Optimal Order Execution with Reinforcement Learning

Financial Engineering Department

March 2020 - ver. 1.0

Contents

1 Reinforcement learning for optimal execution

The problem of optimized trade execution in financial context has been extensively studied in the literature (e.g. [1]-[5], [7], [11]). The goal in this problem is to find the optimal strategy to execute a big order within a fixed time horizon, trying to minimize transaction costs, risk and market impact. Many works have addressed this issue using algorithms of machine learning (e.g. [6], [8], [10]).
In this report we present an analysis based on a reinforcement learning (RL) technique adapted from [8], with the aim of comparing the performance of our strategy with that of a Surrogate Smart Order (SSO) policy.

2 Experimental setup

In this section we describe in detail the experiments carried out, with the specification of the dataset used, the methodology adopted and the algorithm implemented.

2.1 Dataset

The analysis has been performed using two different stocks: NL0010877643 (FCA) and IT0003132476 (ENI). The training set for the RL algorithm is constituted by the 1-second sampled FTSE MIB limit order book data of 2018, composed of five levels of prices and volumes for each side and each timestamp, and limited to the continuous trading period only. Figure 1 shows an extract from the book.

TYPE	ask										bid			
LEVEL	0		1		2		3		4		0		1	
PV	price	volume	price	volume	price	volume	price	volume	price	volume	price	volume	price	volume
DATETIME														
2018-01-02 09:05:00.972	13.870	31337.0	13.880	16355.0	13.890	23958.0	13.900	10738.0	13.910	9760.0	13.850	42878.0	13.840	55590.0
2018-01-02 09:05:01.834	13.860	18552.0	13.870	31330.0	13.880	15744.0	13.890	23958.0	13.900	10738.0	13.850	42878.0	13.840	54890.0
2018-01-02 09:05:02.506	13.850	12444.0	13.860	6054.0	13.870	31043.0	13.880	20123.0	13.890	22746.0	13.840	51042.0	13.830	17711.0
2018-01-02 09:05:03.902	13.860	1884.0	13.870	30610.0	13.880	20556.0	13.890	22746.0	13.900	7138.0	13.850	1500.0	13.840	32057.0
2018-01-02 09:05:04.727	13.860	1884.0	13.870	31060.0	13.880	20692.0	13.890	22746.0	13.900	7138.0	13.850	2500.0	13.840	32057.0

Figure 1: Portion of the FTSE MIB limit order book of 2018.

The test set is made up of the orders released by a "List customer specialist in equity markets" in 2018, of which we present a sample in Table 1. Each of them has been split into a certain number of child orders, following the volume curve effectively produced for the corresponding ISIN between the order start and end times (see Sec. 2.1.1 for the details). As displayed in the table, the order features are: the client ID, the quantity q to be executed in each interval, the total size Q[1], the start and end times, the ISIN, the date of execution and the verb ('buy' or 'sell'). The orders such that $q/Q > 0.5$ have been discarded.

[1]If Q is not a multiple of q, the quantity executed in the last interval is less than q.

ID	q	Q	START	END	ISIN	DATE	VERB
11131	1000	20000	1/23/2018 10:19	1/23/2018 17:00	IT0003132476	1/23/2018	sell
11409	200	17200	1/29/2018 9:34	1/29/2018 17:28	IT0003132476	1/29/2018	sell
13025	2000	142500	2/16/2018 10:45	2/16/2018 17:28	IT0003132476	2/16/2018	sell
15292	3170	103170	3/13/2018 15:10	3/13/2018 17:25	IT0003132476	3/13/2018	buy
17560	250	5000	4/9/2018 10:10	4/9/2018 12:00	IT0003132476	4/9/2018	sell
17879	4835	300000	4/11/2018 11:45	4/11/2018 16:25	IT0003132476	4/11/2018	buy
18698	3051	91200	4/20/2018 15:39	4/20/2018 17:28	IT0003132476	4/20/2018	sell
18698	3051	83900	4/20/2018 15:39	4/20/2018 17:28	IT0003132476	4/20/2018	sell
18698	3051	73900	4/20/2018 15:39	4/20/2018 17:28	IT0003132476	4/20/2018	sell
18698	3051	59000	4/20/2018 15:39	4/20/2018 17:28	IT0003132476	4/20/2018	sell
18698	3051	41100	4/20/2018 15:39	4/20/2018 17:28	IT0003132476	4/20/2018	sell
18697	2242	22700	4/20/2018 15:39	4/20/2018 17:28	IT0003132476	4/20/2018	sell

Table 1: Orders dataset example.

