ONLINE LEARNING IN LIMIT ORDER BOOK TRADE EXECUTION

Nima Akbarzadeh, Cem Tekin

Bilkent University
Electrical and Electronics Engineering Department
Ankara, Turkey

Mihaela van der Schaar

Oxford Man Institute of
Quantitative Finance
Oxford, United Kingdom

ABSTRACT

In this paper, we propose an online learning algorithm for optimal execution in the limit order book of a financial asset. Given a certain amount of shares to sell and an allocated time window to complete the transaction, the proposed algorithm dynamically learns the optimal number of shares to sell via market orders at pre-specified time-slots within the allocated time interval. We model this problem as a Markov Decision Process (MDP), which is then solved by dynamic programming. First, we prove that the optimal policy has a specific form, which requires either selling no shares or the maximum allowed amount of shares at each time slot. Then, we consider the learning problem, where the state transition probabilities are unknown and need to be learned on-the-fly. We propose a learning algorithm that exploits the form of the optimal policy when choosing the amount to trade. Our numerical results show that the proposed algorithm performs significantly better than the traditional Q-learning algorithm by exploiting the structure of the problem.

Index Terms— Limit order book, Markov decision process, online learning, dynamic programming.

1. INTRODUCTION

Optimal execution of trades is a problem of key importance for any investment activity [1–4]. Once the decision has been made to sell a certain amount of shares the challenge often lies in how to optimally place this order in the market. In simple terms, we can formulate the objective as selling at the highest price possible. Not only do we want to leave as little a foot-print in the market as possible, but also to sell at a price favorable to the order in question, while ensuring the trade actually gets done.

More formally we define the goal as to sell[1] a specific number of shares of a given stock during a fixed time period in a way that minimizes the accumulated cost of the trade. This problem is also called the optimal liquidation problem. In this problem, the traders can specify the volume and the price of shares that they desire to sell in the limit order book (LOB). A brief discussion of the LOB mechanism is given in [5, 6].

Numerous prior works solve this problem using static optimization approaches or dynamic programming [3, 7]. Several other works tackle this problem using a reinforcement learning approach [4, 5, 8].

Reinforcement learning based methods consider various definitions of state, such as the remaining inventory, elapsed time, current spread, signed volume, etc. Actions are defined either as the volume to trade with a market order or as a limit order. A hybrid method is proposed in [4]: firstly, an optimization problem is solved to define an upper bound on the volume to be traded in each time slot, using the Almgren Chriss (AC) model proposed in [3]. Then, a reinforcement learning approach is used to find the best action, i.e., the volume to trade, which is upper-bounded by a relative value obtained in the optimization problem. Another prior work [5] implements the same approach with a different action set and state space. In all of the above works, the authors used Q-learning to find the optimal action for a given state of the system. In [4, 5] the learning problem is separated into training and test phases, where the Q values are only updated in the training phase, and then, these Q values are used in the test phase.

Unlike prior approaches, we use a model based approach, in which we start with a market model, and then, learn the state transition dynamics of the model in an online manner. For this, we design an algorithm that selects actions by learning the state transition probabilities of the market variables. Specifically, we separate the state space into private and market variables. We define the market variable as the difference in bid price of a time slot from the bid price of the time slot at the beginning of a round. Then, we deduce the form of the optimal policy using the mentioned decomposition of the variables, which reduces the number of actions to learn, and hence, speeds up both optimization and learning. The contributions of this paper can be summarized as follows:

- We propose a new model for LOB trade execution with private and market states, and show that the optimal policy has a special structure.
- We propose an online learning algorithm that greedily exploits the estimated optimal policy. Unlike other

[1] This problem can generalized to buying problem as well.

reinforcement learning based approaches [9, 10], this algorithm does not need explorations to learn the state transition probabilities.

- We show that the proposed algorithm provides significant performance improvement over its competitors in real-world datasets.

2. PROBLEM FORMULATION

2.1. States, Actions, Transitions and Cost

The system operates in rounds indexed by $\rho \in \{1, 2, \ldots\}$. Each round is composed of L time slots, where L denotes the *maximum execution time*. The set of time slots is denoted by $\mathcal{L} := \{1, \ldots, L\}$. The current round ends and a new round begins when the maximum execution time is reached.

