BANK OF ENGLAND

Staff Working Paper No. 896

Inferring trade directions in fast markets

Simon Jurkatis

December 2020

Staff Working Paper No. 896
Inferring trade directions in fast markets

Simon Jurkatis[1]

Abstract

The reliability of established trade classification algorithms that identify the liquidity demander in financial markets transaction data has been questioned due to an increase in the frequency of quote changes. Hence, this paper proposes a new method. While established algorithms rely on an *ad hoc* assignment of trades to quotes, the new algorithm actively searches for the quote that matches a trade. Using an ideal data set that identities the liquidity demander I impose various deficiencies to simulate more typical data sets and find that the new method considerably outperforms the existing ones, particularly at lower timestamp precisions: at data timestamped to seconds the misclassification rate is reduced by half. These improvements also carry over into empirical applications. A risk-averse investor would pay up to 33 basis points per annum to base her portfolio allocation on transaction cost estimates obtained from the new method instead of the popular Lee-Ready algorithm. The recently proposed interpolation method (Holden and Jacobsen, 2014) and the bulk volume classification algorithm (Easley, de Prado and O'Hara, 2012), on the other hand, do not offer improvements.

Key words: Trade classification algorithm, trade initiator, transaction costs, portfolio optimization, limit order book, market microstructure.

JEL classification: C8, G14, G17, G19.

(1) Bank of England. Email: simon.jurkatis@bankofengland.co.uk

I would like to thank Evangelos Benos, Bidisha Chakrabarty, Emanuel Gasteiger, Dieter Nautz, Nicholas Vause, Gavin Wallis, Georg Weizsäcker and the Seminar/Conference participants at the Bank of England, Freie Universität Berlin, Humboldt University, the CFE-CMStatistic Conference, the CEF Conference, the FMND Workshop and the YFS Conference for their valuable comments.

1 Introduction

The trade direction of the liquidity demanding side of the order flow is a necessary ingredient for many traditional measures of market liquidity (Huang and Stoll, 1996; Fong et al., 2017) and remains a popular indicator of informed trading (see, e.g., Bernile et al., 2016; Chordia et al., 2017; Hu, 2014, 2017; Muravyev, 2016). Typically, studies on such topics rely on trade classification algorithms, most prominently the Lee and Ready (1991) algorithm, to obtain an indicator of the liquidity demanding side of each transaction, the trade initiator.

Recently, Easley et al. (2016) and O'Hara (2015) note that the new reality of fast markets poses certain challenges to the reliability of established trade classification algorithms.[1] Yet, few attempts have been made to adjust or design new algorithms.

In this paper, I propose a new algorithm and show that it outperforms the common alternatives, including under the challenging conditions of fast markets.

The established methods, most notably the algorithms of Lee and Ready (1991) (LR), Ellis et al. (2000) (EMO) and Chakrabarty et al. (2007) (CLNV), classify trades based on the proximity of the transaction price to the quotes in effect at the time of the trade. This is problematic due to the increased frequency of order submission and cancellation. With several quote changes taking place at the time of the trade, it is not clear which quotes to select for the decision rule of the algorithm. For example, Angel et al. (2015) record an average of almost 700 quote changes per minute for all stocks in the MTAQ data in 2012.[2] In a sample of NASDAQ trade and quote data studied in this paper, I find a median of 17 quote changes per second in which at least one trade occurs.

The problem of imprecise timestamps relative to the frequency of quote changes does not only pertain to U.S. data or the TAQ equity data. Futures transaction data is often studied with relatively low timestamp granularity (see, e.g., Bernile et al., 2016), and European transaction data collected under the Markets in Financial Instrument Directive (MiFID) are timestamped only to seconds.[3]

In two recent studies Chakrabarty et al. (2015) and Panayides et al. (2019) find accuracies of the LR algorithm of around 85% and 79%, respectively. Chakrabarty

[1]The term "fast market" is used here to describe markets that are subject to high-frequency trading and, as a result of that, experience frequent quote submissions, executions and cancellations.

[2]The MTAQ dataset is the consolidated tape of U.S. stock exchanges and probably the most popular intra-day dataset for research in equities.

[3]Transaction data reported under the MiFID I and its successor MiFID II regulatory framework are a frequent source for European regulators when studying financial markets topic (see, e.g., Kremer and Nautz, 2013; Bicu-Lieb et al., 2020).

et al. (2015) use a combination of NASDAQ ITCH and DTAQ data over a three month period in 2011. Panayides et al. (2019) use data from Euronext timestamped to seconds in 2007-2008. For data from the LSE in 2017 that is timestamped to the microsecond the authors find an even worse performance of only 46% accuracy.

Older studies analyzing the accuracy of the LR algorithm, as well as the alternative EMO and CLNV algorithms, find classification accuracies ranging from 75% to 93% (see, e.g., Theissen, 2001; Finucane, 2000; Chakrabarty et al., 2007; Odders-White, 2000; Ellis et al., 2000; Lee and Radhakrishna, 2000). The concern in many of these studies has been more that of a time-delay between reported quotes and trades rather than insufficient timestamp granularity. The effect, however, is the same: the true trade-quote correspondence is unknown. The traditional response has been to lag quote times by varying degree depending on the sample under study.[4]

The algorithm proposed in this paper takes a new approach to the issue of unknown trade-quote correspondence. Instead of selecting a single pair of ask and bid quotes *before* the classification step, it matches the transaction to its corresponding quote at the same time as it is classified. The idea is that a trade executed against the ask must leave its footprint on the ask-side, while a trade against the bid must leave its footprint on the bid-side. Finding these footprints is equivalent to *simultaneously* finding the quote corresponding to a trade *and* classifying it.