2.1.1 Volume curve construction and market VWAPs

The historical daily volume profiles represent how many shares are traded per day. We have carried out our analysis using the 'real' volume curves, realised on a certain day and measurable only at the end of the day, that is, *a posteriori*. These have been constructed starting from all the executions produced on that day for the ISIN in question. We have used these profiles to split each order at hand into various child orders, through the method explained below.

- We have built the cumulative volume curve produced on the order date for the corresponding ISIN, which represents the cumulated traded quantity from 0, at the beginning of the day, to 1, at the end of the day.
- We have cut this curve between the start and end times of the order and divided the ordinate interval of the resulting curve portion into n equal parts, where n is the result of the integer division between Q and q.
- We have found the corresponding abscissa values, t_0, t_1,..., t_n, which indicate the times within which each child order must be executed[2].

Furthermore, for each order, we have computed the volume-weighted average price of its total executions between the start and end times as:

$$\text{VWAP}_{\text{market}} = \frac{\sum_t q_t \cdot p_t}{\sum_t q_t}\,, \tag{1}$$

where q_t and p_t are respectively the volume and price of each execution, produced at time t.

2.2 Methodology

Our experimental methodology consists of the following steps.

1. Identification of the observed state variables derived from the order book and the actions of the training agent.
2. Partition of the training data into episodes and application of the RL algorithm to them, in order for the agent to learn an optimal execution strategy over the state space.
3. Based on the test set, comparison of the performance of our RL policy with that of the SSO strategy.

[2]The times t_i are such that in each interval $[t_i, t_{i+1}]$ the same quantity is exchanged; hence, the order intervals are "equal realised volume intervals".

2.3 RL environment

The elements composing the environment in which our RL agent moves are listed below.

- **States**. Each observed state is a discrete representation of the current configuration of the system. The state variables are: the *remaining time* t, which represents how much time is left until the end of the horizon T, the *remaining inventory* i, which indicates how much volume is left to execute of the total quantity I, and the *spread* s, which is the current bid-ask spread with maximum value S. For each stock analysed, we have performed a specific binning of the state variables, so as to find a compromise between including almost the entire ranges of values involved and limiting the computation time.

- **Actions**. In consequence of observing some state, the training agent chooses an action following some policy. The action space is constituted by limit order prices, measured relatively to the current best (ask or bid) price and expressed in tick sizes, at which we place our remaining shares. For the sell (buy) case, action a means that we are placing all the unexecuted inventory at price $ask - a$ ($bid + a$) through a limit order.

- **Rewards**. Doing an action in a given state results in an immediate reward for the agent. We have defined this reward as the implementation shortfall (IS):

$$\mathrm{IS} = \pm \frac{\sum_{t=0}^{T} q_t \cdot \mathrm{VWAP_t} - q_{ex} \cdot M + 0.6 \cdot n_{exec}}{Q_0 \cdot M} \times 10^4 \,, \tag{2}$$

  where q_t and VWAP_t are respectively the executed volume and the corresponding volume-weighted average price at time t, q_{ex} is the quantity executed due to the action performed, M is the mid-spread price ($(ask + bid)/2$) at the beginning of each episode, and Q_0 is the initial volume to be executed. The sign is added to take into account the verb (+ for buy, − for sell). Furthermore, the term $0.6 \cdot n_{\mathrm{exec}}$ represents the cost deriving from the total number of executions n_{exec}, being 0.6 euros the fee required by the market for each execution produced[3].

2.4 RL algorithm

The algorithm implemented is based on the following two assumptions:

1. the actions of the RL agent do not affect the behavior of all the other market traders;

2. the optimal action in a state at a given time is independent of the actions in all the previous states encountered (Markov property [9]).