States: The system is composed of a finite set of states denoted by $\mathcal{S} := \mathcal{I} \times \mathcal{M}$, where $\mathcal{I}$ denotes the *private state space* and $\mathcal{M}$ denotes the *market state space*. $\mathcal{I} := \{W_{\min}, \ldots, W_{\max}\}$ is the set of inventory levels of shares, where $0 < W_{\min} \leq W_{\max} < \infty$ and $W_{\min}$, $W_{\max}$ are integers. The private state at time slot l of round ρ is denoted by I^l_ρ. We assume that $I^1_\rho = W_\rho$ where $W_\rho \in \mathcal{I}$ is the initial inventory level at round ρ. Next, we define $\mathcal{M}$. For this, let $p_b(\rho, l)$, $p_a(\rho, l)$ and $p_m(\rho, l)$ be the bid, ask and mid-price (the average between bid and ask price) of time slot l in round ρ. The return of round ρ is defined as $\mathrm{Ret}(\rho) := \log(p_m(\rho, L)/p_m(\rho, 1))$. The volatility (standard deviation of the return) up to round ρ is denoted by σ_ρ. Based on this, $\mathcal{M}$ is defined as the set of integers that represent the amount of change in the bid price of a time slot in a round from the bid price in the beginning of the round in units of σ_ρ. The market state at time slot l of round ρ is denoted by M^l_ρ. By definition, $M^1_\rho = 0$ for all rounds.

We define the reference price of round ρ as $p_r(\rho) := p_m(\rho, 1)$ and the bid-ask spread in time slot l of round ρ as $B^l_\rho := p_a(\rho, l) - p_b(\rho, l)$. We have $p_b(\rho, l) = p_m(\rho, l) - B^l_\rho/2$. In our model, when the market is in state M in time slot l of round ρ, the bid price is modeled as $p_b(\rho, l) = p_b(\rho, 1) + M\sigma_\rho$. This means that the bid prices are put into discrete values with width σ_ρ, where each one of them is indexed by $M \in \mathcal{M}$. We let $S^l_\rho := (I^l_\rho, M^l_\rho)$ denote the joint state in time slot l of round ρ.

Actions: Actions are defined to be the amount of shares to be traded with a market order[2] [4, 8]. In each round, a sequence of actions is selected with the aim of minimizing the total trade cost. Let a^l_ρ be the action taken at time slot l in round ρ. We impose the following assumption on the effect of actions to the market states.

[2]A market order to sell is an order to execute a trade at whatever the best prevailing bid price which is a limit order with a price limit of zero at that time.

Assumption. *It is assumed that the order book is resilient to the trading activities.*

The assumption implies that an action in a time slot has to be chosen such that it does not have an influence on the market state during a round. In practice this means that the market order should not be larger than the depth of the order book at the best bid. This is imposed, for instance, in [3, 4], which effectively prevents taking large actions. We assume that the action taken in time slot l of a round cannot be larger than the integer A_l, where $A_l, l \in \mathcal{L}$ is obtained in each round by using the AC model, and hence, we have $\sum_{l=1}^{L} A_l = W_\rho$. Then, we have $a^l_\rho \in \mathcal{A}_l := \{0, \ldots, A_l\}, \forall l \in \mathcal{L} - \{L\}$. For $l = L$, the only possible action is to sell the remaining inventory.

Transitions: The assumption implies that the market state in a round evolves independently from the actions selected by the trader. Hence, the actions only affect the private state, and the market state is modeled as a Markov chain. Let $S' := (I', M')$ and $S := (I, M)$. The state transition probabilities can be written as

$$\Pr(S^{l+1}_\rho = S' | S^l_\rho = S, a^l_\rho = a) = P(M, M')\mathbb{I}(I', I - a),$$
$$S, S' \in \mathcal{S}, a \in \mathcal{A}_l, \rho \geq 1$$

where $\mathbb{I}(a, b)$ represents the indicator function which is zero when $a \neq b$ and one when $a = b$, and $P(M, M')$ denotes the probability that the market state transitions from M to M'.

