Market Microstructure and Algorithmic Trading

Report

A.Y. 2024-2025

February 4, 2025

Contents

1 Summary

"Estimating permanent price impact via machine learning" by R. Philip presents the vector auto-regressive (VAR) models and modified Reinforcement Learning (RL) for estimating permanent price impact. The author noted that, in the literature, researchers estimate the private information content of a trade through the permanent price impact, derived from the impulse response function of a VAR model described as a multivariate linear time series that accounts for quote revisions. The issue lies in misspecification of the model, which does not captures nonlinear relationships between price impact and trade size, and may not be applicable to modern trading environment. In light of that, Hosbruck (1991a) suggests using polynomial terms to capture these nonlinear relationship. The author proposes to modify a RL framework for estimating the permanent price impact of a single trade, since its features make it flexible and useful when nonlinear processes have to be taken into account. In this model is included an iterative learning rule and, as we will see, by increasing the complexity of the model, differently from the traditional VAR framework, the recursive model yields correct conclusions. On the other hand, given that in the RL we take into account the depth imbalance, we decided to present a VAR model including also the depth imbalance, treating it as an endogenous variable. Our goal in this report is to replicate the authors' analysis on a different dataset. The structure of this report will follow this scheme: in Section 2 we provide a data analysis from a statistical point of view; in Section 3 we present the models; in Section 4 we present our data highlighting differences with the authors' dataset; in Section 5 we present the estimation and the results obtained by the estimated models; concluding remarks about the paper follow in Section 6.

2 Statistics of Data

We begin our analysis by investigating some statistical properties of our data, for the scope of the project the results are computed just for one day *(03/02/2020)*, for a more detailed description of the form of the data refer to the section **Data Description.** even if the investigation could be broadened with minimal change to the code provided.

We present two summary statistics tables, the first for the variable Price, that is the efficient price, i.e. the MidPrice. The second for logarithmic returns of the latter.

Table 1: Summary Statistics of the Prices

mean	SD	mode	median	kurtosis	skewness
167.103073	2.334130	165.020000	166.930000	-0.018781	0.604481

For the prices we plot the trend graph where the x-axis is the event-time, every new event makes the time series advance 1 time unit, this convention is kept for all the remaining of the work.

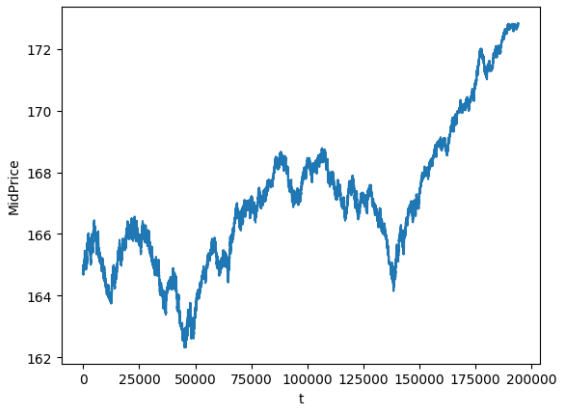

Figure 1: Trend plot of the Price variable. X-axis is the event-time.

If we perform an ADF test on the price series we get the following results:

Statistic	**Value**
ADF Statistic	-0.0912
p-value	0.9503
H0 (Stationarity Test)	Fail to reject
Conclusion	The time series is NOT stationary.
Critical Values	
1%	-3.4304
5%	-2.8616
10%	-2.5668

Table 2: ADF Test Results for variable MidPrice

It appears that the time series is not stationary, moreover it seems to be moved by purely randomness, this is endorsed by the plot of the ACF.

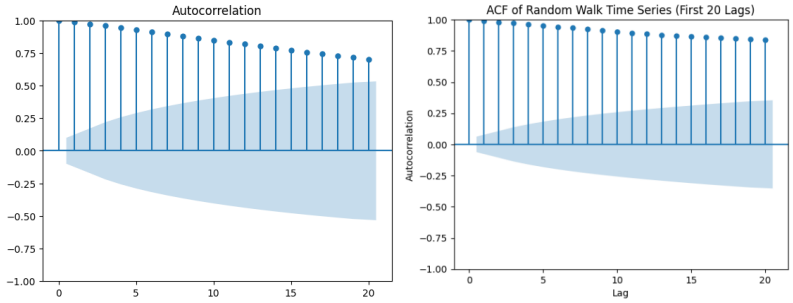

Figure 2: ACF plot, on the left the empirical from our data, on the right one calculated on a synthetic time series driven exclusively by a random walk.

Therefore, for our purposes we need to perform a differencing operation, to transform the time series in one that is at least stationary. So, as usual, we calculate the log returns,

$$r_t = \ln(m_t) - \ln(m_{t-1}),$$

and, as above, we provide the summary statistics table:

Table 3: Summary Statistics of the r_t

mean	SD	mode	median	kurtosis	skewness
$2.43 \cdot 10^{-7}$	0.000063	0.000000	0.000000	12.700168	0.534950

We also show the following plots:

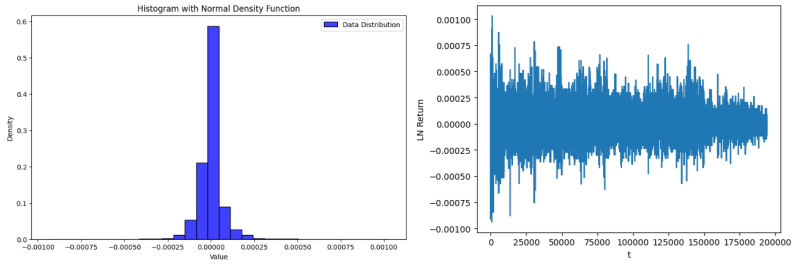

Figure 3: On the left the values frequency function. On the right the trend plot, where x-axis is, again, the event-time.

