Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Special Track on AI in FinTech

Data-Driven Market-Making via Model-Free Learning

Yueyang Zhong[1] , **YeeMan Bergstrom**[2] and **Amy Ward**[3]
[1,3]Booth School of Business, University of Chicago
[2]Proprietary Trading, Chicago
yzhong0@chicagobooth.edu, yee.man.bergstrom@gmail.com, Amy.Ward@chicagobooth.edu

Abstract

This paper studies when a market-making firm should place orders to maximize their expected net profit, while also constraining risk, assuming orders are maintained on an electronic limit order book (LOB). To do this, we use a model-free and off-policy method, Q-learning, coupled with state aggregation, to develop a proposed trading strategy that can be implemented using a simple lookup table. Our main training dataset is derived from event-by-event data recording the state of the LOB. Our proposed trading strategy has passed both in-sample and out-of-sample testing in the backtester of the market-making firm with whom we are collaborating, and it also outperforms other benchmark strategies. As a result, the firm desires to put the strategy into production.

1 Introduction

We consider a financial asset traded on an electronic exchange. Market participants, including institutional investors, market makers, and speculators, can post two types of buy/sell orders. A *market order* is an order to buy/sell a certain quantity of the asset at the best available price in the market. A *limit order* is an order to trade a certain amount at a specified price, known as an *ask price* for a sell order, and a *bid price* for a buy order. Limit orders are posted to an electronic trading system, and all the outstanding limit orders are summarized by stating the quantities posted at each price level in a *limit order book* (LOB), as shown in Figure 1, which is the dominant market structure among exchange-traded U.S. equities and futures. The LOB is available to all market participants.

The limit orders rest or wait in the LOB, and are matched against incoming market orders. A market buy (sell) order executes first at the lowest ask (highest bid) price, and next in ascending (descending) order with higher (lower) priced asks (bids). The execution within each price level is prioritized in accordance with the limit order time of arrival, in a first-come-first-served (FCFS) fashion.

In this paper, we take the perspective of a market-making firm. The market-making firm provides liquidity by submitting limit orders, and removes liquidity by canceling existing limit orders. Provided that the lowest ask price exceeds the highest bid price,[1] the market-making firm earns profit when one market order to buy trades with its resting limit sell order and another market order to sell trades with its resting limit buy order. The challenge is that the market-making firm cannot guarantee always being on both sides of the trade due to the stochasticity of order arrivals, and the resulting movements of the lowest ask and highest bid prices. The market-making firm with whom we partnered prefers to begin with the simplest possible strategy that places at most one order per side. Furthermore, the firm is most interested in a strategy for placing orders at the best bid and ask prices.

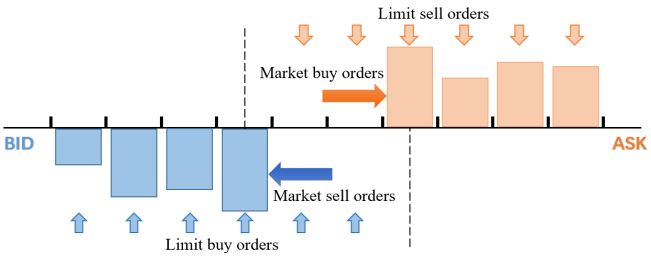

Figure 1: An illustration of a limit order book (LOB)

Our objective is to provide real-time guidance for how to manage the firm's portfolio of limit buy and sell orders on the LOB, so as to maximize the expected net profit, while penalizing mismatch between the amount bought and sold, and ensuring a sufficiently high Sharpe ratio.[2] To do this, we use historical trading data to train a model for real-time decision making. More specifically, we formulate this problem as a Markov decision problem (MDP). Two main issues in solving the MDP are: (1) difficulty in estimating the transition probabilities, and (2) a very large state space (the notorious curse of dimensionality). To overcome these issues and be able to find a well-performing heuristic, we implement a model-free

[1]The arrivals of limit buy orders with bid prices higher than the lowest ask price will be fulfilled immediately, similarly for the arrivals of limit sell orders with ask prices lower than the highest bid price; thus the highest bid price does not exceed the lowest ask price.

[2]The Sharpe ratio measures the return of an investment compared to its risk. Usually, any Sharpe ratio greater than 1.0 is considered acceptable to good by investors. A ratio higher than 2.0 is rated as very good. A ratio of 3.0 or higher is considered excellent.

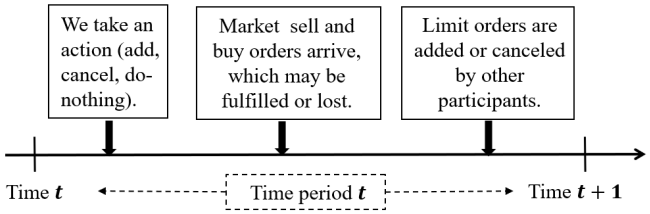

Figure 2: Timing of LOB events

Q-learning algorithm together with state aggregation.

2 Model

We model this problem as a finite-horizon discrete-time MDP. The simplified assumed timing of events happening in the LOB is illustrated in Figure 2. The objective is to provide a strategy for when (and when not) to have one buy and/or one sell order resting on the LOB. The assumption that at most one buy and one sell order can rest on the buy and sell side respectively is based on a high-frequency trading convention to (1) backtest whether the simple strategy is profitable, (2) see how the simple strategy performs in production, and (3) expand to more complicated order strategies (such as order stacking).