Cost: The cost of trade in a round is defined as the *implementation shortfall*, which is given as

$$\mathrm{IS}_\rho := \left(W_\rho p_r(\rho) - \sum_{l=1}^{L} a^l_\rho p_b(\rho, l) \right) / (W_\rho p_r(\rho)) \qquad (1)$$

for a sequence of market states $(M^1_\rho, \ldots, M^L_\rho)$, a sequence of actions $(a^1_\rho, \ldots, a^L_\rho)$, an inventory level W_ρ such that $\sum_{l=1}^{L} a^l_\rho = W_\rho$, and a reference price $p_r(\rho)$. Let $X_\rho := \left(W_\rho, p_r(\rho), \sigma_\rho, B^1_\rho\right)$ be the trade vector of round ρ and let $\mathcal{X}$ be the support of this vector. By using the state definition and B^1_ρ, (1) can be re-written as $\mathrm{IS}_\rho = \sum_{l=1}^{L} C_{X_\rho}(M^l_\rho, a^l_\rho)$ where

$$C_{X_\rho}(M^l_\rho, a^l_\rho) := \frac{1}{W_\rho p_r(\rho)} \left[a^l_\rho \left(\frac{B^1_\rho}{2} - M^l_\rho \sigma_\rho \right) \right].$$

2.2. Value Functions and the Optimal Policy

If the state transition probabilities were known in advance, then, the optimal policy can be computed by dynamic programming. In this subsection, we consider this case to gain insight on the form of the optimal policy.

A deterministic Markov policy with time budget L specifies the actions to be taken for each state and trade vector at each time slot. Let $\boldsymbol{\pi} := (\pi_1, \pi_2, \ldots, \pi_L)$ denote such a policy, where $\pi_l : \mathcal{S} \times \mathcal{X} \to \mathcal{A}_l$. We use $\pi_l(M_l, I_l, X)$ to denote the action selected by policy $\boldsymbol{\pi}$ in time slot l when the joint state is (M_l, I_l) in time slot l given X.[3,4] Let Π denote the set of all deterministic Markov policies with time budget L.

The cost incurred by following policy $\boldsymbol{\pi}$ given trade vector $X \in \mathcal{X}$ is $C_X^{\boldsymbol{\pi}} = \sum_{l=1}^{L} C_X(M_l, \pi_l(M_l, I_l, X))$. The optimal policy that minimizes $\mathbb{E}[C_X^{\boldsymbol{\pi}}]$ is given as $\boldsymbol{\pi}^*(X)$ where the expectation is taken over the randomness of the market states. Let $V_l^*(M, I, X)$ denote the expected cost (value function) of policy $\boldsymbol{\pi}^*(X)$ starting from state $S = (M, I)$ at time slot l given X. The Bellman optimality equations [11, 12] are given below: $\forall M \in \mathcal{M}$, $\forall I \in \mathcal{I}$, $\forall X \in \mathcal{X}$, $\forall l \in \mathcal{L} - \{L\}$,

$$Q_l^*(M, I, X, a) := C_X(M, a) + \sum_{M' \in \mathcal{M}} P(M, M') V_{l+1}^*(M', I - a, X) \quad (2)$$

$V_l^*(M, I, X) = \min_{a \in \mathcal{A}_l} Q_l^*(M, I, X, a), \forall l \in \mathcal{L} - \{L\}$ and $V_L^*(M, I, X) = C_X(M, I)$. The optimal action can be computed by $\pi_l^*(M, I, X) = \arg\min_{a \in \mathcal{A}_l} Q_l^*(M, I, X, a)$ and we know that $\pi_L^*(M, I, X) = I$. In order to obtain the optimal policy, these equations need to be solved backwards from time slot L down to 1.

3. ON THE FORM OF THE OPTIMAL POLICY

In this section, we show that the optimal policy takes a simple form, which reduces the set of candidates for the optimal action in each time slot to two.