We perform an ADF test on the variable and find it to suggests to reject H0 and assume stationarity:

Statistic	**Value**
ADF Statistic	-90.7380
p-value	0.0
H0 (Stationarity Test)	Reject
Conclusion	The time series seems to be stationary.
Critical Values	
1%	-3.4304
5%	-2.8616
10%	-2.5668

Table 4: ADF Test Results for variable r_t

Moreover we present the plot of the ACF:

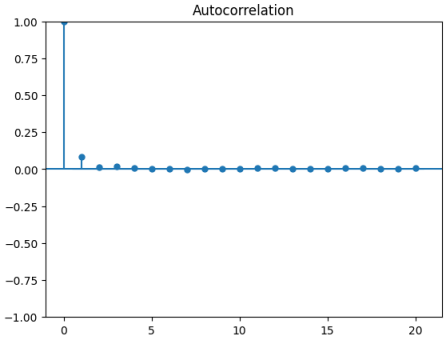

Figure 4: ACF plot for r_t.

Both of the tests are coherent with the fact that prices are random and returns are unpredictable. The log returns are likely independent. There is no apparent autocorrelation structure, meaning the returns are unpredictable from one period to the next. This is typical of markets that exhibit efficient market behavior where prices moves randomly due to new, unforeseen information entering the market. Since the ACF is zero for all lags beyond 0, this indicates there's no long-term

memory or dependency in the time series, which means the series doesn't exhibit characteristics like trends, cycles, or mean reversion.

Another ACF plot that is interesting to analyze is the one of the variable "Direction" (χ), that is defined as:

$$\chi_n = \begin{cases} +1 & \text{if } n \in \mathcal{B}, \\ -1 & \text{if } n \in \mathcal{S}, \end{cases}$$

where:

- $\mathcal{B}$ is the set of buy orders,
- $\mathcal{S}$ is the set of sell orders.
- n is the n-th order.

The plot is:

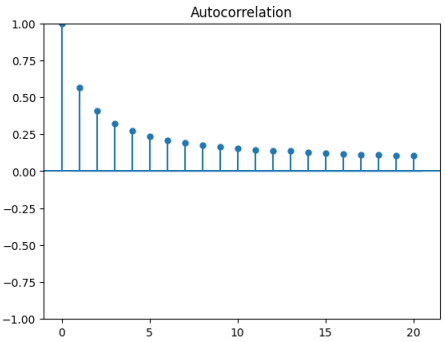

Figure 5: ACF plot for Direction.

An interesting aspect of this plot is the apparent exponential decay pattern. It would be valuable to determine the exponent that governs this decay process. We do this by setting the following equation:

$$y = x^{\alpha},$$

then to estimate alpha we perform a linear regression on the log/log transformation of the above:

$$\ln(y) = \alpha \ln(x),$$

where:

- $y = ACF(lag)$, the dependent variable in the regression.
- $x = lag$, the lag, without the first, that is equal to 0, to avoid divergences to infinity.

Statistic	**Value**
R-squared (uncentered)	0.990
Adj. R-squared (uncentered)	0.989
Variable	**Coefficient**
α	-0.7988
Standard Error	0.019

Table 5: OLS Regression Results without constant.

So we extrapolate that $\alpha \approx -0.8$ this is consistent to the result obtained by [**Bouchaud et al.,2004, Lillo and Farmer,2004**], moreover if we add a constant to the regression:

Statistic	**Value**
R-squared	0.998
Adj. R-squared	0.998
Variable	**Coefficient**
const	-0.5190
α	-0.5838
Standard Error Const	0.012
Standard Error α	0.006

Table 6: OLS Regression Results with constant.

Those α values are associated with a Hurst exponent H, calculated as:

$$H = 1 - \frac{|\alpha|}{2}$$

$$H = \begin{cases} 0.7081 & \text{with constant,} \\ 0.6 & \text{without constant.} \end{cases}$$

These values are consistent with the empirical findings of the works cited above that the time series of the orders signs is a long-memory process.

3 Model presentation

3.1 VAR models

We present a *VAR model with one market participant*, by introducing first the VAR system proposed by *Hasbrouck (1991a)*:

$r_t = \sum_{i=1}^{\infty} \alpha_i r_{t-i} + \sum_{i=0}^{\infty} \beta_i \chi_{t-i} + \epsilon_{1,t}$

$\chi_t = \sum_{i=1}^{\infty} \delta_i r_{t-i} + \sum_{i=1}^{\infty} \theta_i \chi_{t-i} + \epsilon_{2,t}$

Where we identify r_t as the change in the natural logarithm of the midpoint price following a trade at time t. The variable χ_t represents the trade indicator, which in our case corresponds to the *trade sign*. The immediate price impact of a single trade is given by β_0 while the permanent price impact is given by the vector moving average (MVA) representation of the VAR model:

$$\begin{bmatrix} r_t \\ \chi_t \end{bmatrix} = \begin{bmatrix} a(L) & b(L) \\ c(L) & d(L) \end{bmatrix} \begin{bmatrix} \epsilon_{1,t} \\ \epsilon_{2,t} \end{bmatrix},$$

where L is a lag operator. Then, we have $r^{\chi}_{t+\infty}$ measured by the impulse response function[1]: $r^{\chi}_{t+\infty} = \sum_{i=0}^{\infty} b_i \epsilon_{2,t}$. This means that we want to find the response of $r^{\chi}_{t+\infty}$ to a shock of the second variable, ie χ_t. So, here we are assuming that it exists a linear relation between the trade indicator variable and the midquote price.
But, as *Hasbrouck (1991a)* suggests, this model is misspecified when there is a nonlinear relation between quote and trade variable, thus we present a *Nonlinear VAR model*:

$r_t = \sum_{i=1}^{\infty} \alpha_i r_{t-i} + \sum_{j=1}^{n} \sum_{i=0}^{\infty} b_{ij} \chi^j_{t-i} + \epsilon_{1,t}$

$\chi^k_t = \sum_{i=1}^{\infty} \delta^k_i r_{t-i} + \sum_{j=1}^{n} \sum_{i=1}^{\infty} c^k_{ji} \chi^j_{t-i} + \epsilon^k_t$ for $k = 1, ..n$,

where χ^j_t is $+1(-1)$ if a trade is a buyer (seller) initiated and its trade size falls in quantile j, and 0 otherwise, making more flexible assumption that trades in the same trade sizes quantile have equal price impact. Then, we provided another VAR model which takes into account also the Depth Imbalance:

$$\begin{aligned} r_t &= \sum_{i=1}^{\infty} \alpha_i r_{t-i} + \sum_{i=1}^{\infty} \gamma_i di_{t-i} + \sum_{i=0}^{\infty} \beta_i \chi_{t-i} + \epsilon_{1,t}, \\ \chi_t &= \sum_{i=1}^{\infty} \delta_i r_{t-i} + \sum_{i=1}^{\infty} \eta_i di_{t-i} + \sum_{i=1}^{\infty} \theta_i \chi_{t-i} + \epsilon_{2,t}, \\ di_t &= \sum_{i=1}^{\infty} \kappa_i r_{t-i} + \sum_{i=1}^{\infty} \mu_i di_{t-i} + \sum_{i=0}^{\infty} \nu_i \chi_{t-i} + \epsilon_{3,t}. \end{aligned}$$

where χ_t is +1(-1) if a trade is a buy (sell), and di_t represent the DI.

[1]The calculation of this IRF is presented in **Appendix A**

3.2 Recursive approach

We start by presenting the RL framework, in which an agent interacts with its environment in discrete time steps. At each step, the agent observes the current state of the environment, s. Based on this observation, the agent selects an action, a, from a predefined set of possible actions A, which is then executed within the environment. As a result, the environment transitions to a new state, s', and the agent receives an immediate reward, R , associated with the transition. The agent's goal is to maximize the cumulative, long-term discounted sum of rewards by making decisions that optimize this outcome. In this section, we present the RL framework and introduce an algorithm, known as Q-learning, that enables the agent to learn the optimal action to take in each state. Thus, we define the expected immediate reward as $R(s, a)$ and the probability of making a transition from s to s' given a as $T(s, a, s')$. For RL, the agent's goal is to find the optimal policy, i.e., the sequence of actions that maximizes the cumulative discounted rewards over time. A policy, denoted by π , is used to compute the optimal value of a state. The optimal state value V*(s) is typically defined as:

$$V^*(s) = \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right] \quad 0 < \gamma < 1$$

which is typically expressed as a system of S simultaneous equations:

$$V^*(s) = \max_{a} \left(R(s, a) + \gamma \sum_{s' \in S} T(s, a, s')(V^*(s')\right) \quad \forall s \in S$$

for every state s, there are A possible actions, leading to S x A possible state-action pairs, denoted as $\langle s, a \rangle$. Each state-action pair is associated with a value $Q^*(s, a)$, which is the expected infinite discounted reward the agent accumulates if it performs action a in state s and then follows the optimal policy. This value can be decomposed into two parts: the immediate reward for taking action a, denoted as $R(s, a)$, and the cumulative future rewards obtained by executing the optimal actions in subsequent states:

$$Q^*(s, a) = R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') \max_{a'}(Q^*(s', a')).$$

The Q-learning update rule is based on value iteration and can be expressed as:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a) + \alpha(R(s_t, a_t) + \gamma \max_{a}(Q_t(s_{t+1}, a)),$$

where α is the learning rate. *Watkins and Dayan (1992)* demonstrated that Q-learning converges to the optimal Q-values Q^* with probability 1, provided that all actions are sampled sufficiently often in every state and the Q-values are represented in a discrete form.
In order to adapt the RL framework, we consider the agent to be the trader. Then,

$R(s,a)$ can be interpreted as the immediate price impact due to an action a, as buy or sell, in a state s, which can be either a positive or negative depth imbalance (DI). Thankfully to its specifications, this model can be extended to a more realistic scenario by allowing to a different market conditions and available actions. In account of that, we specify the market reaction $\overline{a}$, so that we can take into account the probability of transitioning from the state s to a tuple $< s', \overline{a} >$, via action a, $T(s, a, < s', \overline{a} >)$, which can be ordered and then represented by a transition probability of a Markov chain. Since the aim of the model is to model the permanent price impact of a single trade, regardless of the trader's objective function, we do not need to $\max_a$, so the long term permanent shift in the midpoint triggered by one initial trade is expressed by:

$$\underbrace{Q^*(s,a)}_{\text{permanent price impact}} = \underbrace{R(s,a)}_{\text{immediate impact}} + \gamma \sum_{s' \in S} \sum_{a' \in A} \underbrace{T(s,a,\langle s',a' \rangle)}_{\substack{\text{probability} \\ \text{of observing} \\ \text{future action } \bar{a}}} \underbrace{Q^*(s',a')}_{\substack{\text{permanent price} \\ \text{impact caused} \\ \text{by action } \bar{a}}} . \quad (1)$$