2.1 LOB State Variable

Assume there are n price levels in the order book, indexed by $\mathcal{P} := \{1, 2, \ldots, n\}$. At time $t \in \mathcal{T} := \{0, 1, 2, ..., T\}$, $|R^1_{tp}| \in \{0, 1\}$ denotes whether there exists a limit order belonging to us at price $p \in \mathcal{P}$, and $|R^2_{tp}| \in \{0, 1, 2, \ldots\}$ denotes the total number of limit orders resting from other market participants at price $p \in \mathcal{P}$. We distinguish between the bid and the ask side according to whether R^i_{tp} $(i = 1, 2)$ is negative or positive; $R^i_{tp} < 0$ $(i = 1, 2)$ for the bid side, and $R^i_{tp} > 0$ $(i = 1, 2)$ for the ask side. Whenever the state is such that we have an order resting, we conservatively assume that our order rests at the back of the queue. The implication is that our model will tend to underestimate the frequency at which our orders are executed, resulting in an underestimation of profit.

The best bid and ask prices (also called the market bid and ask prices) can be expressed as a function of $R_t = (R^1_{tp}, R^2_{tp})_{p\in\mathcal{P}}$.

- The best bid price (which is the highest bid price) is $\beta_{R_t} := \max\{p \in \{0, 1, ..., n\} : R^1_{tp} + R^2_{tp} < 0\}$.
- The best ask price (which is the lowest ask price) is $\alpha_{R_t} := \min\{p \in \{1, ..., n, n+1\} : R^1_{tp} + R^2_{tp} > 0\}$.

In the above, $p = 0$ and $p = n + 1$ represent the degenerate cases of no bids and no asks, respectively. Since the best bid and ask prices can be determined from R_t, there is no need to include them as part of the state variable. Then, the pre-decision state variable at time t is given by R_t.

2.2 Decision Variable

A trading policy can be decomposed into a sequence of actions taken at the best bid and/or the best ask price. The available actions are to add, cancel, or do nothing, and we encode this using 0 and 1. A 0 on the bid side implies we do not want an order resting at the best bid price, and so we cancel any existing order on the bid side, and otherwise do nothing. A 1 implies we do want an order resting at the best bid price, and so we place an order at the best bid price and simultaneously cancel any existing order on the bid side. This leads to the allowable action space for any state R_t being $\mathcal{A} := \{(0, 0), (0, 1), (1, 0), (1, 1)\}$, where the two components in an action pair correspond to the action on the bid side and the ask side, respectively. Later, this will be useful for us to restrict the action space when there is too much mismatch between the amounts bought and sold, in which case the allowable actions will be a subset of $\mathcal{A}$. The state after taking an action $A_t = (A_{t1}, A_{t2}) \in \mathcal{A}$ can be expressed by two n-dimensional vectors R^{a1}_t and R^{a2}_t, defined as

$$\begin{aligned} R^{a2}_{tp} :=& R^2_{tp}, \quad \text{for all } p \in \{1, 2, \ldots, n\} \\ R^{a1}_{tp} :=& \begin{cases} 1, & \text{if} \quad A_{t1} = 1 \text{ and } p = \beta_{R_t} \text{ or} \\ & \quad\ A_{t2} = 1 \text{ and } p = \alpha_{R_t} \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{1}$$

2.3 Exogenous Order Arrivals and Cancellations

Let $\hat{D}^{MB}_t$ and $\hat{D}^{MS}_t$ be the number of units demanded respectively by market buy and sell orders, which arose between time t and $t + 1$. We have at most one resting order at α_{R_t} and one at β_{R_t}, which rest at the end of queue, and none elsewhere. The implication is that our orders execute if $\hat{D}^{MB}_t$ and/or $\hat{D}^{MS}_t$ is no fewer than the number of orders resting at the best ask and/or best bid; the state can then be updated in terms of the first n-dimensional vector, for all $p \in \mathcal{P}$,

$$R^{m1}_{tp} := \begin{cases} 0, & \text{if } p = \alpha_{R_t} \text{ and } \hat{D}^{MB}_t \geq R^{a1}_{tp} + R^{a2}_{tp}, \\ 0, & \text{if } p = \beta_{R_t} \text{ and } \hat{D}^{MS}_t \geq R^{a1}_{tp} + R^{a2}_{tp}, \\ R^{a1}_{tp}, & \text{otherwise.} \end{cases} \tag{2}$$

In order to update the second n-dimensional vector, which represents the resting orders from other market participants, we require more detailed knowledge. Define $p^\alpha_{R_t}$ to be the highest ask price against which a market buy order will execute.[3] If there are enough limit orders resting at the lowest ask price to fill the incoming market buy orders (i.e., $\hat{D}^{MB}_t \leq R^{a1}_{t\alpha_{R_t}} + R^{a2}_{t\alpha_{R_t}}$), then $p^\alpha_{R_t} = \alpha_{R_t}$ and the trade quantity at price α_{R_t} is $k^\alpha_{R_t} := \hat{D}^{MB}_t$. Otherwise $p^\alpha_{R_t} > \alpha_{R_t}$, the trade quantities at any ask prices p lower than $p^\alpha_{R_t}$ exactly equals the number of resting orders at the price, and the trade quantity at price $p^\alpha_{R_t}$ can be expressed by

$$k^\alpha_{R_t} := \hat{D}^{MB}_t - (R^{a1}_{t\alpha_{R_t}} + R^{a2}_{t\alpha_{R_t}}) - \textstyle\sum_{p=\alpha_{R_t}+1}^{p^\alpha_{R_t}-1} R^{a2}_{tp},$$

