploring algorithmic trading strategies.

The Almgren-Chriss Model. In the Almgren-Chriss setup, a trader needs to sell a specified amount of an asset, initially priced at S_0, over a time horizon from 0 to T. The selling occurs at regular time intervals, from $t = 1$ to T. By the end of this perriod, the trader should have sold all of the asset, leaving no remaining inventory. The objective is to figure out the best strategy for selling the asset at each time interval, denoted as $u_1, u_2, \cdots, u_T$, where u_t is the amount sold at time t. This model assumes that selling the asset affects its price in two ways: a temporary impact, which causes short-term price changes due to immediate supply and demand shifts, and permanent impact, which alters the asset's baseline for the duration for the trading period[1].

In the market simulator we defined, the environment follows the Almgren-Chriss Model. This model is specifically tailored to capture the dynamics of optimal trade execution in financial markets. It considers factors such as temporary and permanent market impacts of trades, effectively simulating how large orders can influence market prices. By integrating the Almgren-Chriss Model, our simulation environment realistically mimics market conditions, allowing the Q-learning algorithm to interact with a market that responds dynamically to trade actions.

The Almgren-Chriss Model also suggests two naive strategies for the optimal execution problem: Equal Sized Packets and Eager Agent. In later experiment, we will compare the performance of the Q-Learning agent with these two naive strategies.

Equal Sized Packets. The first strategy involves executing trades at a constant speed, characterized by equal-sized packets. In this approach, each trade amount n_k is determined as $\frac{X}{N}$, resulting in the remaining inventory x_k being $(N - K)\frac{X}{N}$ for any $1 \leq K \leq N$. This method ensures a uniform distribution of trades over time.

Eager Agent. The second strategy is an 'Eager Agent' approach, where the entire position is executed immediately at time 0. Here, n_1 is equal to X, the total inventory, and all subsequent trades from n_2 to n_N are zero, resulting in x_1 and x_N also being zero. This strategy represents a scenario where the trader liquidates their position as quickly as possible.

Reinforcement Learning in Finance. resulting in the remaining inventory Reinforcement learning (RL) in the financial context involves a trader or investor operating within an environment that may be unknown to them. In this setting, the trader takes actions (a_t) which which influence their future state (s_{t+1}) in the market, considering factors like market impact or price impact. The success of these actions is measured by a reward signal (r_t), which could be based on return or risk-adjusted return. The overarching goal of RL in this context is to select actions that maximize future rewards, essentially focusing on maximizing profit.

Markov Decision Processes(MDP). A Markov Decision Process (MDP) in the context of financial trading is defined by a tuple consisting of several key elements: S, the set of states, which is finite and has a size denoted as $|S|$; s_1, the initial state which is a member of the set S; and A, the set of actions, also finite with its size represented as $|A|$. The process advances in discrete time intervals or rounds, marked as $t = 1, 2, \ldots, H$, starting from the initial state s_1. In each round t, the agent observes the current state s_t from set S, takes an action a_t from set A, and then moves to the next state s_{t+1}, also in S, while receiving feedback in terms of a reward signal r_t from set R. This framework can be applied to financial trading scenarios, where the state variable s might include factors like mid-price, current inventory, and other market conditions, the action a could be the number of shares to liquidate at a specific time, and the reward r could be based on various metrics such as cash inflow, implementation shortfall, Sharpe ratio, return, or profit and loss.

Finite Horizon Markov Decision Processes. Optimal execution as a reinforcement learning problem falls into the realm of the Finite Horizon Markov Decision Processes.

$$V_t^*(s) = \sup_{\Pi} V_t^{\Pi}(s) := \sup_{\Pi} E^{\Pi}\left[\sum_{u=t}^{T-1} r_u(s_u, a_u) + r_T(s_T)|s_t = s\right]$$

subject to

$$s_{u+1} \sim P_u(s_u, a_u),\ a_u \sim \pi_u(s_u),\ t \leq u \leq T-1$$

Let $Pu : S \times A \to P(S)$ denote the transition kernel at time u. The reward at time u is represente by $r_u(s, a)$, a real-valued random variable. The terminal reward is given by $r_T(s)$,also a real-valued random vairiable. Finally, we consider a randomized Markovian policy $\Pi = \{\pi_u\}_{u=0}^{T}$: here $\pi_u : S \to P(A)$ at time u.