Then, to derive an iterative update rule that converges to point estimates of the permanent price impact for each experience tuple, the Q-learning rule has been slightly modified as:

$$Q_{t+1}(s_t, a_t) = R(s_t, a) + \gamma \sum_{s' \in S} T(s_t, a_t, \langle s', \overline{a} \rangle) Q_t(s', \overline{a}). \quad (2)$$

If all actions are repeatedly sampled across all states and the action values are represented discretely, the Q values will eventually converge to Q^* with probability 1 as we iterate *(2)* (*see Watkins and Dayan (1992)*). Thus we defined a set of action, A - sell and buy - and a set of market states, S. Given this parameters it is possible to estimate the immediate reward and the transition probability function. The former has been estimated as the average demeaned change in the log midpoint price for the corresponding observations. The latter measures the probability of transitioning to future experience tuples from current experience tuples, as a transition probability matrix of a Markov chain $(T(i,j))$ for which: $\sum_{j=1}^{n(S)n(A)} T(i,j) = 1$ which leaves $n(S)n(A)(n(S)n(A)-1)$ parameters to estimate. By assuming our stochastic process to be a stationary Markov chain and we define $N_{i,j}$ as the number of times i is followed by j, the MLE estimate of $(T(i,j))$ is:

$$T(i,j) = \frac{N_{i,j}}{\sum_{j=1}^{n(S)n(A)} N_{i,j}} \quad (3)$$

For estimating the permanent price impact, we initiate the permanent price impact estimates for each action market state combination $Q(s,a)$ to 0 and iteratively we update the permanent price impact of *(2)* until the estimates converge. In this model the discount factor γ is akin to the time value of money, as the immediate price impact of trades further in the future receive more discounting. Because we are measuring the price impact of trades over extremely short time horizons, $\gamma \sim 1$, but for ensuring the convergence $\gamma < 1$ we take $\gamma = 0.95$.

4 Data Description

We received tick frequency data from the LOBSTER dataset [2] , which offers high-quality limit order book data for each trading day. The dataset is structured into: a message file that logs all messages received by the NASDAQ server, each row containing a timestamp (measured in seconds from midnight with nanosecond precision), event type, order ID, size, price, and order direction; and an order book file which contains snapshots of the limit order book, updated after each event recorded in the Message File. It includes the top n price levels for bid (buy) and ask (sell) orders. An example from the dataset is shown in *Figure 6-7*. For our analysis, we used the Microsoft (MSFT) data of March 2020, meaning that a complete message file for MSFT contains millions of intraday events. Since for our analysis we need to take into account only the *Execution of a visible limit order* and *Execution of a hidden limit order*, we extrapolated those data from the message file and from the order book file. Thus, by modifying the initial dataset we proceed to compute the midprice (m), the Logarithmic returns (r) and the Depth Imbalance (DI) as:

$$m_t = \frac{A_{price} + B_{price}}{2} \tag{4}$$

$$r_t = \ln(\frac{m_{t+1}}{m_t}) \tag{5}$$

$$DI = \frac{B_{size} - A_{size}}{B_{size} + A_{size}} \tag{6}$$

Time (sec)	Event Type	Order ID	Size	Price	Direction
⋮	⋮	⋮	⋮	⋮	⋮
34713.685155243	1	206833312	100	118600	-1
34714.133632201	3	206833312	100	118600	-1
⋮	⋮	⋮	⋮	⋮	⋮

Figure 6: Message File example

Ask Price 1	Ask Size 1	Bid Price 1	Bid Size 1	Ask Price 2	Ask Size 2	Bid Price 2	Bid Size 2	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1186600	9484	118500	8800	118700	22700	118400	14930	...
1186600	9384	118500	8800	118700	22700	118400	14930	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Figure 7: Order Book File example

[2]https://lobsterdata.com/

5 Estimation and results

Here we want to focus on how we have estimated the model, their specifications and the results obtained by using the data described as in Section 4.

5.1 Reinforcement Learning model

Here we provide an explanation of our proposal of RL model. We assume two possible LOB's state that are functions of the Depth Imbalance, $S = \{-DI, +DI\}$, where $-DI_t$ and $+DI_t$ indicate, respectively, the negative and the positive imbalance of the LOB at time t. By assuming two possible trade actions in $A = \{+1, -1\}$, where $a_t = +1$ representing the buyer- initiated trades and $a_t = -1$ for seller-initiated trades, we obtain four possible state-action combinations $\{(-DI, -1), (-DI, +1), (+DI, -1), (+DI, +1)\}$. This approach assumes only the open market state. Now it's important to set the Reward function ($R(s_t, a_t)$), which depends on the state (s_t) and the action taken in place (a_t), here there is the comparison between the estimation proposed by "Philip 2019" and our estimation.

	Philip(2019)	**Empirical estimation**
$R(-DI, -1)$	-1	-0.226596
$R(+DI, -1)$	-0.5	-0.098192
$R(-DI, +1)$	0.5	0.169712
$R(+DI, +1)$	1	0.327593

Table 7: Temporary Price Impact

We have built four dataset by filtering the one with event types (4,5). Each of these datasets represents the different states and possible actions, and contain the log return. We calculated the average of these returns and multiplied them by 10000 to obtain the data in basis points.