assuming $\hat{D}^{MB}_t \leq (R^{a1}_{t\alpha_{R_t}} + R^{a2}_{t\alpha_{R_t}}) + \sum_{p=\alpha_{R_t}+1}^{n} R^{a2}_{tp}$ (where the summation in the above display is the empty set if $p^\alpha_{R_t} = \alpha_{R_t} + 1$). In the rare case that the total number of limit orders resting on the book is not enough to fill all the incoming market buy orders (that is, if $\hat{D}^{MB}_t > (R^{a1}_{t\alpha_{R_t}} + R^{a2}_{t\alpha_{R_t}}) + \sum_{p=\alpha_{R_t}+1}^{n} R^{a2}_{tp}$), then $p^\alpha_{R_t} = n$ and the

[3] This assumes the non-degenerate case that there are ask orders resting on the LOB ($\alpha_{R_t} < n + 1$).

trade quantity at price n is the amount resting at that price, so that $k^{\alpha}_{R_t} := R^{a2}_{tn}$ (excess demand is lost). Similarly, define $p^{\beta}_{R_t}$ to be the lowest bid price against which a market sell order will execute,[4] and $k^{\beta}_{R_t}$ to be the trade quantity at price $p^{\beta}_{R_t}$. Then the second component of the state after the arrival of market orders is, for all $p \in \mathcal{P}$,

$$R^{m2}_{tp} := \begin{cases} 0, & \text{if } p \in \{p^{\beta}_{R_t}+1, \ldots, \beta_{R_t}\} \cup \\ & \quad \{\alpha_{R_t}, \ldots, p^{\alpha}_{R_t}-1\}, \\ R^{a2}_{tp} - k^{\alpha}_{R_t}, & \text{if } p = p^{\alpha}_{R_t}, \\ R^{a2}_{tp} - k^{\beta}_{R_t}, & \text{if } p = p^{\beta}_{R_t}, \\ R^{a2}_{tp}, & \text{otherwise,} \end{cases} \tag{3}$$

where R^{a2}_t is as defined in (1).

Finally, the other market participants add and cancel orders between time t and $t+1$, denoted by $\hat{O}_t = (\hat{O}_{tp})_{p\in\mathcal{P}}$ and $\hat{C}_t = (\hat{C}_{tp})_{p\in\mathcal{P}}$. This results in the state update

$$\begin{aligned} R^{o1}_{tp} &:= R^{m1}_{tp}, \\ R^{o2}_{tp} &:= R^{m2}_{tp} + \hat{O}_{tp} - \hat{C}_{tp}, \end{aligned} \quad \text{for all } p \in \mathcal{P}, \tag{4}$$

and has the restriction that $R^{m2}_{tp} + \hat{O}_{tp} \geq \hat{C}_{tp}$, for all $p \in \mathcal{P}$; i.e., the number of orders canceled cannot exceed the number of orders present.

2.4 Transition Function

According to the timing of events as shown in Figure 2, there are three state updates from time t to $t+1$, which are elaborated in equations (1), (2), (3), and (4). Then, we can write the pre-decision state vector in the next decision epoch as

$$R_{t+1} = (R^{o1}_{tp}, R^{o2}_{tp}). \tag{5}$$

2.5 Objective Function

Over the course of each day, the market-making firm gains profit and incurs loss when market orders execute against the market maker's resting limit orders. For a given state and action pair, (R_t, A_t), and arrival of market buy and sell orders, $(\hat{D}^{MB}_t, \hat{D}^{MS}_t)$, we define the contribution function as the common financial metric *profit and loss* (PnL):

$$C(R_t, A_t, \hat{D}^{MB}_t, \hat{D}^{MS}_t) := E^{\beta} \cdot (m_{R_t} - \beta_{R_t}) + E^{\alpha} \cdot (\alpha_{R_t} - m_{R_t}), \tag{6}$$

where binary variables E^{β} and E^{α} indicate respectively whether we have a resting order that was executed on the bid side and on the ask side, and $m_{R_t} := (\alpha_{R_t} + \beta_{R_t})/2$ denotes the mid price.

Since PnL is accounted for relative to the mid price, it is necessary to include another term in the objective function that penalizes the potential change in cash value due to movements in the mid price. To do this, the decision-maker must also track his open position, or inventory level. Recalling that $A_t = (A_{t1}, A_{t2})$ has first component corresponding to an action on the bid side and second component corresponding to an action on the ask side, the open position, or inventory level, between time t and $t+1$ after the arrival of market orders is defined as

$$\begin{aligned} inv_t := &\sum\nolimits_{i=0}^{t} \mathbb{1}\{A_{i1}=1\}\mathbb{1}\{\hat{D}^{MS}_i \geq R^{a1}_{i\beta_{R_i}} + R^{a2}_{i\beta_{R_i}}\} \\ &- \sum\nolimits_{i=0}^{t} \mathbb{1}\{A_{i2}=1\}\mathbb{1}\{\hat{D}^{MB}_i \geq R^{a1}_{i\alpha_{R_i}} + R^{a2}_{i\alpha_{R_i}}\}, \end{aligned} \tag{7}$$

where the first and second summation represent the cumulative amount bought and sold, respectively. Then, if Δm_t denotes the change in the mid price between time period $t-1$ and t (defined to be 0 for $t=0$), the objective function is

$$V(R_t, A_t, \hat{D}^{MB}_t, \hat{D}^{MS}_t, inv_t) := C(R_t, A_t, \hat{D}^{MB}_t, \hat{D}^{MS}_t) + inv_t \cdot \Delta m_t. \tag{8}$$

Some papers like [Spooner *et al.*, 2018] also studied two alternative penalty terms in the objective function, *symmetrically dampened PnL*: $\eta \cdot inv_t \cdot \Delta m_t$, and *asymmetrically dampened PnL*: $\min(0, \eta \cdot inv_t \cdot \Delta m_t)$, to disincentivize trend-following and bolster spread capture, because a dampening applied to the inventory term reduces the profit gained through speculation (i.e., following behavior) relative to that from capturing the spread. We also tried both in our experiments, but they do not display a better performance, so here we only consider the basic objective function in (8).