Recent proposals to counter the problem of modern fast markets for trade classification include Easley et al.'s (2012) Bulk Volume Classification (BVC) algorithm and Holden and Jacobsen's (2014) interpolation method. The latter interpolates trade and quote times of imprecisely timestamped data before applying one of the traditional algorithms. The BVC algorithm is a more radical change to trade classification and questions the use of the aggressor flag in the context of extracting informed trading from the order flow altogether. Chakrabarty et al. (2015) and Panayides et al. (2019) show that the LR algorithm outperforms the BVC algorithm with respect to identifying the trade initiator (though the results by Panayides et al. (2019) depend

[4]Recommendations to lag quotes by a certain amount are either based on indirect evidence (see, e.g., Lee and Ready, 1991; Bessembinder, 2003; Henker and Wang, 2006; Piwowar and Wei, 2006; Chakrabarty et al., 2012) or parametric estimations of optimal delay times (Vergote, 2005; Rosenthal, 2012). Chakrabarty et al. (2012) recommend to lag quotes by 1 second, Henker and Wang (2006) recommend to use the last quote from the second before the trade, Piwowar and Wei (2006) and Vergote (2005) find optimal delay times for quotes between 1 and 2 seconds, Peterson and Sirri (2003) and Bessembinder (2003) recommend a 0 lag for quotes, considering 5 seconds intervals ranging from 0 to 30 seconds. For a survey of the use of different lag-rules in all published papers in the Journal of Finance, Journal of Financial Economics and the Review of Financial Studies between 2006 and 2011 see Holden and Jacobsen (2014).

on specific modelling and sample choices).

To evaluate the new algorithm against the LR, EMO, CLNV and BVC algorithms I use data from NASDAQ's electronic limit order book. The sample runs from May to July 2011 with a total of over 134 million transactions timestamped to nanoseconds. The data contain the trade direction of the executed standing orders in the limit order book. Hence, the liquidity supplying and demanding side for each transaction is known, which allows me to evaluate the ability of the algorithms to recover this information.

The NASDAQ data, of course, do not contain the same number of trades and quote changes as, for example, the consolidated tape and possibly other high-frequency databases.[5] This is, however, not a problem *per se* as we are interested in the effect of high order submission and cancellation rates *relative* to the data timestamp precision. To simulate this problem I truncate the timestamp precision of the NASDAQ data at various frequencies.

I find that the new algorithm outperforms the competing classification algorithms. At every considered timestamp precision the new algorithm does not perform worse than the others and it offers considerable improvement in classification accuracy at lower timestamp precisions. For the data timestamped to the second the new algorithm correctly classifies the trade initiator for 95% of the trading volume, compared to 90% for the best competitor, the EMO algorithm. Importantly, I find that the interpolation of timestamps considerably decreases the classification accuracy for the LR, EMO and CLNV algorithm compared to the traditional approach of working with the last quote before the time of the trade based on the original timestamp. Also the BVC algorithm generally performs worse than the traditional approaches.

To give the improvement in classification accuracy more economic meaning, I apply the trade classification methods to the estimation of transaction costs. The transaction costs in turn are used in a portfolio optimization exercise. The results show that an investor with a mean-variance utility function would be willing to forgo up to 33 basis points on yearly returns to use the proposed algorithm to estimate transaction costs instead of the LR algorithm.[6]

The improved accuracy of the proposed algorithm derives from using order book dynamics to identify trade-quote correspondences. The extent to which it can do

[5]For example, Angel et al. (2011) report that NASDAQ's market share in NASDAQ-listed stocks decreased from 53% in April 2005 to around 30% in April 2009.

[6]An online Appendix further demonstrates improvements in the estimation of the order imbalance.

so depends on the precise data structure which will vary with different datasets. To demonstrate how improvements can still be achieved under very limited order book information, I repeat the analysis with noise added to transaction and quote timestamps. The setup is chosen to resemble what one might encounter in data from a consolidated tape with different latencies from the various exchanges to the tape. The algorithm continues to outperform existing methods when adapted to this new data structure.

The remainder of the paper is structured as follows. Section 2 introduces the traditional algorithms, as well the proposed Full-Information algorithm.[7] Section 3 presents the NASDAQ TotalView-ITCH data used in the following exercises. The main results on the classification accuracy obtained under the most accurate and granular data structure are presented in Section 4. Section 5 introduces adjustments to the proposed algorithm for a less granular and accurate data structure, as one might encounter when working with data from a consolidated tape, and presents results on the classification accuracy. In Section 6 the various algorithms are utilized to optimize portfolios under transaction costs. Section 7 presents a separate comparison to the BVC algorithm. Finally, Section 8 concludes.

2 Classification Algorithms

2.1 The LR, EMO and CLNV Decision Rules

The LR algorithm compares the transaction price to the mid-point of the ask and bid quote at the time the trade took place. If the transaction price is greater (smaller) than the mid-point the trade is buyer-(seller-)initiated. If the transaction price is equal to the mid-point, the trade initiator is assigned according to the tick-test: if the transaction price is greater (smaller) than the last price that is not equal to the current transaction price, the trade is buyer-(seller-)initiated.

The alternatives to the LR algorithm examined in this paper are the EMO and CLNV algorithms (Ellis et al., 2000; Chakrabarty et al., 2007). The EMO algorithm classifies a trade as buyer-(seller-)initiated if the transaction price is equal to the ask (bid) price. For all trades off the quotes the tick-test is used. The CLNV algorithm assigns the trade initiator to the buying (selling) side if the transaction price is equal to the ask (bid) or up to 30% of the spread below (above) the ask (bid). For all

[7]Implementations in `python` are available here https://github.com/jktis/Trade-Classification-Algorithms.

trades above (below) the ask (bid) or within a 40% range of the spread around the mid-point the tick-test is used. Table 4 in the appendix summarizes the classification algorithms in pseudo code.

2.2 Quote-Matching Rules

The LR, EMO and CLNV algorithms require assigning one bid and ask quote to each trade in order to classify it. In an ideal data environment where at the time of the trade we record only one quote change, we know that the quotes in effect at the time of the trade are the last ones recorded before the time of the trade. With several quote changes occurring at the same time as the trade, however, the choice is less clear. For example, with one trade and three quote changes recorded at the same millisecond, the quotes corresponding to the trade could be the last quotes from before the millisecond or one of the first two recorded at the millisecond. The convention in such a case is to take the last ask and bid from before the time of the trade.

An alternative suggested by Holden and Jacobsen (2014) to circumvent the problem of imprecise timestamps advises transforming the timestamps to a higher precision. This is done by interpolating the recorded times according to

$$t = s + \frac{2i-1}{2I}, \quad i = 1, \dots, I$$

where t is the interpolated timestamp and s is the originally recorded time. I is the number of trades or the number of changes at the ask or bid at time s depending on which timestamp to interpolate. The algorithm then proceeds as described above using the last ask and bid price from before the time of the trade according to the interpolated time.