The following table represents the transition matrix $T(s, a, \langle a', \widetilde{s}\rangle)$ where the (a, s) are the actual LOB's state and $\langle a', \widetilde{s'}\rangle$ is the possible future scenario and the possible associated action. So, $T(+DI, 1, \langle +DI', \widetilde{-1}\rangle)$ is the probability that there is one buy in presence of positive depth imbalance and this is followed by a sell in presence of positive depth imbalance. It can be observed that higher values are concentrated along the diagonal, which suggests a "trend-following effect." One possible explanation for this is that the data was not fully "cleaned," meaning that large volumes of the best bid or ask may have been filled by more than one order.

NextState / State	**(+DI', $\widetilde{-1}$)**	**(+DI', $\widetilde{1}$)**	**(-DI', $\widetilde{-1}$)**	**(-DI', $\widetilde{1}$)**
$(+DI, -1)$	0.672779	0.108538	0.175424	0.043260
$(+DI, 1)$	0.191899	0.522115	0.087820	0.198166
$(-DI, -1)$	0.259568	0.078282	0.557068	0.105082
$(-DI, 1)$	0.082121	0.202082	0.175628	0.540169

Table 8: Transition Matrix

Now we are ready to compute the permanent price impact using (2). The recursive approach suggest us to select a starting point for the iteration, we set all possible value of Q to zero, so $Q_0(+DI,-1), Q_0(+DI,1), Q_0(-DI,-1), Q_0(-DI,1) = 0$. An example of this iteration is:

$$\begin{aligned} Q_1(+DI,1) = R(+DI,1) &+ \gamma(T(+DI,1,\langle +DI',\widetilde{1}\rangle)Q_0(+DI',\widetilde{1})+ \\ &+ (T(+DI,1,\langle +DI',\widetilde{-1}\rangle)Q_0(+DI',\widetilde{-1})+ \\ &+ (T(+DI,1,\langle -DI',\widetilde{1}\rangle)Q_0(-DI',\widetilde{1})+ \\ &+ (T(+DI,1,\langle -DI',\widetilde{-1}\rangle)Q_0(-DI',\widetilde{-1}) = \\ &= 0.327593 \end{aligned}$$

We can see that the first step of the permanent price impact estimation depends only by the temporary price impact $R(+DI,1)$. In the same way we can compute the second step of Q's estimation:

$$\begin{aligned} Q_2(+DI,1) = R(+DI,1) &+ \gamma(T(+DI,1,\langle +DI',\widetilde{1}\rangle)Q_1(+DI',\widetilde{1})+ \\ &+ (T(+DI,1,\langle +DI',\widetilde{-1}\rangle)Q_1(+DI',\widetilde{-1})+ \\ &+ (T(+DI,1,\langle -DI',\widetilde{1}\rangle)Q_1(-DI',\widetilde{1})+ \\ &+ (T(+DI,1,\langle -DI',\widetilde{-1}\rangle)Q_1(-DI',\widetilde{-1}) = \\ &= 0.485765 \end{aligned}$$

The procedure is the same for all elements of Q.

Table 9 reports the progression of our permanent price impact. We conclude that the permanent price impact of a buy when a positive depth imbalance exists is 0.6852

Q	Q_0	Q_1	Q_2	**...**	Q_{99}	$Q^\star$
$Q_t(+DI,-1)$	0	−0.0981	−0.1579	...	−0.2006	−0.2006
$Q_t(+DI,1)$	0	0.3279	0.4857	...	0.6852	0.6852
$Q_t(-DI,-1)$	0	−0.2265	−0.3293	...	−0.3815	−0.3815
$Q_t(-DI,1)$	0	0.1697	0.2742	...	0.4559	0.4559
Change	...	0.8224	0.4249	...	0.0001	0.0000

Table 9: Estimates of the permanent price impact (in basis points) for all experience tuples at the end of each iteration of the learning rule defined by (2). The bottom row, labeled "Change", reports the sum of the total change in permanent price impact estimates after each iteration. the values are in basis points.

5.2 VAR model

We present the results Linear Var with lag order of 5[3].

Variable	Coefficient	Std. Error	t-Stat	Prob
r_t Equation				
const	0.000003	0.000000	18.565	0.000
L1.LN Return	0.036060	0.002352	15.333	0.000
L1.Direction	0.000011	0.000000	56.878	0.000
L2.LN Return	−0.038922	0.002373	−16.404	0.000
L2.Direction	0.000001	0.000000	6.779	0.000
L3.LN Return	0.000835	0.002383	0.351	0.726
L3.Direction	−0.000000	0.000000	−0.201	0.841
L4.LN Return	−0.004228	0.002381	−1.776	0.076
L4.Direction	−0.000000	0.000000	−1.222	0.222
L5.LN Return	−0.001801	0.002381	−0.756	0.449
L5.Direction	−0.000001	0.000000	−6.121	0.000
Direction Equation				
const	−0.075812	0.001895	−40.000	0.000
L1.LN Return	1785.930063	29.401200	60.743	0.000
L1.Direction	0.451602	0.002350	192.176	0.000
L2.LN Return	−1252.644483	29.664081	−42.228	0.000
L2.Direction	0.105581	0.002562	41.214	0.000
L3.LN Return	−75.245819	29.787219	−2.526	0.012
L3.Direction	0.046350	0.002570	18.032	0.000
L4.LN Return	−204.627554	29.768814	−6.874	0.000
L4.Direction	0.039973	0.002555	15.644	0.000
L5.LN Return	−110.986569	29.765888	−3.729	0.000
L5.Direction	0.041816	0.002289	18.270	0.000
Correlation Matrix of Residuals				
LN Return	1.000000	0.263591		
Direction	0.263591	1.000000		

Table 10: Summary of Regression Results, Linear VAR(5)

In the equation for LN Return, the constant is small, indicating a near-zero baseline. The first lag of LN Return (0.036) has a positive effect on the current LN Return, while the first lag of Direction (0.000011) also has a small positive effect. The second lag of LN Return (-0.038) negatively affects the current LN Return, and the second lag of Direction (0.000001) has a small positive effect. Higher lags have minimal or no significant effect.