Q-Learning and Markov Desicion Processes. The Q-learning Algorithm, grounded in the principles of Markov Decision Processes (MDP), serves as the backbone of our trading strategy. This reinforcement learning technique is adept at making sequential decisions, learning optimal actions based on rewards and punishments in a given state of the market. It operates under the premise that future rewards are influenced by current actions, a characteristic of MDPs, thus enabling the algorithm to adapt and optimize its strategy over time.

Parameter Selection for First Experiment. Since the Q-learning training code is given, I didn't change the parameter in the environment setup for the first experiment. In the first experiment, the trained agent operates under a default environment setting characterized by specific parameters. The coefficient for temporary market impact, denoted by β, is set at 0.01. This coefficient plays a crucial role in determining how the agent's actions influence the immediate market conditions. Additionally, the degree of market impact, represented by d, is fixed at 2, indicating the extent to which the agent's actions affect the market. Finally, the standard deviation of market noise, symbolized by σ, is

established at 0.5. This value signifies the level of randomness or unpredictability inherent in the market, which the agent must contend with during its decision-making process. These parameters collectively define the environmental conditions in which the agent is trained and tested, shaping its interactions and strategies within the simulated market scenario.

Convergence for Q-Learning in the First Experiment. The Q-learning Algorithm converges around episodes 300. From the graph, we can see the loss values are very large to begin with. But later, the algorithm quickly learns a substantial portion of the policy, resulting in a rapid decrease in loss, which then levels off once most of the learning is completed, proving convergence.

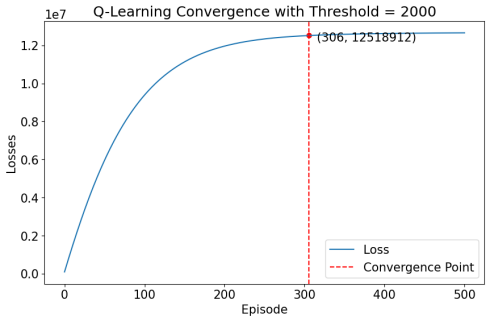


Figure 1: Q-Learning Convergence for First Experiment

Comparing First Experiment's Q-Learning to Naive Strategies. The comparison of Sharpe ratios among different trading strategies reveals significant insights. The Q-Learning strategy, with a Sharpe ratio of 0.9203, demonstrates a robust performance, indicating a favorable risk-adjusted return. This high Sharpe ratio suggests that the Q-Learning approach effectively balances risk while achieving returns, showcasing its efficiency in navigating market dynamics.

In stark contrast, the Equal Sized Packets strategy results in a Sharpe ratio of 0. This outcome implies that the strategy neither gains nor loses money, or that its returns are exactly offset by its risk. It could indicate a conservative strategy that fails to capitalize on market opportunities, resulting in a neutral performance.

The Eager Agent strategy, with a Sharpe ratio of -0.5774, indicates a poor performance with a negative risk-adjusted return. This strategy, characterized by liquidating the entire position at the outset, evidently entails significant risk without commensurate returns. It might be reflective of impulsive trading behavior that fails to account for market trends or timing considerations.

Overall, these results clearly favor the Q-Learning strategy, underlining its superiority in achieving a balance between risk and return as compared to the more simplistic Equal Sized Packets and Eager Agent strategies. This underscores the potential of advanced trading algorithms like Q-Learning in enhancing trading performance in complex financial markets.

Parameter Selection for Second Experiment. In a new experimental setup, changes are made only to the degree of market impact and the standard deviation of market noise, while other parameters remain the same. The key alterations are in the degree of market impact, d , which is now increased to 3 from 2, and the standard deviation of market noise,σ, elevated to 0.8 from 0.5.These changes imply a more pronounced market impact and greater volatility or randomness in market conditions, respectively. Such adjustments in the trading environment are crucial for assessing the adaptability and performance of the Q-Learning Agent under varying market dynamics.