2.3 The Full-Information Classification Algorithm

The algorithm proposed here considers all ask and bid quotes that could have been in place at the time of the trade. It then assigns a trade to a quote by using as much information as possible from the data. To understand how we can use the information contained in the data (consisting of transaction prices, volume, best ask, best bid and the volumes available at the quotes) to determine the trade-quote correspondence we need to make some assumptions about the data structure.

Data Structure 1.

(i) Each transaction against a visible order leads to a corresponding reduction in volume available at the respective quote.

(ii) Trades and quotes are reported in the correct order.

Both assumptions hold for the present data set, but they may not for others. Regarding assumption (i), consider, for example, a market order that is too large to be filled by a single limit order. The assumption then states that the order book displays the successive steps in the completion of the market order. That is, if a market buy order for 100 shares trades against two limit orders for 50 shares each, the order book will first show a reduction of 50 shares at the bid and then another reduction of the same size. Since these changes happen basically instantaneously, the order book data may instead display only a singe immediate drop of 100 shares. Also, assumption (ii) may not seem justified when working with data from a consolidated tape due to the different latencies of the exchanges to the tape. Hence, I will relax both assumptions later and discuss adjustments to the proposed algorithm.

Before describing the procedure of the algorithm let me introduce a bit of notation for the transaction and ask data. The notation for the bid data follows analogously to that of the ask.

Notation

- Transaction index: $i \in \{1, \ldots, I\}$
- Transaction price and volume: p_i and v_i
- Recorded transaction time: s_i
- Ask quote index: $j \in \mathcal{J}_a = \{1, \ldots, J_a\}$
- Ask price and volume: a_j and v_j^a
- Change in volume at the j-th ask to the next:

$$\Delta v_j^a = \begin{cases} v_j^a - v_{j+1}^a & \text{if } a_j = a_{j+1} \\ v_j^a & \text{if } a_j < a_{j+1} \\ -1 & \text{otherwise,} \end{cases}$$

Explanation: If the ask price increases from j to $j+1$ ($a_j < a_{j+1}$), then all of the volume at that the j-th ask must have disappeared (either because of a trade or because the order was cancelled), and hence $\Delta v_j^a = v_j^a$.[8] If the ask price decreases from j to $j+1$ ($a_j > a_{j+1}$), then a new sell order must have been submitted with a better limit price than that of the j-th quote. So a trade cannot have taken place at the j-th quote. This is indicated by -1.

- Recorded time of an ask: s_j^a. This indicates the time from which point on the j-th ask price and volume determine the best visible ask.
- The collection of ask quote indices with the same timestamp s: $\mathcal{N}_s^a = \{j \in \mathcal{J}_a \colon s_j^a = s\}$.
- Interpolated time of an ask: $t_j^a = s_j^a + n_j^a/(|N_s^a| + 1)$ with $n_j^a \in \{1, \dots, |N_s^a|\}$
- Auxiliary variable: l^a. This will be used to approximate the ask quote at the time of an execution against a hidden order
- Trade direction of the liquidity demanding party: o_i (1 for buy, -1 for sell)

The algorithm works as follows (see Figure 1 for a graphic representation):

Step 1 – Quote Selection and Matching: Starting with the first trade $i = 1$, we collect all ask and bid quotes against which the trade could have been executed only by considering the timing, i.e. for the ask

$$\mathcal{C}_a = \max\{j \in \mathcal{J}_a \colon s_j^a < s_i\} \cup (\mathcal{N}_{s_i}^a \setminus \max \mathcal{N}_{s_i}^a).$$

These are the last quotes from before the time of the trade and all but the last quote at the time of the trade. We initialize the variable $l^a = a_k$ with $k = \min \mathcal{C}_a$. Analogously, we obtain $\mathcal{C}_b$ and l^b for the bid.

Using transaction price and volume, search for the first match among the selected ask and bid quotes:

$$\boldsymbol{\alpha} = \min\{j \in \mathcal{C}_a \colon p_i = a_j \text{ and } v_i = \Delta v_j^a\}.$$

Analogously we obtain the first match among the bid quotes denoted $\boldsymbol{\beta}$.

Step 2 – Unique Match: If we find a match among the ask quotes, but not for the bid quotes the trade has been executed on the ask and we set $o_i = 1$. Go back to Step 1 and proceed with trade $i + 1$. If $s_{i+1} = s_i$ we use the same collection of ask and bid quotes and set $l^a = a_\alpha$, otherwise we update $\mathcal{C}_a$ and $\mathcal{C}_b$

[8]For the bid quote, the second case reads "if $b_j > b_{j+1}$".

and newly initialize l^a and l^b. (If we find that there is no match among the ask quotes, but at least one among the bids, all updates are made analogously.)

Step 3 – Earliest Match: If we find a match among both the ask and bid quotes, we classify the trade according to which quote seems to be affected first. That is, if $t^a_{\boldsymbol{\alpha}} < t^b_{\boldsymbol{\beta}}$, we set $o_i = 1$. Go back to Step 1 and proceed with trade $i + 1$, making the same adjustments as described under Step 2. Additionally we omit the $\boldsymbol{\alpha}$*th*-ask from further comparisons by subtracting from $\Delta v^a_{\boldsymbol{\alpha}}$ the size of the transaction v_i. If $t^a_{\boldsymbol{\alpha}} > t^b_{\boldsymbol{\beta}}$, we set $o_i = -1$ and updates are made accordingly.