In the second equation for Direction, the constant is negative. The first lag of LN Return (1785.93) has a large positive effect on Direction, while the first lag of Direction (0.4516) also has a positive impact. The second lag of LN Return (-1252.64) negatively affects Direction, and the second lag of Direction (0.1056) has a smaller positive effect. Higher lags show weaker effects but are still significant for Direction. But we can see how the residuals are positively correlated with $\rho \sim 0.2636$, this could worsen the analysis of the IRF, since a shock in one variable influences also the shocks in another variable. One can try to orthogonalize them, but this is out of the scope of our project.
By plotting the IRF, showed in *Figure 8*, it is possible to see in the upper right-hand side of the plotted-figure that given a Direction's change, we have a positive immediate temporary impact on the Logarithmic Returns, which is present until the 5th lag that disappears as time elapses.

[3]As the author suggested, according to the literature, we perform our model on 5 lags, since a greater number of lags could lead to problems as overfitting

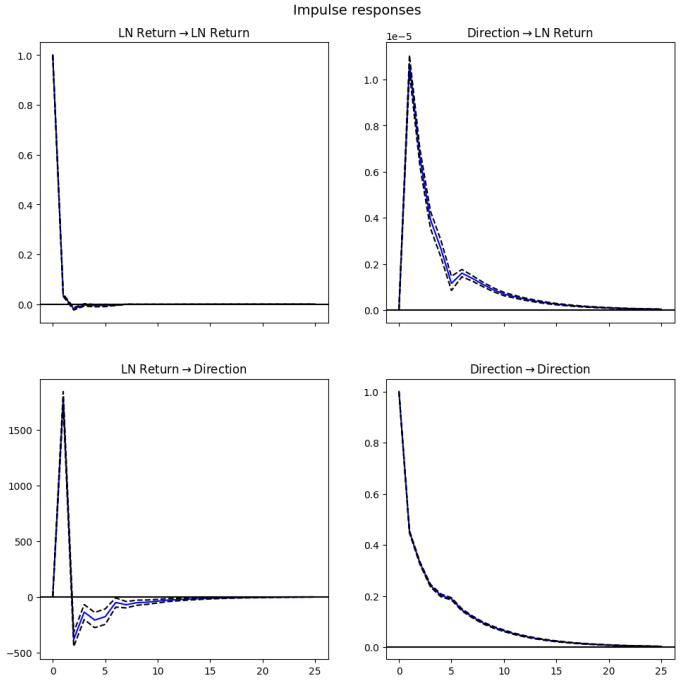

Figure 8: Analysis of IRF.

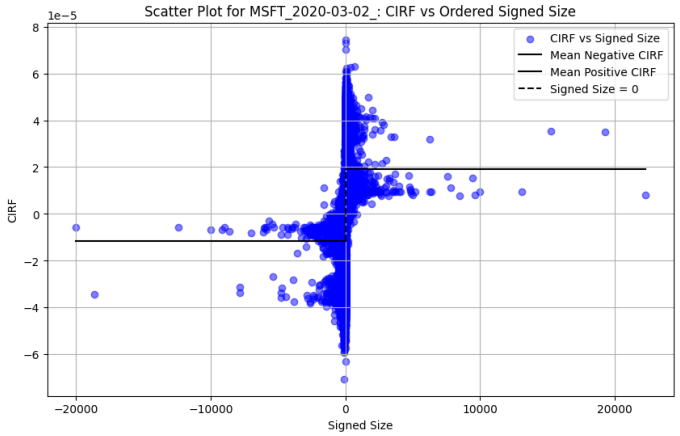

Figure 9: CIRF, on the x-axis the signed size of the order.

The plot above shows the CIRF computed against the signed size of the relative order. The x-axis represents the signed trade size, but it doesn't have an impact on the CIRF values, as one would expect, considering the fact that the linear VAR doesn't take into account the magnitude of the order size. Instead, the factor influencing the CIRF is whether the trade is a buy or a sell.

Let us, now, move the analysis to the other VAR model specified in the paper, that is the VAR with the added feature:

$$\chi_n^k = \begin{cases} +1(-1) & \text{if } n \in \mathcal{B}(n \in \mathcal{S}) \wedge Vol(n) \in Quantile_k, \\ 0 & \text{otherwise .} \end{cases}$$

So, as the paper did, we present the first lag coefficients for each equation:

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
	χ_t^1	χ_t^2	χ_t^3	χ_t^4	χ_t^5	χ_t^6	χ_t^7	χ_t^8	r_t
$c_{1,1}^k$	0.127407	0.065057	0.052615	0.053627	0.099585	0.021561	0.023223	0.041039	0.000010
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
$c_{2,1}^k$	0.060373	0.091956	0.058414	0.130778	0.036707	0.018893	0.022760	0.040540	0.000011
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
$c_{3,1}^k$	0.049691	0.056574	0.076920	0.175963	0.302880	0.022656	0.021533	0.044189	0.000011
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
$c_{4,1}^k$	0.032572	0.092951	0.160269	0.061703	0.242480	0.021197	0.018484	0.043819	0.000011
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
$c_{5,1}^k$	0.073648	0.039104	0.031785	0.038317	0.195865	-0.001189	-0.003783	0.036682	0.000011
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.602**	**0.095**	**0.00**	**0.00**
$c_{6,1}^k$	0.029722	0.031989	0.036335	0.043300	0.005492	0.231489	-0.008724	0.039664	0.000010
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
$c_{7,1}^k$	0.034991	0.036915	0.037481	0.042756	0.005437	-0.006842	0.243860	0.035537	0.000011
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.03**	**0.00**	**0.00**	**0.00**
$c_{8,1}^k$	0.039486	0.046012	0.053340	0.059375	0.035865	0.029939	0.031492	0.195717	0.000012
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
δ_1^k	209.6773888	88.63	91.315651	34.112245	297.647739	401.091792	455.025941	216.499314	0.035956
	0.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**

Table 11: First lag coefficients and p-value for each equations.