3 Data Analysis

Our dataset is a common and competitive futures contract traded on the Chicago Mercantile Exchange (CME)'s Globex electronic trading platform in 2019. It is Level II order book data, which provides a more granular information than the trade and quotes (TAQ) data mostly used by traders to do financial analysis. Since trading is extremely active during the time near market open and market close, the dynamics of the LOB may differ significantly during these time periods, as compared to the behavior throughout the remainder of the trading day. Hence, we truncate the data to the timeframe 9:00 a.m.–14:30 p.m. for every day.

In contrast to most of the literature where the time stamps are only accurate to 1 second, our event-by-event data records the state of the LOB with microsecond decimal precision, once an order submission, order cancellation, or order execution occurs. From this, we extract time-stamped detailed information on order adds, order cancels, and order transactions at each of the highest 10 price levels on the bid side and the lowest 10 price levels on the ask side.[5] Then, we aggregate the data at the second level and construct six time series, for the following six order book events: (1) market buy orders; (2) market sell orders; (3) limit buy orders; (4) limit sell orders; (5) cancellations on the bid side; (6) cancellations on the ask side. The reason we aggregate at the second level is that our purpose is to derive a strategy that can have slow execution speed (i.e., need not execute at the microsecond level or faster). This is because the market-making firm with whom we partnered does not view speed as its primary competitive advantage.

[4] This assumes the non-degenerate case that there are bid orders resting on the LOB ($\beta_{R_t} > 0$).

[5] For the product we study, the difference between the best bid and ask prices, i.e., the spread, is rarely more than one tick. A spread of more than one tick occurs less than 0.01% of the time. As a result, in contrast to some of the past literature [Spooner *et al.*, 2018], we do not need to record the spread for decision making purposes.

3.1 Independence Check

For analysis purposes, we would like to know that $\hat{D}_t^{MB}$, $\hat{D}_t^{MS}$, $\hat{O}_t$, $\hat{C}_t$ are independent across time, as well as independent of each other. The intuitive reason this may be true is that aggregation at the second level is large enough to minimize the impact of any following behavior occurring in the other market participants, as that usually happens at much finer timescales. In other words, although the data may show slight autocorrelations at the microsecond level, one second is large enough for that autocorrelation to be negligible.

We investigate the autocorrelation and cross-correlation of the sizes and inter-arrival times of the aforementioned six time series, and also examine the cross-correlation between different price levels for each time series. The absence of correlation is necessary but not sufficient to show that successive observations of a random variable are independent. However, in our particular application setting, no correlation for both the observations and their common variants (e.g., square, inverse) should suffice as an indication of independence, in the same spirit of [Cont and De Larrard, 2012].

Autocorrelation

We first study the autocorrelations regarding size, and the results show that all autocorrelation coefficients are significantly close to zero for all time-lag separations by the Durbin-Watson test. We also investigate the autocorrelation of the square and inverse of the size. The results remain the same. Thus, we conclude that the order sizes of each of the order book events are all independent.

Doing a similar check for inter-arrival times, we find that the sequences of inter-arrival times for market buy and sell orders are positively autocorrelated with autocorrelation coefficient around 0.2, but the Durbin-Watson statistic is not statistically significant. The inter-arrival times of limit orders and cancellations are significantly positively correlated but the correlation coefficients are both smaller than 0.1. As mentioned earlier, such small autocorrelations can be ignored when we aggregate at the second level. Moreover, note that the large number of observations in the high-frequency data induces narrow confidence bands and spurious significance; thus when the number of observations is large, statistically significant autocorrelations do not indicate practical significance if the correlations are very small. Therefore, we can state that the arrivals of all events are also independent.

Cross-Correlation

When we examine the cross-correlation between different time series, we use the Spearman's coefficient to measure correlation, where $+1$ and -1 represent strong positive and negative correlation respectively, and 0 represents no correlation. The result shows that the Spearman's correlation coefficients, with p-value smaller than 0.05, are all very close to zero (smaller than 0.01). Thus, we conclude that the order sizes and the arrivals of these six order book events are pairwise uncorrelated/independent; and the sizes and arrivals of limit orders and cancellations at different price levels are uncorrelated as well.

3.2 Distribution Fitting

Traditionally, to solve a MDP, we need information on the transition matrix. To this end, we investigate the distribution of order size and inter-arrival time for market orders, limit orders, and cancellations on the bid and ask sides. After dropping the last 1% outliers in the dataset, we try more than 50 common discrete and continuous distributions to fit the data, but it turns out that the p-values of the chi-squared test or the KS-test for all the fits are close to zero, and the sums of squared errors of prediction (SSE) are much greater than 1, which are both indicative of very poor fits. We also considered estimating an empirical distribution, but that did not pass statistical tests, due to the heavy-tail pattern of our data. Hence, it is difficult to estimate the MDP transition matrix.