Convergence for Q-Learning in the Second Experiment. Similar to the first experiment, the algorithm converges at 317 episode. Since the market is more complicated compared to last experiment, it takes the agent longer to learn the environment.

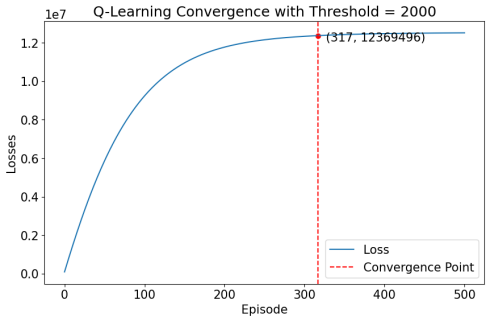


Figure 2: Q-Learning Convergence for Second Experiment

Comparing Second Experiment's Q-Learning to Naive Strategies. In the new experimental setting, the Sharpe ratios for different trading strategies present a clear picture of their relative performances. The Q-Learning strategy, adapted to the modified environment, achieved a significantly higher Sharpe ratio of 1.6893. This impressive figure indicates a substantial improvement in risk-adjusted returns compared to the previous experiment. The notable increase in the Sharpe ratio suggests that the Q-Learning strategy is highly effective in navigating the altered market conditions, particularly the increased market impact and noise.

On the other hand, the Equal Sized Packets strategy maintained a Sharpe ratio of 0, consistent with the previous experiment. This outcome continues to suggest a neutral performance, where the strategy neither generates significant returns nor incurs notable risks. It may indicate a conservative approach that fails to leverage market movements effectively.

The Eager Agent strategy, characterized by executing the entire position at the outset, again registered a negative Sharpe ratio of -0.5774, mirroring its performance in the initial setting. This persistently negative value highlights the strategy's inability to manage risk effectively, leading to poor risk-adjusted returns.

Overall, these results demonstrate the superior adaptability and performance of the Q-Learning strategy in varied market environments, as evidenced by its markedly improved Sharpe ratio in the new experiment. In contrast, the unchanged results for the Equal Sized

Packets and Eager Agent strategies reinforce their limitations in terms of risk management and return optimization.

Financial Insights The experimental results offer valuable financial and economic insights, particularly highlighting the efficacy of advanced algorithmic strategies like Q-Learning in dynamic market environments. The significant improvement in the Sharpe ratio for the Q-Learning strategy in the altered environment, where market impact and noise were increased, underscores its adaptability and sophistication. Unlike simpler strategies such as Equal Sized Packets and Eager Agent, which showed no improvement or continued poor performance, the Q-Learning approach adeptly adjusted to heightened volatility and market impact. This suggests that algorithmic strategies can be highly effective in complex and rapidly changing financial markets, providing an edge in managing risk while optimizing returns. Economically, this indicates a potential shift in trading paradigms, where reliance on traditional, less dynamic strategies may be less favorable compared to algorithmic, data-driven approaches. The negative performance of the Eager Agent strategy in both scenarios further emphasizes the importance of strategic timing and market analysis over impulsive or simplistic trading behaviors. These insights could influence future trading practices, risk management approaches, and the development of more resilient and responsive trading algorithms in the financial sector.

Extra Credit In the environment set-up provided, I found out that the reward function only includes temporary market impact. It doesn't penalize holding high inventory at each time. Therefore, as extract credit, I'd like to explore that whether adding an inventory cost to the reward function would affect the agent's behavior or not.