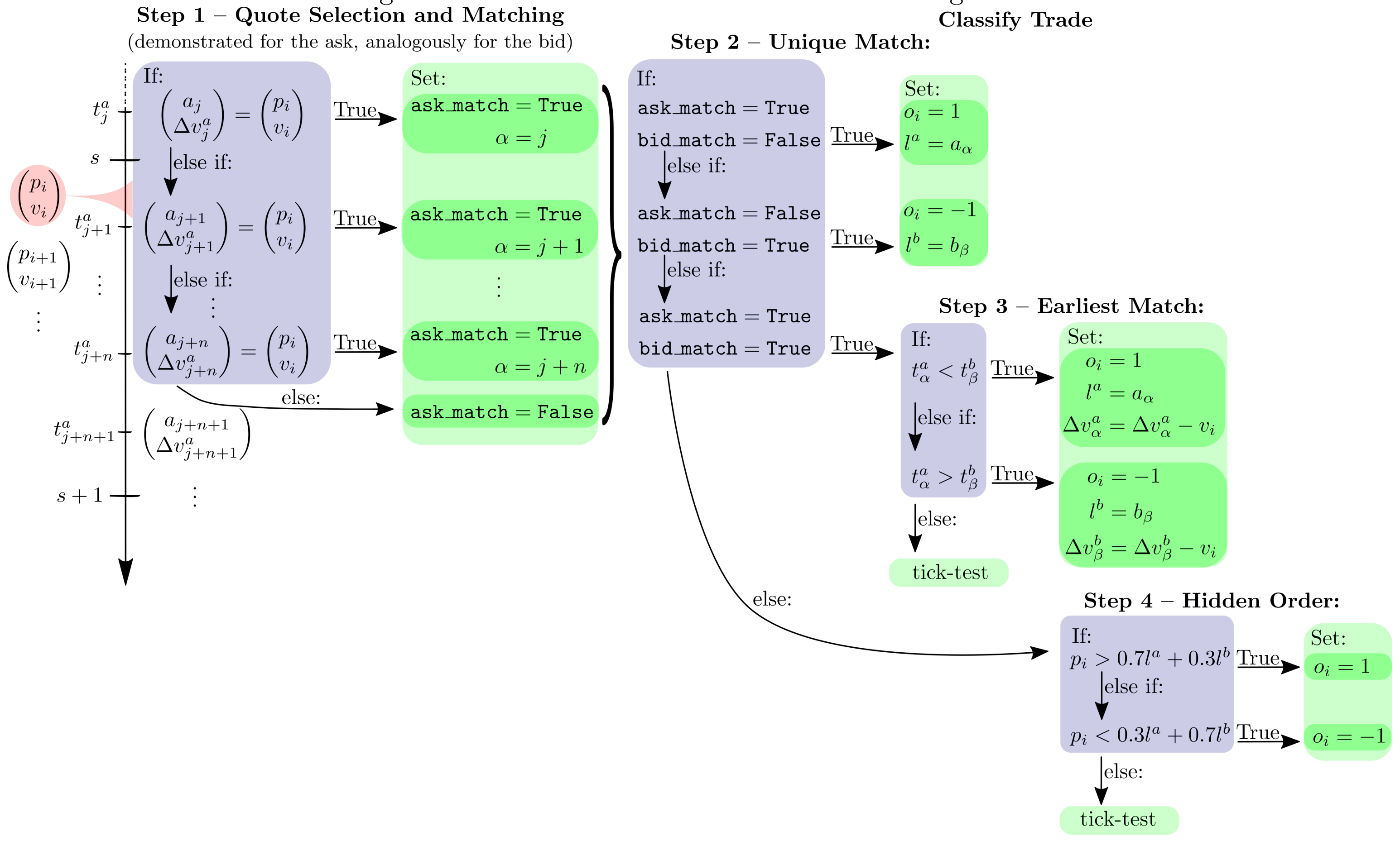
Step 4 – Hidden Order: If we cannot find a match among the ask and bid quotes, we are likely to face a trade against a hidden order. These are classified according to their position in the spread (similar to Chakrabarty et al., 2007), which is approximated by l^a and l^b. If $p_i > 0.7l^a + 0.3l^b$ and $l^a > l^b$ we set $o_i = 1$. If $p_i < 0.3l^a + 0.7l^b$ and $l^a > l^b$ we set $o_i = -1$. Go back to Step 1, proceed with trade $i + 1$ and update $\mathcal{C}_a$ and $\mathcal{C}_b$ if $s_{i+1} > s_i$.

Step 5 – Tick-test: Any trade that could not be classified in Step 2 to Step 4 is classified by the tick-test.

Remark to Step 3 The idea of using the interpolated time to classify trades that match with both an ask and bid quote is as follows. Observing ask and bid quotes to equal each other within the same, say, second of a trade, may be due to the price impact of the trade. That is, quotes are updated in the direction of the trade initiator. This should be reflected in a relatively early interpolated time of the corresponding quote for the following reasons. First, in case of a buyer-initiated trade we may expect more activity on the ask side because of the information contained in the trade that leads to the price impact. Traders will either submit buy orders to take advantage of stale limit orders or cancel their stale limit orders in response to the trade. Either way, $|N^a_s|$ will be relatively large. Second, in case of a buyer-initiated trade, the trade executed *first* on the ask and then bid quotes were updated subsequently upwards. That is, $\boldsymbol{\alpha}$ will be relatively small while $\boldsymbol{\beta}$ relatively large. In total, this means that $t^a_{\boldsymbol{\alpha}}$ will be smaller than $t^b_{\boldsymbol{\beta}}$.

To avoid further conflicts between ask and bid quotes with the same price and volume characteristics, the quote to which the trade is matched is omitted from assignments of subsequent trades. This assumes that a quote can only be hit once, which means that we have eliminated the counter-party to each trade in the data.

Figure 1: The Full-Information Classification Algorithm



Notes: This figure shows the process of the Full-Information algorithm to classify a trade. The variables are defined in the **Notation** list. In **Step 1** we collect all ask and bid quotes against which the trade could have executed considering only the timing of the trade and the quotes. Starting with the first ask and bid quote, respectively, from these collections we search for an exact match of the quote and its volume change with the transaction price and volume. If a match could be found, we set an indicator variable to `True` and memorize the index of the respective quote. In **Step 2**, if only the ask/bid side matches the trade, we classify it as buyer-/seller-initiated and assign the respective quote to the auxiliary variable l^a/l^b, which is used to construct the spread in case of hidden order executions. In **Step 3**, if both sides match the trade, it is classified according to the interpolated time of the matched quotes. The corresponding quote is then omitted from further proceedings by subtracting the transaction volume from the volume change at the quote. In **Step 4** the trade is classified by the position of the price within the spread, which is approximated by the auxiliary variables. Trades not classified in any of these steps are classified by the tick-test.

Alternatively, if the counter-party is not omitted from the data for the classification process, one would drop the corresponding quote after it has been assigned twice to a trade.