4 Q-learning Model

Our statistical analysis in Section 3 suggests that first estimating transition probabilities for the MDP in Section 2, and next applying standard MDP solution techniques will not yield satisfactory results. This observation motivates us to consider a stochastic iterative (also called stochastic approximation) method called Q-learning. Q-learning is a model-free algorithm which can be applied to obtain an optimal control policy for an MDP when the transition rewards and the transition probabilities are unknown. Previous works, such as [Bertsekas and Tsitsiklis, 1996; Tsitsiklis, 1994; Jaakkola *et al.*, 1994], have shown the convergence property of Q-learning.

However, [Powell, 2007] and others have observed that the Q-learning algorithm only works well in small state and action spaces, and even in modest spaces, the performance may not be good. The LOB state variable, (R_{tp}^1, R_{tp}^2) for any given $t \in \mathcal{T}$, has $n = 20$ price levels, and, for each price level, two possible values for R_{tp}^1 and an infinite number of possible values for R_{tp}^2, which we truncate to size 1000 for implementation purposes. This results in a lower bound on the LOB state space size that is $1000^{20} = 1 \times 10^{60}$, and this does not account for the history-dependent inventory level. That large number motivates us to create a state aggregation function, in the same spirit as [Pepyne *et al.*, 1996], allowing us to reduce the original large-scale MDP to a much smaller, and more easily implementable, size.

4.1 Aggregation Method

From the extensive literature on market microstructure, such as [Cartea and Jaimungal, 2016; Spooner *et al.*, 2018; Cartea *et al.*, 2018], these are some attributes commonly used to describe the condition of the market and the decision-maker: the imbalance of the book size on both the bid and ask side, the magnitude of the market price movement, the trade volume, the relative strength index (RSI), the net amount bought and/or sold, and the current PnL. After experimenting with many different combinations of the aforementioned state attributes, we find the best results using the five attributes listed below. Note that the attributes used to describe the condition of the market (the first three below) come directly from the LOB data, whereas the attributes used to describe the decision-maker (the last two below) are history-dependent

and must be updated in real-time as the market-making firm executes trades. At each time $t \in \mathcal{T}$,

- $bidSpeed(BS) \in \{0, 1\}$: indicates whether the market sell orders exceed the book size at the best bid price, and is defined as
$$BS := \mathbb{1}\{\hat{D}_t^{MS} > R^1_{t\beta_{R_t}} + R^2_{t\beta_{R_t}}\}.$$
- $askSpeed(AS) \in \{0, 1\}$: indicates whether the market buy orders exceed the book size at the best ask price, and is defined as
$$AS = \mathbb{1}\{\hat{D}_t^{MB} > R^1_{t\alpha_{R_t}} + R^2_{t\alpha_{R_t}}\}.$$
- $avgmidChangeFrac(MF) \in \{0, \pm 1, \pm 2\}$: characterizes the relative change in the average mid price[6] from time period $t-1$ to t compared to the range in the mid price over these two time periods, defined as f_t. The parameter $f \in [-1, 1]$ determines the direction of the mid price movement (positive or negative), and if that movement is large or small; that is, $|MF| = 2$ if $|f_t| > f$, $|MF| = 1$ if $|f_t| \in (0, f]$, and $MF = 0$ if $f_t = 0$.
- $invSign(IS) \in \{0, \pm 1, \pm 2\}$: characterizes the side and magnitude of open positions, inv_t, as defined in (7). The state is oversold if $inv_t < 0$ and overbought if $inv_t > 0$. The parameter $I \in (0, \infty)$ determines if the firm is oversold or overbought by a large amount, in which case $|inv_t| > I$ and $|IS| = 2$, or by a small amount, in which case $|inv_t| \in (0, I]$ and $|IS| = 1$. The balanced state $IS = 0$ occurs when $inv_t = 0$.
- $cumPnL(CP) \in \{0, 1\}$: indicates whether the cumulative PnL, defined from equation (6) as
$$pnl_t := \sum_{i=0}^{t} C(R_t, A_t, \hat{D}_t^{MB}, \hat{D}_t^{MS}), \qquad (9)$$
is large or small, as determined from the parameter $P \in (-\infty, \infty)$. Specifically, $CP = 1$ if $pnl_t \leq P$ and $CP = 0$ if $pnl_t > P$.

The attributes $bidSpeed$ and $askSpeed$ together characterize market volatility. For instance, $bidSpeed = 1$ and $askSpeed = 0$ indicate a sell-heavy market, which might be followed by a decrease of market (bid and ask) prices; thus it is suggestive to place limit sells and cancel limit buys. The $avgmidChangeFrac$ measures the magnitude and direction of mid price changes, which provide signals about whether orders should be placed on the bid or ask side. The $invSign$ indicates if there is a high mismatch between the amount bought and sold, and could be used to restrict order placement on the overbought/oversold side, even when conditions are favorable otherwise. Lastly, the $cumPnL$ monitors the decision-maker's profitability in real time, and could induce more conservative behavior in the face of large losses, or more risky behavior in the face of large gains. Overall, there are three parameters that must be configured: f, I, and P.

[6]We choose mid price rather than market bid and/or market ask prices to represent the state of the book due to the fact that the market bid and ask prices always move in the same direction.