The first hypothesis ensures that the order book evolves independently of the actions of our policy, while the second one guarantees that, once we have optimized all the final states (at $t = 0$), we can determine the optimal actions for all the immediately preceding states, and so on until $t = T$. The learning thus proceeds backwards in time.

2.4.1 Training phase

The training procedure is a sort of Q-learning and is bin-based, so that it can move backwards in time, relying, in every bin, on the information stored in the subsequent bin. Indeed, we have partitioned the training data into episodes, and we have divided every episode into a certain

[3]Every time the book timestamp changes and/or the limit order matches a book level of the opposite side, one execution is produced.

number of time bins with variable length (for the details, see Sec. 2.6), with every bin consisting of intervals of 30 seconds each. At the beginning of each interval, the agent observes the state and submits a limit order[4], going on like this until it executes all the remaining quantity or reaches the end of the bin. If, after one or more bin "steps", it gets to the end of the episode with some unexecuted shares, it then places a market order on the opposite side of the book to force their execution.
The learning algorithm works as follows, for every t, episode and i:

- randomly select time and inventory within the relative bins, on a grid with a spacing of 30" and 50 shares respectively (in order to cover a wide range of values);
- synchronise the agent at the book timestamp corresponding to the given time and define the current state s;
- define the admissible actions in the state s;
- try every possible action a in the state s, which results in a reward $r(s,a)$[5] and brings the system to a new state s';
- update the expected reward $Q(s,a)$ deriving from taking each action and following the optimal policy subsequently, through the following rule:

$$Q(s,a) = Q(s,a) + \frac{1}{n(s,a)+1}\left[r(s,a) + \min_a Q(s',a) - Q(s,a)\right], \tag{3}$$

  where $n(s,a)$ is the number of times that the agent has already tried action a in the state s. The Q-values are then written on a Q-table, which results in a (state $\times$ action) matrix.

It is worth noting that, apart from the random choice of time/inventory values, the learning procedure as it is turns out to be fully deterministic.

2.4.2 Test phase

For testing our RL strategy, we have divided the orders dataset into episodes, following the same method adopted for the training phase. Each episode is constituted by an order interval, where the quantity q must be executed. For each episode, the test procedure is made up of the following steps:

- synchronise the agent at the book timestamp corresponding to the start of the interval and define the current state s;
- define the admissible actions in the state s;
- based on the trained Q-table, select the action a giving the minimum Q-value in the state s[6];
- with the selected action, perform a bin "step", which results in some executed quantity and realized countervalue. These values are summed up to give quantity and countervalue relative to each interval, q_{int} and ctv_{int}.

At the end of this procedure, for each order we have been able to compute the volume-weighted average price, $\text{VWAP} = \sum \text{ctv}_{\text{int}} / \sum q_{\text{int}}$, where the sums are all over the order intervals.

[4] A limit order is executed if its price meets the opposite side of the book.

[5] The bin reward $r(s,a)$ is calculated as the sum of all the immediate rewards obtained through each 30" step.

[6] If the sum of the matrix elements in correspondence of some state s is 0, the action is chosen randomly between the admissible ones.

2.5 Surrogate Smart Order

The strategy with which we have compared our RL policy is a sort of Smart Order, that we have called Surrogate Smart Order (SSO). The SSO simulation consists in the execution of the quantity q of the child order at the end of each interval, by submitting a market order. This strategy differs from what the actual Smart Order policy does, since the latter relies on placing subsequent limit orders at the current best price within each interval, with the hope that one or more of them will be "hit" by some market order, and eventually going to the market, at the end of the interval, with any unexecuted shares. In a similar way to the RL strategy, we have calculated the volume-weighted average prices deriving from the SSO policy, for each of the orders in question.