Parameter Selection for Extra Credit Experiment In this experiment, the Reward Function is modified to include inventory costs. The calculation of the reward function at each time step t is now influenced by both the executed price and the inventory levels. For the time steps from 0 up to $T-1$, the reward $r_t(p, i, a)$ is defined as the product of $X(p, a)$, which is the average executed price, and the action a, and is calculated as $\gamma_1 \times (1-a)^2$. Here p represents the current mid-price, and a is the action taken by the agent. At the terminal time step T, the reward r_T is computed differently. It is the product of the average executed price $X(p, a)$ and the remaining inventory i minus a term$\gamma_2 \times i^2$ that represents the cost of holding inventory. This formulation reflects the fact that holding inventory incurs a cost, which is factored into the final reward. In this setup, γ_1 and γ_2 are the inventory cost coefficients. For the purpose of this experiment,γ_1 is set at 0.5 and γ_2 at 0.05. These coefficients quantify the cost associated with the inventory, influencing the decision-making process of the trading agent. By incorporating these inventory costs into the reward function, the model becomes more realistic and aligned with real-world trading scenarios, where holding inventory can have significant financial implications.

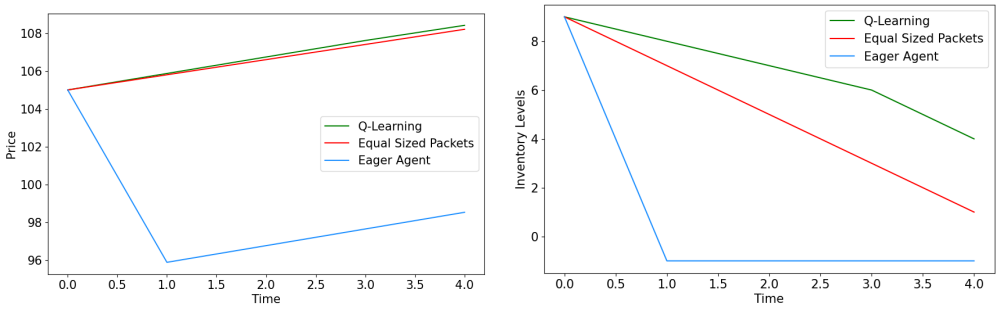
Extra Credit Experiment Result. In the extra credit section of the experiment, where inventory costs were incorporated into the reward function, the Sharpe ratios for the various

trading strategies exhibited noteworthy differences. The Q-Learning strategy, adapted to this new reward function, achieved an exceptionally high Sharpe ratio of 3.3284. This marked increase in the Sharpe ratio indicates a significant enhancement in risk-adjusted returns, suggesting that the Q-Learning approach is extremely effective in managing the complexities introduced by inventory costs.

Contrastingly, the Equal Sized Packets strategy continued to show a Sharpe ratio of 0. This consistency across different experimental settings indicates that this strategy, while not generating significant returns, also does not incur substantial risks. It remains a neutral, albeit conservative, approach that does not actively capitalize on market movements or inventory considerations.

The Eager Agent strategy, characterized by an immediate liquidation of the entire position, again recorded a negative Sharpe ratio of -0.5774. This persistent negative performance underscores the strategy's ineffectiveness in a scenario where inventory costs are critical, reflecting a poor risk-return balance.

These results, particularly the outstanding performance of the Q-Learning strategy under the new reward structure, highlight the potential of sophisticated algorithmic trading strategies in environments where inventory considerations are crucial. The findings also reiterate the limitations of more simplistic trading strategies in handling the nuances of inventory costs and market dynamics.



(a) Price Over Time (b) Inventory Over Time

Figure 3: Price Inventory Graph Compared for Extra Credit Trader and Naive Strategies.

Conclusion. In conclusion, the experiments demonstrated the advantages of utilizing advanced algorithmic strategies like Q-Learning in dynamic and complex trading environments. Their ability to process and adapt to various market factors, including inventory costs, positions them as valuable tools for traders seeking to maximize returns while effectively managing risk.

References

[1] Hambly, Ben M.; Xu, Renyuan; Yang, Huining. (2021). "Recent Advances in Reinforcement Learning in Finance". SSRN.

[2] Wenhang Bao and Xiao-Yang Liu. (2019). "Multi-Agent Deep Reinforcement Learning for Liquidation Strategy Analysis". arXiv:1906.11046v1 [q-fin.TR] 24 Jun 2019.