4.2 Q-learning Model

Given an LOB state $R_t \in \mathcal{R}$ defined in Section 2.1, inventory value inv_t defined in (7), and cumulative PnL pnl_t defined in (9), the state space aggregation function is

$$G(R_t, inv_t, pnl_t) := (BS_t, AS_t, MF_t, IS_t, CP_t), \text{for all } t \in \mathcal{T}. \qquad (10)$$

Let ran(G) be the range of the state space aggregation function G, and denote the aggregated state space by $\mathcal{G} :=$ ran(G). Then, it is straightforward that the size of the aggregated state space is $2 \times 2 \times 5 \times 5 \times 2 = 200$, which is small enough that the Q-learning algorithm has good performance.

We restrict the admissible action space in Section 2.2 to prevent placing orders when the $invSign$ is either $+2$ or -2, meaning we have had many more limit buy orders execute than limit sell orders or vice versa. This is because there is a high level of risk associated with such imbalance. For a given aggregated state $s \in \mathcal{G}$, let s_{IS} denote the fourth component of the right-hand-side of (10). The restricted admissible action space is

$$\mathcal{A}_s := \begin{cases} \{(0,0),(1,0)\}, & \text{if } s_{IS} = -2, \\ \{(0,0),(0,1)\}, & \text{if } s_{IS} = +2, \\ \{(0,0),(0,1),(1,0),(1,1)\}, & \text{otherwise.} \end{cases} \qquad (11)$$

As in [Watkins and Dayan, 1992], the Q factor $Q(s, a)$ represents the value of taking action a when in aggregated state s. The recommended action when in state s is

$$a^*(s) = \arg\max_{a \in \mathcal{A}_s} Q(s, a), \qquad (12)$$

and we record these in a lookup table called Q table. The resulting size of the Q table used to look up the recommended action associated with any given state is $2 \times 2 \times 5 \times 3 \times 2 \times 4 + 2 \times 2 \times 5 \times 2 \times 2 \times 2 = 640$.

4.3 Algorithm

The core of a Q-learning algorithm is the iterative updates of Q factors based on sample paths. However, since we have no information about real-time inventory and PnL in our dataset, for a given aggregated state, we select a sample path in the dataset based on the first three dimensions of the state and randomly set an inventory level and PnL value consistent with the $invSign$ term and the $cumPnL$ term, respectively. For ease of exposition, we define a sampling-related aggregation function by, for all $R_t \in \mathcal{R}$,

$$\Gamma(R_t) := (BS_{R_t}, AS_{R_t}, MF_{R_t}). \qquad (13)$$

For any aggregated state $s \in \mathcal{G}$, define $\mathcal{M}_s := \{R_t \in \mathcal{R} : \Gamma(R_t) = (s_{BS}, s_{AS}, s_{MF})\}$, where s_{BS}, s_{AS}, s_{MF} denote the first three components of the right-hand-side of (10), to be the set of full states that can be mapped into aggregated state s. Let $\tau_s := \{t : R_t \in \mathcal{M}_s\}$ be the set of timestamps at which the full state of the order book is an element of the set $\mathcal{M}_s$. Suppose a sample path ω starts at time $t \in \tau_s$. Then, we denote the immediate exogenous information (i.e., adds, cancels, and trades) in the following one second by $\hat{O}(\omega)$, $\hat{C}(\omega)$, $\hat{D}^{MB}(\omega)$, and $\hat{D}^{MS}(\omega)$. Let Ω^s be the set of all possible sample paths for aggregated state s.

Algorithm 1 Q-learning algorithm pseudocode

Initialization: Set $Q_0(s,a) = 0$, $\alpha_0(s,a) = \alpha_0$, $K_0(s,a) = 0$, for all $s \in \mathcal{G}$, $a \in \mathcal{A}_s$, and stopping criterion $\bar{N}$; and set $n = 0$.
while $n = 0, 1, 2, \ldots$ **do**
  Randomly select $S_n^Q \subseteq \{(s,a) | s \in \mathcal{G}, a \in \mathcal{A}_s\}$;
  for $(s,a) : s \in \mathcal{G}, a \in \mathcal{A}_s$ **do**
    if $(s,a) \in S_n^Q$ **then**
      (i) Randomly select a sample path $\omega_n^s \in \Omega^s$, and denote its initial full state by R_n^s. Randomly set an exact value for inventory level and cumulative PnL based on s_{IS} and s_{CP}, denoted by inv_n^s and pnl_n^s. (ii) Update the full state to R_{n+1}^s, as detailed in equations (1)–(4) in Section 2. Denote the updated inventory level and cumulative PnL by inv_{n+1}^s and pnl_{n+1}^s. (iii) Translate the updated full state $(R_{n+1}^s, inv_{n+1}^s, pnl_{n+1}^s)$ into aggregated state by $\bar{s} = G(R_{n+1}^s, inv_{n+1}^s, pnl_{n+1}^s)$. (iv) Update $K_n(s,a) = K_{n-1}(s,a) + 1$.
      (v) Compute Q_{n+1} by Equation (14) with $\alpha_n(s,a) = \frac{\alpha_0}{1+K_n(s,a)}$.
    else
      $Q_{n+1}(s,a) = Q_n(s,a)$;
      $K_n(s,a) = K_{n-1}(s,a)$.
    end if
  end for
  if $n > \bar{N}$ **then**
    break **while**.
  else
    $n = n + 1$.
  end if
end while
return For any $R_t \in \mathcal{R}$ in which the decision-maker's current inventory and PnL is inv and pnl, the optimal action is: $\arg\max_{a \in \mathcal{A}_s} Q_{n+1}(G(R_t, inv, pnl), a)$.