The table represent for each column the estimated coefficients for the equation specified in the second row, i.e. column 1 contains the coefficients of the equation:

$$\chi_t^1 = \sum_{i=1}^{5} \delta_i^1 r_{t-i} + \sum_{i=1}^{5} c_{1,i}^1 \chi_{t-1}^1 + \sum_{i=1}^{5} c_{2,i}^1 \chi_{t-1}^2 + ... + \sum_{i=1}^{5} c_{8,i}^1 \chi_{t-1}^8 + \varepsilon_t^1.$$

and since the table refers to the first lag, $i = 1$. Consistently with the work in the paper by **Philip** and [**Hasbrouck(1991a), Engle and Patton(2004), Dufour and Engle(2000)**] nearly every, statistically significant, coefficient are positive showing a degree of autocorrelation, a thing that we already showed in the section "Statistic of Data", moreover as in the paper analyzed, the column χ_t^1 decrease, in a more spurious way, not monotonically, and inversely χ_t^8 increases between $c_1^8 = 0.041039$ and $c_8^8 = 0.195715$, thus meaning that a small trade is more likely to be followed by another small trade, while a large one is more likely to happen before a large trade.

Since the RL model considers the Depth Imbalance (DI), we also estimate a VAR model with three variables, including DI as an explanatory variable. In *Figure 10*, we present the resulting IRF of this model. [4]
Here, we are interested in looking at how a change in the DI, affects the logarithmic returns. In the upper right-hand side of the plot we highlight the immediate positive impact that it has, with a rapid decline, falling below the initial value, which might let us think to a negative effect on next lags.

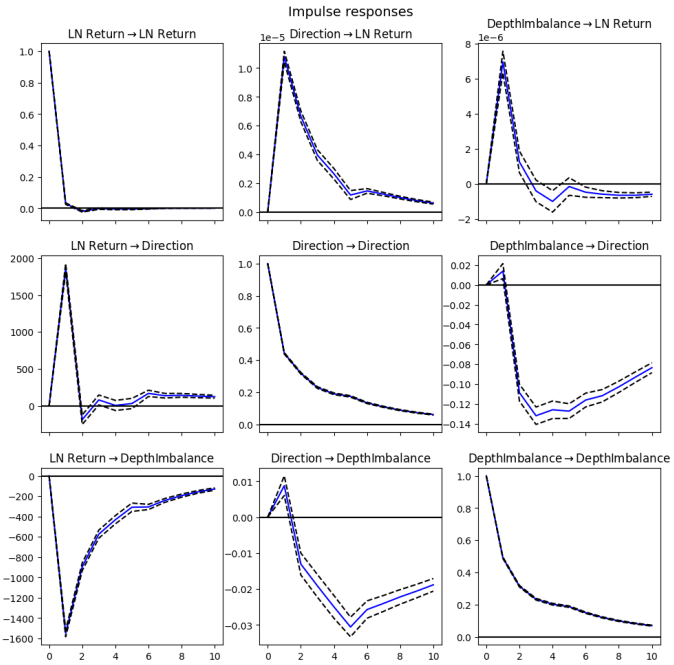

Figure 10: IRF analysis of the 3 equations VAR.

[4]The coefficients table is presented in **Appendix B**.

6. Conclusion

In this study, we successfully replicated the methodology presented by Philip (2019), analyzing the permanent price impact using both a Vector Auto-Regressive (VAR) model and a Reinforcement Learning (RL) approach. Our results confirm that traditional VAR models, despite being widely used in market microstructure analysis, may suffer from misspecification when dealing with nonlinear relationships between trade size and price impact.

The RL-based recursive approach offers a more flexible framework for capturing these nonlinear dynamics, as it can account for state-dependent variations in price impact, particularly through the inclusion of Depth Imbalance (DI) as an explanatory variable. Our empirical results indicate that the RL model produces estimates that align more closely with the expected behavior of market impact, reinforcing the argument that machine learning techniques can enhance traditional econometric approaches in financial modeling.

However, some limitations remain. Our analysis does not consider the influence of different market participants, which could provide a deeper understanding of price impact formation. Furthermore, the RL model is highly dependent on the choice of hyperparameters, such as the discount factor, which may affect convergence properties.

Future research could extend this work by incorporating a broader dataset with multiple assets, testing the robustness of the models under different market conditions, and exploring deep reinforcement learning techniques to further refine price impact estimation. Integrating these methodologies into high-frequency trading strategies could provide valuable insights into optimizing execution algorithms and minimizing trading costs.