Theorem. *Let* $g_X(M) := (B/2 - M\sigma)/(p_r W)$ *where* $X = (W, p_r, \sigma, B)$.[5] *Then, the optimal action at each time slot is*

$$\pi_l^* = \begin{cases} 0 & \text{if } g_X(M_l) > \mathbb{E}[g_X(M_L)|M_l] \\ A_l & \text{if } g_X(M_l) \le \mathbb{E}[g_X(M_L)|M_l] \end{cases}, \forall l \in \mathcal{L} - \{L\}$$

and $\pi_L^* = I_L$.

Proof. The proof is given in the online appendix [13]. □

The theorem shows that the optimal action at each time slot depends on the current market state and the distribution of the market state at the final time slot given the current state. The trader may decide to sell all of the available limit at the current time slot or save the shares up to the final time slot. The intuitive reason behind the result is that we have a linear cost function in a and $g_X(M)$. If the expected market state in the final time slot is greater than the current market state, we desire to wait and sell the maximum allowed amount of shares to sell in the current time slot in the final time slot. The reason for this is that, the final time slot is the only time slot where we can sell more than the pre-defined limit. Thus, the set of candidate optimal actions is given as $\mathcal{A}_l^* := \{0, A_l\}$, $\forall l \in \mathcal{L} - \{L\}$. Therefore, the learning problem reduces to learning the best of these two actions in each time slot.

4. THE LEARNING ALGORITHM

In this section, we propose a learning algorithm that learns the optimal policy by exploiting the form of the optimal policy given in the previous section by learning the state transition probabilities of the market variables. This algorithm is named as *Greedy exploitation in Limit Order Book Execution* (GLOBE) and its pseudocode is given in Algorithm 1.

Algorithm 1 GLOBE

```
1:  Input: $L, \mathcal{S}$
2:  Initialize: $\rho = 1, N_1(M, M') = N_1(M) = 0, \forall M, M' \in \mathcal{M}$
3:  while $\rho \ge 1$ do
4:      $\hat{P}_\rho(M, M') = \frac{N_\rho(M, M') + \mathbb{I}(N_\rho(M) = 0)}{N_\rho(M) + |\mathcal{M}|\mathbb{I}(N_\rho(M) = 0)}$
5:      Update $\sigma_\rho$ and temporary price impact parameter of the AC model based on the past observations
6:      Observe $X_\rho = (W_\rho, p_r(\rho), \sigma_\rho, B_\rho^1)$
7:      Compute $A_l$ based on the AC model [3,4], $\forall l \in \mathcal{L}$
8:      Compute the estimated optimal policy by dynamic programming using the action set $\mathcal{A}_l^*$, $\forall l \in \mathcal{L} - \{L\}$ and $\hat{P}_\rho(M, M')$ values.
9:      $I_\rho^1 = W_\rho, l = 1$
10:     while $l < L$ do
11:         Observe market state $M_\rho^l$, obtain $a_\rho^l \in \mathcal{A}_l^*$ using the estimated optimal policy
12:         Calculate $C_{X_\rho}(M_\rho^l, a_\rho^l)$
13:         $I_\rho^{l+1} = I_\rho^l - a_\rho^l$
14:         $l = l + 1$
15:     end while
16:     $a_\rho^L = I_\rho^L$
17:     $\rho = \rho + 1$
18:     Compute $N_\rho(M)$ and $N_\rho(M, M')$, $\forall M, M' \in \mathcal{M}$
19: end while
```

By round ρ, GLOBE observes $(\rho - 1)(L - 1)$ state transitions. Let $N_\rho(M, M')$ denote the number of occurrences of a state transition from market state M to M' and $N_\rho(M)$ denote the number of times market state M is visited by round ρ. The estimate of the transition probability from state M to M' used at round ρ is denoted by $\hat{P}_\rho(M, M') := N_\rho(M, M')/N_\rho(M)$ for $N_\rho(M) > 0$. At the beginning of each round, GLOBE implements dynamic programming with action set $\mathcal{A}_l^*$ instead of $\mathcal{A}_l$ to find the estimated optimal policy, and follows that

[3]When clear from the context, we will drop the arguments, and represent the action selected by the policy in time slot l by π_l.

[4]We replace M_ρ^l and I_ρ^l with M_l and I_l when the round is clear from the context.