Remark to Step 4 The spread in Step 4 is constructed from the auxiliary variables l^a and l^b. They serve to approximate the ask and bid valid at the time of the execution of the hidden order. They are initialized to the first ask and bid valid at the time of the trade. If we are able to classify a trade involving a visible order, the corresponding auxiliary variable is updated. In that way, due to the correct order of trades, we obtain a better approximation of the spread at the time of the hidden order execution.

Remark to Step 5 I follow the design of the traditional algorithms and use the tick-test to classify the most ambiguous cases. This can be motivated by the finding of Perlin et al. (2014) who show that the misclassification rate of the tick-test is upper-bounded by 50%.

3 Data

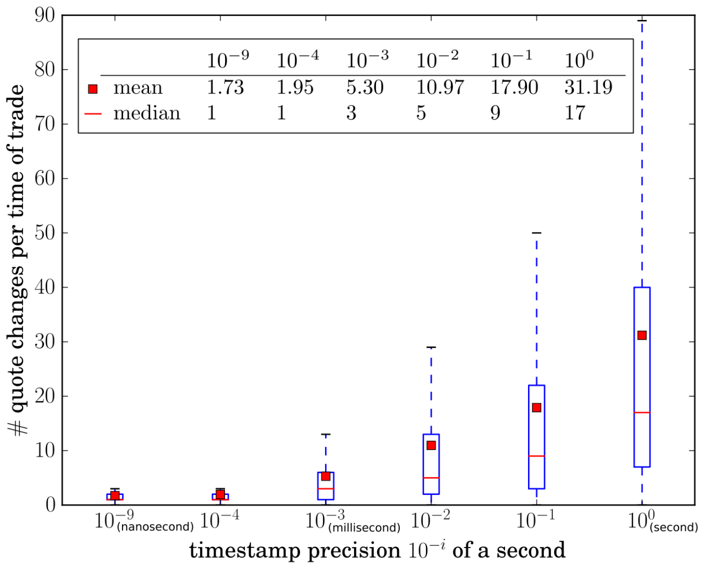
The evaluation of the algorithms is based on equity trading data from NASDAQ's electronic limit order book constructed from NASDAQ's TotalView-ITCH data.[9] The trade data contain all transactions against visible and hidden limit orders with information on the price and volume of the transaction. The order book data contain the development of the order book. That is, whenever a visible limit order that affects the best quotes is submitted, canceled (partially or completely) or executed, the order book contains an entry of the best bid or ask indicating the new price and volume available. Changes regarding hidden orders are not displayed in the order book.

The data covers the continuous trading phase from 9:30 am to 4 pm for all trading days during the 3 month period May to July 2011. I selected the 30 largest stocks (by market capitalization) in 2015 from the 11 NASDAQ industry sectors.[10] Following

[9]The reconstruction from the TotalView-ITCH data is done by the software LOBSTER, which in turn produces the order book data and messages files containing the information on the events causing the changes in the order book. A detailed description of how I obtain the trade and quote data from these files is given in the online appendix.

[10]These sectors are: Basic Industries, Finance, Capital Goods, Healthcare, Consumer Durables, Consumer Non-Durables, Public Utilities, Consumer Services, Technology, Energy, Transportation, Miscellaneous.

Figure 2: Distribution of the Number of Quote Changes at Different Frequencies



	10^{-9}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}
■ mean	1.73	1.95	5.30	10.97	17.90	31.19
— median	1	1	3	5	9	17

Notes: For each time where at least one trade takes place, I count the number of quote changes with the same timestamp. This figure shows the distribution of these counts (outliers omitted) in terms of boxplots for different timestamp precisions. For example, 50% of the milliseconds with at least one trade also display 3 or more quote changes.

Chakrabarty et al. (2015), I drop stock-days with an end-of-day price of less than one dollar or with less than 10 trades, which leaves me with a total of 19842 stock-days. Summary statistics are provided in Table 5 in the appendix.[11]

The proposed algorithm is designed to offer improvements when the number of quote changes at a given timestamp is relatively high, i.e. at high quotation frequency relative to the timestamp precision. Figure 2 plots the distribution of the number of quote changes (outliers omitted) at the times of trades for different timestamp precisions: the original precision of nanoseconds (10^{-9} of a second), as well as 10^{-4} to 10^{0} of a second.

At a precision of 10^{-4} of a second or higher, most of the trade times experience only a small number of quote changes, which would allow us to match trades to their quotes based only on the timing of the two. With decreasing timestamp precision, however, this number increases quickly. At a precision of seconds, the median number of quote changes with the same timestamp as that of a trade is 17. With 17 quote changes at the time of a trade, we cannot deduce, just from the timing of the two, which quote belongs to the trade.

[11]A list with all ticker names studied in this paper is given in the online appendix.