2.6 Parameter setting

The values of the tick size, and consequently of the bid-ask spread, along with the features of the orders concerned, depend on the particular stock. Therefore, for some parameters involved, we have adopted specific valorisations for each ISIN, as we show below:

- inventory bins (in shares): [0, 100, 300, 600, 1000, 2000, 5000] ($I = 6$), for FCA; $[0, 100, 300, 600, 1000, 2000, 8000]$ ($I = 6$), for ENI;
- time bins (in minutes): [0, 1, 3, 6, 10, 20, 50] ($T = 6$), for both;
- spread bins (in tick sizes): [0, 1, 2] ($S = 2$), in the period 01/01/2018 - 26/01/2018, for both; [0, 1, 2, 3, 4, 5] ($S = 5$), in the period 29/01/2018 - 31/12/2018, for both;
- actions (in tick sizes): [-3, -2, -1, 0, 1, 2, 3], for both[7];
- tick size: 0.01, in the period 01/01/2018 - 26/01/2018, for both; 0.002, in the period 29/01/2018 - 31/12/2018, for both.

We have performed the training/test separately for sell and buy cases and distinguishing between the periods with different tick size and spread.

1) For orders between 01/01/2018 and 26/01/2018: training from 01/01/2018 to 26/01/2018 for FCA/ENI; consecutive episodes in a day start every 15 minutes, with each episode lasting 50 minutes.

2) For orders between 29/01/2018 and 31/12/2018: training from 01/11/2018 to 31/12/2018 for FCA and from 01/07/2018 to 31/08/2018 for ENI[8]; consecutive episodes in a day start every 30 minutes, with each episode lasting 50 minutes.

3 Results

In this section we present the outcomes of our analysis. As criterion for comparing the performances of the two implemented strategies, we have chosen the total Profit and Loss (P&L) deriving from each of them, computed with respect to the market. In order to do this, we have

[7] The action space has been defined on the base of the price excursions and the spread width in the book.

[8] Performing the learning from 29/01/2018 to 31/12/2018 would have cost an enormous computation time; thus, we have restricted our training to a (sufficiently long) period of two months where the price trend is almost stable, in order not to affect the advantage of a strategy over the other.

calculated the P&L relative to each order, both in cash (euro) and in basis point units, through the following formulas:

$$\text{P\&L}_{\text{strat}}\,[\text{cash}] = Q \cdot (\text{VWAP}_{\text{strat}} - \text{VWAP}_{\text{market}}) \tag{4}$$

and

$$\text{P\&L}_{\text{strat}}\,[\text{bps}] = \frac{\text{VWAP}_{\text{strat}} - \text{VWAP}_{\text{market}}}{\text{VWAP}_{\text{market}}} \times 10^4\,, \tag{5}$$

where Q is the total order volume, while $\text{VWAP}_{\text{strat}}$ and $\text{VWAP}_{\text{market}}$ are the volume-weighted average prices obtained through the strategy (RL or SSO) and the market respectively. Below we show the tables with the P&L statistics elaborated for FCA (Figures 2 and 3, for SSO and RL respectively) and ENI (Figures 4 and 5, for SSO and RL respectively).

	p&l statistics in cash:	p&l statistics in bps:	p&l total cash in euro:	costs in euro (60 cents * exec):
count	217	217	72964.38	10540.2
mean	336.241397	2.471927		number of executions:
std	368.833786	4.181892		17567
min	-277.784768	-19.302695		
25%	69.621988	1.523150		
50%	241.026586	2.309777		
75%	462.138658	2.929751		
max	2474.534869	32.269534		

Figure 2: P&L statistics for NL0010877643: SSO strategy versus market.

	p&l statistics in cash:	p&l statistics in bps:	p&l total cash in euro:	costs in euro (60 cents * exec):
count	217	217	64267.22	13451.4
mean	296.162305	1.054253		number of executions:
std	405.385093	6.152339		22419
min	-649.312359	-48.167145		
25%	38.986730	0.919081		
50%	230.845891	2.181249		
75%	495.240235	3.238691		
max	2770.542869	19.389343		

Figure 3: P&L statistics for NL0010877643: RL strategy versus market.