[5]$C_X(M, a) = a g_X(M)$.

policy during the round. Alternatively, it can also use the rule given in the Theorem at each time slot of the round in order to decide on whether to sell A_l or 0, by finding the expected market state in the final time slot of the round using $\hat{P}_\rho(M, M')$ values. The above procedure repeats in each round.

5. NUMERICAL ANALYSIS

Our numerical analysis is based on four real-world datasets that contain the order book data for Google, Amazon, Intel and Microsoft shares traded on NASDAQ.[6] [7]

In this problem, the trader wants to sell W_ρ number of shares in round ρ at the best price using market orders. We take the difference in the bid prices as the only market variable and the private variable is assumed to be the inventory level. The number of states varies from dataset to dataset based on the volatility scale. We assume that when the market is at state M in time slot l of round ρ, we have $p_b(\rho, 1) + (M - 0.5)\sigma_\rho \leq p_b(\rho, l) \leq p_b(\rho, 1) + (M + 0.5)\sigma_\rho$. To reduce the number of market states, we use $20\sigma_\rho$ instead of σ_ρ in the market state definition and the above inequalities, for which GLOBE is shown to work well. The initial inventory level of each round is drawn uniformly at random from $[10, 100]$. The time horizon of the original data is approximately 6 hours and 30 minutes. Each data instance for each time slot is created by taking the average of the mid/bid/ask prices for every 10 second interval. Then, the dataset is divided into rounds, where each round consists of $L = 4$ consecutive time slots.

The volatility parameter used in the AC model is updated in an online manner as we observe more data. Furthermore, similar to [4], we set the permanent price impact parameter to 0. In addition, we set $\lambda = 0.1$ in the AC model given in [3], and the temporary price impact parameter is calculated and updated according to [3]. Next, we describe the algorithms that we compare GLOBE against:[8]

(1) Equal Shares (EQ): In this method, we divide the shares equally among the time slots and at each time slot of round ρ, we sell $\lfloor W_\rho/L \rceil$ [7]. (2) Almgren Chriss (AC) [3]: The volatility and temporary price impact parameters are updated after each round. (3) Q-Exp: This is a Q-learning based method, which uses the state space defined in [4] and the action space defined in our paper. It uses the ϵ-greedy policy [14] to explore (5%) or exploit (95%). In this method, the market state space is the combination of bid-ask spread and bid volumes, where each variable consists of 10 different states. (4) Q-Mat: This is the method proposed in [4], but with the action space defined as in our paper. This method uses the first half of the dataset as the training set to calculate the Q values, and the rest as the test set. (5, 6, 7) Q-Exp+, Q-Mat+ and GLOBE+: These are almost the same as Q-Exp, Q-Mat and GLOBE, where the only difference is that the set of available actions in time slot l ranges from 0 to $2A_l$. Unlike GLOBE, GLOBE+ does not use the result of the Theorem to reduce the size of the action sets.

For each method, we calculate the Averaged Cost Per Round (ACPR) at the end of each round, which is given as $\text{ACPR}_\rho = \sum_{i=1}^{\rho} \text{IS}_i/\rho$. Then, we compare ACPR_R of each method (alg) against the AC model by round R, using a performance metric similar to the one used in [15], which is called the Relative averaged Cost per round (RC), given as

$$\text{RC}_R(alg) := \frac{\text{ACPR}_R(AC) - \text{ACPR}_R(alg)}{|\text{ACPR}_R(AC)|} \times 100.$$

In Table 1 we report the RC of the algorithms for the second half of the dataset (test dataset)[9]. We observe that GLOBE and GLOBE+ outperforms other methods in the considered datasets. GLOBE+ also outperforms GLOBE since it has access to a larger action set.

Table 1: RC of the algorithms at the end of the time horizon with respect to the AC model calculated over the test set.