4 Main Results

4.1 Classification Accuracy at Different Timestamp Precisions

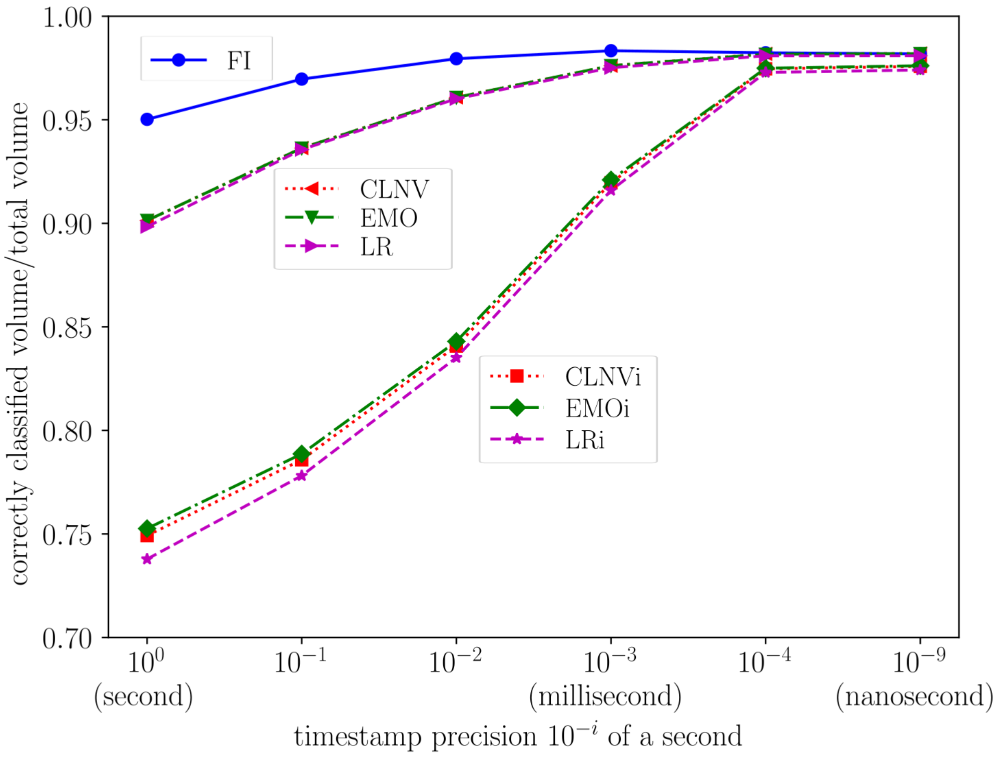
I analyze the improvements we can achieve over the traditional algorithms by applying them and the proposed Full-Information algorithm (FI) to the data with varying timestamp precisions. The traditional algorithms are used in combination with the quote matching rule of using the last quotes from before the time of the trade (denoted by LR, EMO and CLNV), and using the interpolated time of trades and quotes (denoted by LRi, EMOi and CLNVi) as proposed by Holden and Jacobsen (2014). The timestamp precisions are chosen to be of 10^{-i} of a second for $i = 0, 1, 2, 3, 4$, as well as the original data precision of nanoseconds (10^{-9} of a second). I evaluate the quality of the algorithms on the basis of correctly classified trading volume. Figure 3 presents the sum of correctly classified trading volume over total trading volume over the entire sample. Table 6 in the appendix provides the corresponding numbers, along with the means and standard deviations across the sample.

The results show that the FI algorithm dominates the others. At the original timestamp precision of nanoseconds all of the algorithms correctly classify around 98% of trading volume. Approaching the timestamp precision of seconds, however, the performance of the traditional algorithms falls off more quickly than that of the FI algorithm. At the precision of seconds the traditional algorithms correctly classify around 90% of trading volume, in contrast to 95% correctly classified volume by the FI algorithm. That is, the FI algorithm reduces the number of misclassified shares by half.

Table 6 in the appendix shows that the FI algorithm also dominates in terms of the variation in classification accuracy. While the standard deviation of the stock-day classification accuracy barely moves for the FI algorithm (from 2.09%-points at nanosecond to 2.4 at second precision), the standard deviation of the traditional algorithms increases from the same level of around 2.1%-points to more than 3.4.

Another important finding of this exercise is that the performance of the traditional algorithms deteriorates by interpolating the timestamp precision. The method has gained considerable traction over the past years (e.g. Chordia et al., 2017; Brogaard et al., 2018; Brennan et al., 2018), but the actual effect on the identification of the trade initiator has not been tested. The evidence presented in Holden and Jacobsen (2014) is based on a comparison between the MTAQ and DTAQ data, but

Figure 3: Classification Accuracy at Different Timestamp Precisions



Notes: This figure depicts the fraction of correctly classified trading volume by the FI algorithm and the traditional algorithms using the last quotes from before the time of the trade (EMO, CLNV and LR), and using the interpolated time of trades and quotes (EMOi, CLNVi and LRi). The algorithms are applied to the data with reduced timestamp precisions (10^{-i} of a second for $i = 0, \ldots, 4$), and using the original precision of nanoseconds (10^{-9} of a second). These correspond to median number of quote changes at the time of trades ranging from 17 (for $i = 0$) to 1 (for $i = 9$).

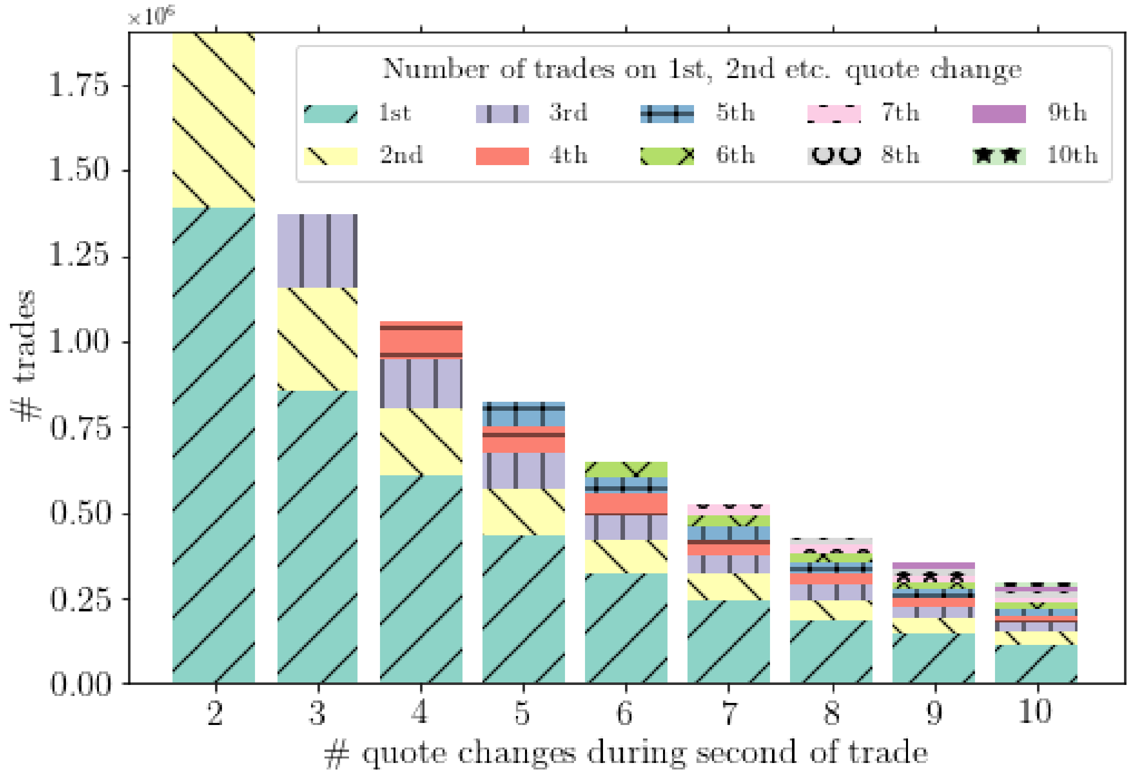
not on the actual knowledge of the trade initiator.