Appendix

Appendix A

The response of a random vector y_t to a shock of the type $\varepsilon_{j,t}$,over an horizon h, is,:

$$\frac{\partial}{\partial \varepsilon_{j,t}}(\sum_{s=0}^{\infty} \Psi_s \varepsilon_{t+h-s}) = \Psi_h \varepsilon_j$$

And we calculate this practically with a recursion, that gives us the VMA representation, given an order of p lags:

$$\Psi_0 = I, \Psi_s = \sum_{i=1}^{p} \Pi_i \Psi_{s-i}, s = 1, 2, ...$$

Where Π_i are the coefficients matrices estimated via the VAR.
Once we get the Ψ_s we can explicit the VMA representation:

$$y_t = \sum_{s=0}^{p} \Psi_s \varepsilon_{t-s}$$

Example of order 3:

$\Psi_0 = I$
$\Psi_1 = \Pi_1 \Psi_0$
$\Psi_2 = \Pi_1 \Psi_1 + \Pi_2 \Psi_0$
$\Psi_3 = \Pi_1 \Psi_2 + \Pi_2 \Psi_1 + \Pi_3 \Psi_0$

And finally we can compute the CIRF as:

$$CIRF = \varepsilon_{2,t} \cdot \sum_{i=1}^{p} b_i$$

Where b_i are the coefficients of the matrix Ψ in the position 1,1.

Appendix B

Table 12: Summary of Regression Results

Variable	Coefficient	Std. Error	t-Stat	Prob	Equation
const	0.000003	0.000000	19.014	0.000	LN Return
L1.LN Return	0.029435	0.002367	12.437	0.000	LN Return
L1.Direction	0.000011	0.000000	57.141	0.000	LN Return
L1.DepthImbalance	0.000007	0.000000	22.377	0.000	LN Return
L2.LN Return	−0.029983	0.002441	−12.285	0.000	LN Return
L2.Direction	0.000001	0.000000	7.295	0.000	LN Return
L2.DepthImbalance	−0.000003	0.000000	−7.297	0.000	LN Return
L3.LN Return	0.000733	0.002447	0.300	0.764	LN Return
L3.Direction	0.000000	0.000000	0.480	0.631	LN Return
L3.DepthImbalance	−0.000000	0.000000	−0.075	0.940	LN Return
L4.LN Return	−0.004335	0.002446	−1.772	0.076	LN Return
L4.Direction	−0.000000	0.000000	−0.801	0.423	LN Return
L4.DepthImbalance	−0.000000	0.000000	−0.552	0.581	LN Return
L5.LN Return	−0.002751	0.002444	−1.126	0.260	LN Return
L5.Direction	−0.000001	0.000000	−5.256	0.000	LN Return
L5.DepthImbalance	0.000001	0.000000	2.396	0.017	LN Return
const	−0.079098	0.001888	−41.886	0.000	Direction
L1.LN Return	1869.428634	29.467329	63.441	0.000	Direction
L1.Direction	0.442256	0.002353	187.979	0.000	Direction
L1.DepthImbalance	0.013847	0.003872	3.577	0.000	Direction
L2.LN Return	−1049.440770	30.388522	−34.534	0.000	Direction
L2.Direction	0.102338	0.002556	40.035	0.000	Direction
L2.DepthImbalance	−0.134604	0.004306	−31.257	0.000	Direction
L3.LN Return	−157.776844	30.468349	−5.178	0.000	Direction
L3.Direction	0.043450	0.002565	16.943	0.000	Direction
L3.DepthImbalance	−0.018526	0.004327	−4.281	0.000	Direction
L4.LN Return	−245.656350	30.456218	−8.066	0.000	Direction
L4.Direction	0.037104	0.002542	14.595	0.000	Direction
L4.DepthImbalance	−0.005716	0.004318	−1.324	0.186	Direction
L5.LN Return	−127.066768	30.425787	−4.176	0.000	Direction
L5.Direction	0.036793	0.002281	16.132	0.000	Direction
L5.DepthImbalance	−0.013258	0.003874	−3.422	0.001	Direction
const	0.001667	0.001113	1.498	0.134	DepthImbalance
L1.LN Return	−1550.151281	17.366830	−89.259	0.000	DepthImbalance
L1.Direction	0.008753	0.001387	6.313	0.000	DepthImbalance
L1.DepthImbalance	0.491356	0.002282	215.345	0.000	DepthImbalance
L2.LN Return	−104.041733	17.909743	−5.809	0.000	DepthImbalance
L2.Direction	−0.004495	0.001507	−2.983	0.003	DepthImbalance
L2.DepthImbalance	0.083617	0.002538	32.946	0.000	DepthImbalance
L3.LN Return	−21.740374	17.956790	−1.211	0.226	DepthImbalance
L3.Direction	−0.002806	0.001511	−1.856	0.063	DepthImbalance
L3.DepthImbalance	0.041650	0.002550	16.331	0.000	DepthImbalance
L4.LN Return	−11.706974	17.949640	−0.652	0.514	DepthImbalance
L4.Direction	−0.007136	0.001498	−4.763	0.000	DepthImbalance
L4.DepthImbalance	0.041289	0.002545	16.226	0.000	DepthImbalance
L5.LN Return	58.443095	17.931705	3.259	0.001	DepthImbalance
L5.Direction	−0.008308	0.001344	−6.181	0.000	DepthImbalance
L5.DepthImbalance	0.033714	0.002283	14.767	0.000	DepthImbalance
Correlation Matrix of Residuals					
LN Return	1.000000	0.266753	0.110009		
Direction	0.266753	1.000000	0.041477		
DepthImbalance	0.110009	0.041477	1.000000		

References

- Bouchaud et al.,2004
- Engle and Patton, 2004
- Dufour and Engle, 2000
- Watkins and Dayan, 1992
- Hasbrouck, 1991a