Method / Dataset	GOOG	AMZN	INTC	MSFT
EQ	0.145	0.017	-0.161	0.075
Q-Exp	0.303	0.128	-0.459	-0.138
Q-Mat	0.398	0.316	-0.365	-0.141
Q-Exp+	0.252	0.116	-0.64	0.057
Q-Mat+	0.532	1.158	-0.451	-0.284
GLOBE	3.064	2.783	3.167	2.555
GLOBE+	6.527	7.563	7.414	5.988

6. CONCLUSION

In this paper, we proposed an online learning algorithm for trade execution in LOB. We modeled this problem as an MDP using a novel market state definition, and derived the form of the optimal policy for this MDP. Then, we developed a learning algorithm that learns the optimal action using the state transition probability estimates. We also showed that our method outperforms its competitors in a small scale real-world finance dataset. As a future work, we will investigate the performance of GLOBE on larger real-world datasets with more different types of stocks.

7. ACKNOWLEDGMENT

The authors would like to thank Thomas Flury from MAN AHL for his guidance and feedback.

[6]https://lobsterdata.com/info/DataSamples.php.

[7]For simulations, to create both increasing and decreasing trends, the data is expanded by attaching a flipped version of the data to the end of the original data. Then, this extended data is repeated four times. For instance, given the original data "123", the expanded data becomes "123321123321123321123321".

[8]The results of all of the Q-learning based methods are averaged over 50 iterations.

[9]This is done in order to have a fair comparison between the models.

8. REFERENCES

[1] Deepan Palguna and Ilya Pollak, “Non-parametric prediction in a limit order book,” in *Proc. IEEE Global Conf. Signal and Information Processing (GlobalSIP)*, 2013, pp. 1139–1139.

[2] Yiyong Feng, Daniel P Palomar, and Francisco Rubio, “Robust optimization of order execution,” *IEEE Transactions on Signal Processing*, vol. 63, no. 4, pp. 907–920, 2015.

[3] Robert Almgren and Neil Chriss, “Optimal execution of portfolio transactions,” *Journal of Risk*, vol. 3, pp. 5–40, 2001.

[4] Dieter Hendricks and Diane Wilcox, “A reinforcement learning extension to the Almgren-Chriss framework for optimal trade execution,” in *Proc. IEEE Conf. Computational Intelligence for Financial Engineering & Economics (CIFEr)*, 2014, pp. 457–464.

[5] Yuriy Nevmyvaka, Yi Feng, and Michael Kearns, “Reinforcement learning for optimized trade execution,” in *Proc. 23rd Int. Conf. Machine Learning*, 2006, pp. 673–680.

[6] Rama Cont and Adrien De Larrard, “Price dynamics in a Markovian limit order market,” *SIAM Journal on Financial Mathematics*, vol. 4, no. 1, pp. 1–25, 2013.

[7] Dimitris Bertsimas and Andrew W. Lo, “Optimal control of execution costs,” *Journal of Financial Markets*, vol. 1, no. 1, pp. 1–50, 1998.

[8] Alexander A. Sherstov and Peter Stone, “Three automated stock-trading agents: A comparative study,” in *Int. Workshop Agent-Mediated Electronic Commerce*, 2004, pp. 173–187.

[9] Peter Auer and Ronald Ortner, “Logarithmic online regret bounds for undiscounted reinforcement learning,” *Advances in Neural Information Processing Systems*, vol. 19, pp. 49, 2007.

[10] Peter Auer, Thomas Jaksch, and Ronald Ortner, “Near-optimal regret bounds for reinforcement learning,” in *Advances in Neural Information Processing Systems*, 2009, pp. 89–96.

[11] Richard Bellman, “Dynamic programming and stochastic control processes,” *Information and Control*, vol. 1, no. 3, pp. 228–239, 1958.

[12] Dimitri P. Bertsekas, *Dynamic programming and optimal control*, vol. 1, Athena Scientific Belmont, MA, 1995.

[13] Nima Akbarzadeh, Cem Tekin, and Mihaela Van der Schaar, “Supplemental material for online learning in limit order book trade execution,” available at `http://kilyos.ee.bilkent.edu.tr/~cemtekin/GlobalSIPs.pdf`.

[14] Richard S. Sutton and Andrew G. Barto, *Reinforcement learning: An introduction*, vol. 1, MIT Press Cambridge, 1998.

[15] D. Palguna and I. Pollak, “Mid-price prediction in a limit order book,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 6, pp. 1083–1092, 2016.