The deteriorating effect of the interpolation method can be explained by the relationship between the timing of quote changes and trades. The idea behind the interpolation method is that trades and quotes are equally distributed over a given time interval, which is true if we look at quotes and trades unconditionally.

In a classification exercise, however, one looks at quote changes conditional on that a trade occurred. A trade may lead to several successive quote changes such that the number of quote changes to the right of the quote change that was triggered by the trade is likely to be greater than the number of quote changes to its left. This implies that a trade is likely to be placed behind the quote change that was triggered by the trade if we interpolate following Holden and Jacobsen (2014).

Figure 4 confirms these considerations. It shows how often a trade triggered the

Figure 4: Relation between Quote Changes and Trades during Single-Trade Seconds



Notes: The Figure shows the number of times a trade triggers the first, second, third, ..., 10-th quote (y-axis) within seconds of 2, 3, ..., 10 quote changes (x-axis). Only seconds containing a single trade are used.

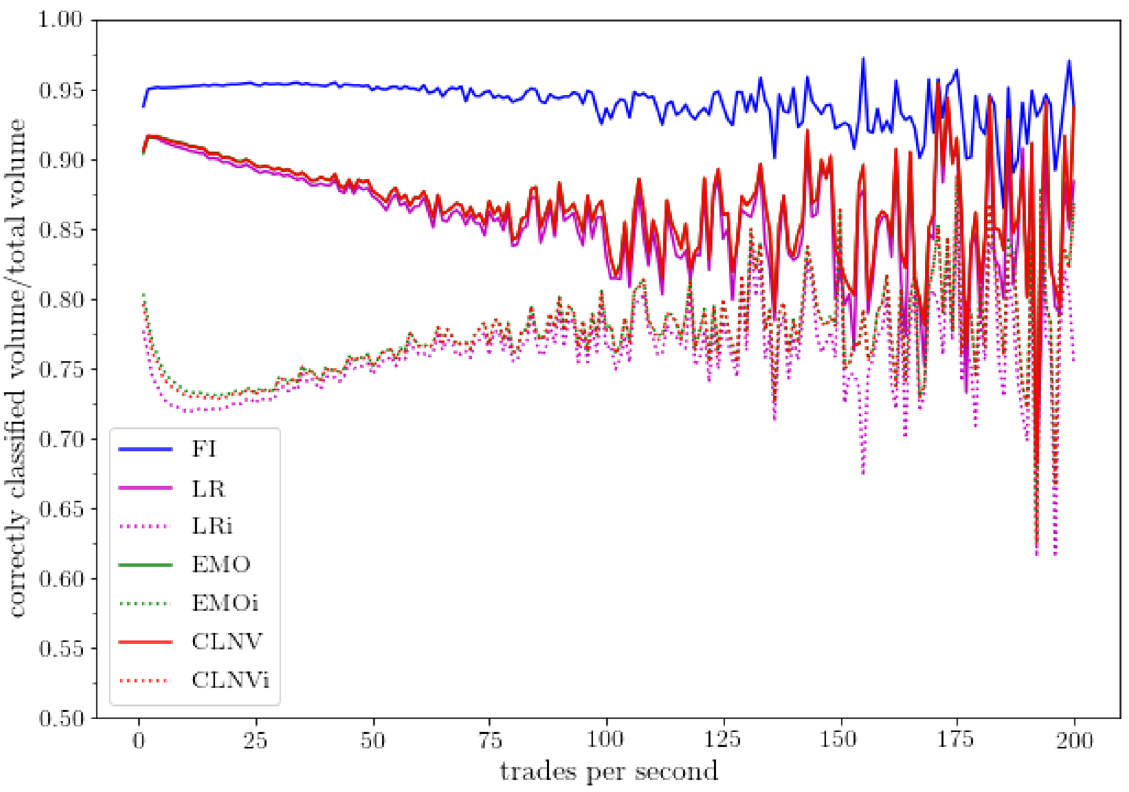
first, second, third etc. quote change (y-axis) for a given number of quote changes during the second of a single trade (x-axis). The first bar shows that across all cases with 2 quote changes during the second of a trade almost 1.4 million trades are related to the first quote change, whereas only around 0.5 million to the second.

4.2 Classification Accuracy and Number of Trades

The improvement that the FI algorithm achieves over the traditional ones is derived from particularly active trading intervals. To show this more clearly Figure 5 plots for each algorithm the fraction of correctly classified trading volume depending on the number of trades (from 1 to 200) during the second of the trade for the data timestamped to seconds.

Both the traditional algorithms under the traditional quote matching rule and the FI algorithm show an initial improvement starting from only one trade per second.

Figure 5: Classification Accuracy Depending on Number of Trades Per Second



Notes: The Figure shows the correctly classified volume over total volume (y-axis) across all seconds with a given number trades (x-axis). All algorithms have been applied to the data timestamp to the second.

Single trade seconds often involve hidden orders which are more difficult to classify for both types of algorithms. Only the FI algorithm, however, maintains its high classification accuracy of around 95% even at very active trading seconds, while that of traditional algorithms gradually drop to 85% and lower. So the improvements shown earlier are even more pronounced when markets are indeed very fast.

4.3 Classification Accuracy at the Individual Classification Steps

The Full-Information algorithm classifies trades at different steps, depending on the criteria that apply to the specific trade. The uncertainty of the classification increases with each step in the algorithm and we would, thus, expect the accuracy to differ with the different classification criteria.