The Q-learning algorithm, more specifically detailed in the pseudocode in **Algorithm 1**, follows the steps below. We set the maximum number of iterations $\bar{N}$ to be large enough such that the resulting objective function has become stable. At each iteration, we make a uniform random selection of some certain number of state-action pairs to update. For each aggregated state s of interest, we randomly select a sample path $\omega \in \Omega^s$ that has initial full state $R(\omega) \in \mathcal{M}_s$. Then we update the full state based on the action and exogenous information from the sample path, calculate the inventory level $inv(\omega)$ and cumulative PnL, and further translate the updated full state into an aggregated version, denoted by $\bar{s}$, using aggregation function G defined in equation (10). Then, we can update the Q factor according to the update function

$$Q_{n+1}(s,a) = (1 - \alpha_n(s,a)) \cdot Q_n(s,a) + \alpha_n(s,a) \cdot (V(R(\omega), a, \hat{D}^{MB}(\omega), \hat{D}^{MS}(\omega), inv(\omega)) + \gamma \max_{v \in \mathcal{A}_{\bar{s}}} Q_n(\bar{s}, v)), \quad (14)$$

where $V(R(\omega), a, \hat{D}^{MB}(\omega), \hat{D}^{MS}(\omega), inv(\omega))$ is as defined in equation (8), and the learning rate $\alpha_n(s,a)$ is as defined in the Q-learning algorithm pseudocode.

4.4 Resulting Q Factors

The trading policy learned from the Q-learning algorithm can be summarized by the following several rules: (1) it is profitable to place limit orders on the more active side—that is, it is better to add a limit buy order on a sell-heavy market, and vice versa; (2) market-making is not directional—that is, movements in the mid price do not affect the decisions regarding where to place an order, which aligns with the market-making strategy structure in [Menkveld, 2013]; (3) the optimal market-making strategy depends on the level of inventory, and maintaining inventory near zero is preferable, which is consistent with [Guilbaud and Pham, 2013]; (4) market makers control their cumulative PnL by canceling all orders in many cases when cumulative PnL is low.

5 Results

5.1 Performance Evaluation

Our dataset provides information on the LOB during time periods at which our partner market-making firm was trading, but we do not know the strategies the firm was using. In other words, we do not know what is called the behavior policy in the off-policy evaluation literature. The implication is that we cannot use any of the three main off-policy evaluation methods: *direct method* [Bertsekas *et al.*, 1995; Lagoudakis and Parr, 2003; Sutton and Barto, 2018], *importance sampling*[Swaminathan and Joachims, 2015], and *doubly robust method* [Dudík *et al.*, 2014; Jiang and Li, 2015; Robins *et al.*, 1994].

Fortunately for evaluation purposes, our partner market-making firm represents a small percentage of the trades recorded in the LOB. As a result, it is reasonable to assume that the orders placed by our partner market-making firm are not unduly influencing the other players in the market, and so the information we see recorded in the LOB regarding the arrival of market buy and sell orders, and the arrival of limit buy and sell orders, as well as cancellations, by other market participants should not change too much when our partner market-making firm trades according to a different strategy. This suggests that a straightforward backtest evaluation of the profit that would have been made, and the associated Sharpe ratio, is a representative test of the performance of our proposed trading strategy, in which orders are executed according to the Q-table output from the Q-learning algorithm.

Our partner market-making firm developed its own backtester to evaluate any proposed trading strategy. This is accomplished by using historical data to reconstruct the trades that would have occurred in the past using the proposed trading strategy, and recording the resulting cumulative PnL and associated Sharpe ratio. We first used the backtester to conduct in-sample experiments, and to use the results of those experiments to set algorithm parameters (specifically, the f, I, and P thresholds defined in Section 4.1). We further used the backtester to set two external controls, one that forces closing all open positions if the PnL becomes too negative, and another that forces closing if the maximum drawdown becomes too large. After finalizing our algorithm's parameters and the aforementioned external control parameters, we tested once on the out-of-sample data in the backtester.

The out-of-sample performance results of our algorithm in the backtester resulted in an average daily PnL with three orders of magnitude, and a Sharpe ratio above 3. This passed the firm's test standards. Consequently, our partner market-making firm desired to put our algorithm into production.

5.2 Benchmarks

To further anchor the performance of our algorithm in the literature, we compare with a set of benchmarks as below.

From [Spooner *et al.*, 2018], [Lim and Gorse, 2018] and [Doloc, 2019], common benchmarks include fixed spread-based strategies, random strategies, and the Avellaneda-Stoikov strategy [Avellaneda and Stoikov, 2008]. The fixed spread-based strategy that provides the most relevant benchmark is the one that at all times has limit orders resting at the best bid and ask prices. The most natural random strategy benchmark is the one that flips two fair coins in each time period, one to decide whether or not to have an order resting at the best bid price and the other for the best ask price. The Avellaneda-Stoikov strategy is not relevant for us because orders may be placed at prices other than the best bid and ask prices.

Figure 3 compares our Q-learning algorithm against the aforementioned two common benchmarks, as well as against our partner firm's implemented trading strategy. For this, there is no need to conduct in-sample testing because the parameters of the benchmark strategies are all fixed, and we use the same external controls[7] for the benchmark strategies as we used when we implemented our Q-learning algorithm. Figure 3 clearly shows that our algorithm attains the highest cumulative PnL over the one-month out-of-sample trading period. As for the Sharpe ratio, the only benchmark strategy for which the Sharpe ratio is positive is our partner firm's implemented trading strategy; however, the ratio is still smaller than our proposed Q-learning strategy.