	p&l statistics in cash:	p&l statistics in bps:	p&l total cash in euro:	costs in euro (60 cents * exec):
count	342	342	155770.62	14299.8
mean	455.469641	2.503333		number of executions:
std	737.182183	2.931764		23833
min	-149.264962	-9.587167		
25%	52.195862	1.246355		
50%	190.628307	1.985558		
75%	750.429485	3.141277		
max	10335.032607	20.666150		

Figure 4: P&L statistics for IT0003132476: SSO strategy versus market.

	p&l statistics in cash:	p&l statistics in bps:	p&l total cash in euro:	costs in euro (60 cents * exec):
count	342	342	111816.31	19563.0
mean	326.948273	0.954062		number of executions:
std	616.994242	5.639971		32605
min	-845.304196	-78.308911		
25%	19.156009	0.438436		
50%	149.650420	1.463646		
75%	403.554999	2.700609		
max	6080.530607	22.881507		

Figure 5: P&L statistics for IT0003132476: RL strategy versus market.

As we note from the tables above, for every order dataset we have calculated the mean, the standard deviation, the minimum and maximum values, and the 25th, 50th and 75th percentiles of the order VWAPs (both in euros – *p&l statistics in cash* – and in basis points – *p&l statistics in bps*). Moreover, *count* is the total number of orders (both buy and sell) used for the test: 217 for FCA and 342 for ENI. The field *p&l total cash in euro*, computed as *count* $\times$ *mean* (in cash), indicates how much we spend on average using the RL/SSO strategy with respect to the market, while *costs in euro* is the cost deriving from the total order executions of the strategy. By comparing the results obtained for the two policies, we get:

- for FCA: *p&l total cash in euro* (RL - SSO) = -8697.16 €; *costs in euro* (RL - SSO) = 2911.20 €;
- for ENI: *p&l total cash in euro* (RL - SSO) = -43954.31 €; *costs in euro* (RL - SSO) = 5263.20 €.

Thus, by considering the P&L only, with our RL strategy we earn about 12% and 28% more with respect to the SSO policy, for FCA and ENI respectively, while the cost coming from the order executions is always higher for the RL policy compared to the simulated smart order. In conclusion, the overall gain of the RL over the SSO strategy is of about 7% and 23%, respectively for FCA and ENI.
Though further analyses would be required to confirm these results, this first analysis shows that a reinforcement learning approach is capable of performing significantly better than a naive strategy like the Surrogate Smart Order.

References

[1] Almgren, R. and Chriss, N., "Optimal Execution of Portfolio Transactions". In *Journal of Risk*, 3, 5-39 (2000).

[2] Bertsimas, D. and Lo, A., "Optimal Control of Execution Costs". In *Journal of Financial Markets*, 1, 1-50 (1998).

[3] Cartea, Á. and Jaimungal, S., "Optimal execution with limit and market orders". In *Quantitative Finance*, 15, 12791291 (2015).

[4] Cheng, X., Di Giacinto, M., and Wang, T.-H., "Optimal Execution with Dynamic Risk Adjustment". Preprint. arXiv: 1901.00617v3 (2019).

[5] Gatheral, J. and Schied, A., "Optimal trade execution under geometric Brownian motion in the Almgren and Chriss framework". In *International Journal of Theoretical and Applied Finance*, 14, 225236 (2011).

[6] Hendricks, D. and Wilcox, D., "A reinforcement learning extension to the Almgren-Chriss framework for optimal trade execution". In *Computational Intelligence for Financial Engineering & Economics*, IEEE Conference, 457464 (2014).

[7] Obizhaeva, A. and Wang, J., "Optimal trading strategy and supply/demand dynamics". NBER Working Papers, No 11444, 2005.

[8] Nevmyvaka, Y., Feng, Y., and Kearns, M., "Reinforcement Learning for Optimized Trade Execution". In *Proceedings of the 23rd International Conference on Machine Learning*, 673-680. ACM. (2006).

[9] Puterman, M., "Markov Decision Processes", John Wiley and Sons, New York (1994).

[10] Ritter, G., "Machine Learning for Trading". In *Risk*, 30.10, 8489 (2017).

[11] Alexander Weiss, A., "Executing large orders in a microscopic market model". Preprint. arXiv:0904.4131 (2009).