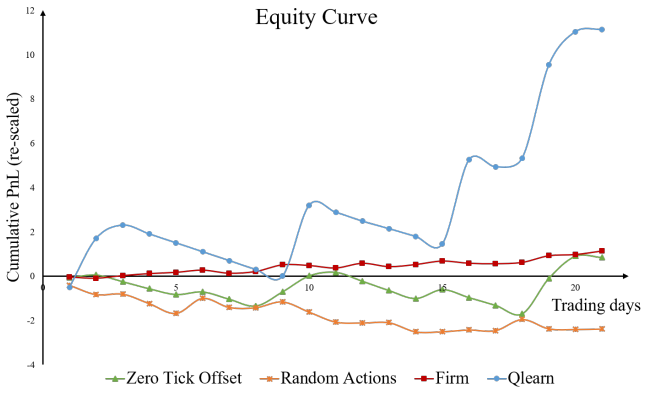

Figure 3: Out-of-Sample Cumulative PnL, with re-scaled y-axis to protect confidentiality

[7]Recall from Section 5.1 that the two external controls force trading to stop if the PnL becomes too negative or if the maximum drawdown becomes too large.

6 Future Work

From Figure 3, we note that under our Q-learning algorithm, we may encounter the following phenomenon: several days in a row we lose money, followed by one day in which we make a large amount of money. This is not ideal from a risk management perspective and motivates our main desire in future work, to smooth out the resulting equity curve. On the days with losses, often we could have been profitable on that day had we locked in the profit earlier, and closed down trading. However, we are still working to develop an algorithmic approach to decide when to close down trading, resulting in the length of the time horizon T being a random variable.

Acknowledgments

We would like to extend our deepest gratitude to Volodymyr Babich, Nathan Kallus, Melanie Rubino for their helpful comments.

References

[Avellaneda and Stoikov, 2008] Marco Avellaneda and Sasha Stoikov. High-frequency trading in a limit order book. *Quantitative Finance*, 8(3):217–224, 2008.

[Bertsekas and Tsitsiklis, 1996] Dimitri P Bertsekas and John N Tsitsiklis. *Neuro-dynamic programming*, volume 5. Athena Scientific Belmont, MA, 1996.

[Bertsekas *et al.*, 1995] Dimitri P Bertsekas, Dimitri P Bertsekas, Dimitri P Bertsekas, and Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

[Cartea and Jaimungal, 2016] Álvaro Cartea and Sebastian Jaimungal. Incorporating order-flow into optimal execution. *Mathematics and Financial Economics*, 10(3):339–364, 2016.

[Cartea *et al.*, 2018] Álvaro Cartea, Ryan Donnelly, and Sebastian Jaimungal. Enhancing trading strategies with order book signals. *Applied Mathematical Finance*, 25(1):1–35, 2018.

[Cont and De Larrard, 2012] Rama Cont and Adrien De Larrard. Order book dynamics in liquid markets: limit theorems and diffusion approximations. *Available at SSRN 1757861*, 2012.

[Doloc, 2019] Cris Doloc. *Applications of Computational Intelligence in Data-Driven Trading*. John Wiley & Sons, 2019.

[Dudík *et al.*, 2014] Miroslav Dudík, Dumitru Erhan, John Langford, Lihong Li, et al. Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4):485–511, 2014.

[Guilbaud and Pham, 2013] Fabien Guilbaud and Huyen Pham. Optimal high-frequency trading with limit and market orders. *Quantitative Finance*, 13(1):79–94, 2013.

[Jaakkola *et al.*, 1994] Tommi Jaakkola, Michael I Jordan, and Satinder P Singh. Convergence of stochastic iterative dynamic programming algorithms. In *Advances in neural information processing systems*, pages 703–710, 1994.

[Jiang and Li, 2015] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. *arXiv preprint arXiv:1511.03722*, 2015.

[Lagoudakis and Parr, 2003] Michail G Lagoudakis and Ronald Parr. Least-squares policy iteration. *Journal of machine learning research*, 4(Dec):1107–1149, 2003.

[Lim and Gorse, 2018] Ye-Sheen Lim and Denise Gorse. Reinforcement learning for high-frequency market making. In *ESANN*, 2018.

[Menkveld, 2013] Albert J Menkveld. High frequency trading and the new market makers. *Journal of financial Markets*, 16(4):712–740, 2013.

[Pepyne *et al.*, 1996] David L Pepyne, Douglas P Looze, Christos G Cassandras, and Theodore E Djaferis. Application of q-learning to elevator dispatcidng. *IFAC Proceedings Volumes*, 29(1):4742–4747, 1996.

[Powell, 2007] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[Robins *et al.*, 1994] James M Robins, Andrea Rotnitzky, and Lue Ping Zhao. Estimation of regression coefficients when some regressors are not always observed. *Journal of the American statistical Association*, 89(427):846–866, 1994.

[Spooner *et al.*, 2018] Thomas Spooner, John Fearnley, Rahul Savani, and Andreas Koukorinis. Market making via reinforcement learning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 434–442. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Swaminathan and Joachims, 2015] Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. In *advances in neural information processing systems*, pages 3231–3239, 2015.

[Tsitsiklis, 1994] John N Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16(3):185–202, 1994.

[Watkins and Dayan, 1992] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.