Table 1: Accuracy of Individual Classification Steps

		timestamp precision: 10^{-i} of a second for $i =$					
		0	1	2	3	4	9
visible	cl. step	(s)			(ms)		(ns)
Panel A: % correctly classified volume							
YES	2	99.85	99.91	99.96	99.99	100.00	100.00
	3	90.31	94.18	95.98	95.11	68.96	–
	4	79.67	80.32	80.70	80.57	80.55	80.57
	5	55.97	55.90	56.63	56.31	31.74	29.80
NO	2	67.25	74.49	84.52	94.50	99.79	99.98
	3	83.44	87.66	87.17	79.64	66.67	–
	4	92.59	94.79	95.75	95.58	94.39	93.86
	5	64.96	65.21	65.94	67.20	68.60	68.81
Panel B: % classified volume							
YES	2	61.79	71.44	81.20	88.55	90.42	90.42
	3	28.55	18.93	9.19	1.86	0.00	0.00
	4	0.00	0.00	0.00	0.00	0.00	0.00
	5	0.07	0.05	0.03	0.01	0.00	0.00
NO	2	1.87	1.84	1.70	1.33	0.80	0.72
	3	0.65	0.30	0.09	0.01	0.00	0.00
	4	4.01	4.23	4.34	4.29	3.87	3.79
	5	3.05	3.20	3.45	3.95	4.92	5.07

Notes: Panel A shows the classification accuracy of the different classification steps applied by the FI algorithm. Panel B shows the percentage of trading volume that is classified under the respective step. The column "cl. step" refers to the step in the classification process at which the trade initiator is assigned.

Table 1 presents the classification accuracies at the individual classification steps. Panel A shows the percentage of correctly classified volume, while Panel B shows the percentage of trading volume that is classified at the respective classification step. The table differentiates between trades executed against visible (visible = YES) and hidden (visible = NO) orders. The column "cl. step" refers to the classification steps (2 to 5) at which the trade initiator is assigned.

Trades against visible orders are almost exclusively classified during Step 2 or 3 of the classification process. That is, any trade that executed against a visible order must have at least one match among the quotes with the corresponding change in volume. Matches between quotes and trades that actually executed against hidden orders, on the other hand, are only accidental and occur rarely. With decreasing timestamp precision the number of hidden orders classified in Step 2 or 3 increases as the number of quotes that we consider during the classification of a trade increases. Overall, however, the number of hidden orders classified in Step 2 and 3 remains relatively small. Trades involving hidden orders are predominantly classified in Step 4 or 5 of the algorithm, as they are supposed to.

The accuracy of the assignments of visible orders in Step 2 of the algorithm is almost 100% at any timestamp precision. With decreasing timestamp precision, however, the number of unambiguous assignments decreases and the algorithm refers to the interpolated times of the matched quotes more often. Although the decrease in overall classification accuracy going from nanoseconds to seconds is largely driven by the substitution of assignments between Step 2 and 3, the interpolated time is still a suitable indicator for the assignment with accuracies between 90 to 95%.

The classification of trades involving hidden orders is inherently more difficult than for visible orders. At nanosecond precision, the number of misclassifications can almost entirely be attributed to trades involving hidden orders. With decreasing timestamp precision the classification accuracy of trades involving hidden orders, however, does not change much. The informativeness and the number of cases assigned to Step 4 is almost the same whether for data timestamped at nanoseconds or seconds. Most of the change in hidden orders classification accuracy is due to a shift from classifications by the tick-test (Step 5) to Step 2 of the algorithm.

5 Classification Accuracy under Noisy Timestamp

5.1 Data Structure

So far, we assumed the same level of data granularity (summarized in Data Structure 1) that is provided by the reconstructed limit order book from the NASDAQ TotalView-ITCH data. The advantage of the FI algorithm over the traditional approaches feeds on the use of information offered from this granularity. In this section, I will relax the assumptions in Data Structure 1 and present appropriate adjustments to the algorithm.

Specifically I will assume:

Data Structure 2. *Aggregated Quote Changes and Random Trade and Quote Order*

(i) At the time of a trade the order book displays the new state of the order book after the completion of all transactions that were carried out due to the same

buy or sell order.

(ii) Trades and quotes can be out of order.

Assumption (i) means that if, for example, a market buy order for 100 shares is executed against two sitting limit orders for 50 shares each, the order book will display a single drop in the volume at the best ask of 100 shares. Since the transaction data still records two trades of 50 shares each, we can no longer match transaction sizes to an equivalent change in the number of share at the quotes.

Assumption (ii) may be a concern if one deals with data from a consolidated tape which timestamps trades and quotes when they are processed at the tape, not when they were executed on the exchange. Due to different latencies for sending information from different exchanges to the same data processor trades and quotes can be out of order. These latencies can be expected to be small, but large enough to affect trades or quotes that are executed or submitted over small intervals.[12]

5.2 Adjusting the FI Algorithm to DS 2

If quotes are indeed frequently out of order we can no longer reliably construct the changes in volume at the quotes. Therefore, the search for a match between a transaction and a quote in Step 2 is changed to (demonstrated for the ask)

$$\boldsymbol{\alpha} = \min\{j \in \mathcal{J}_a : p_i = a_j \text{ and } v_i \leq v_j^a\},$$

and analogously for the bid. From here, the adjusted algorithm proceeds as the baseline version. However, the auxiliary variables l^a and l^b are not updated after a classification at one of the quotes. If a classification is derived under Step 3, the algorithm subtracts the transaction volume from the volume available at the matched quote.[13]

5.3 Classification Accuracy

To study the effect of random trade and quote order, I add exponential noise to the original nanosecond timestamps of both trade and quote data. Note that in doing

[12]The speed of light in a vacuum is roughly $300 * 10^6$ m/s. Sending data from Chicago to New York (a distance of around 1300km) at the speed of light would thus take 4ms. So even at this physically lower limit of transmission time the report delay of a Chicago trade is 4ms compared to a trade at the NYSE where the consolidated tape is located.

[13]Another version of the FI algorithm for an intermediate data structure with aggregated quote changes but trades and quotes still in order is available online.

so, not only will the order of trades and quotes change, but trades may now also be reported before or after their corresponding quote change.

The exponential distribution is given by $F(x;\beta) = 1 - \exp\{-x/\beta\}$ for $x \geq 0$ and I choose $\beta = 10^{-j}$ for $j = 1, \ldots, 4$.[14] I then choose different timestamp precisions at which the algorithms are applied to the data, with the adjusted FI algorithm labelled FI_2. The precision ranges from 10^{-4} of a second to 2.5 seconds. Note that whereas in the previous section a low timestamp precision had a negative effect on the classification accuracy, we may now deliberately reduce the timestamp precision to counteract the effect of noise.

Contrary to the previous data structure, there are regions of timestamp precisions and noise where the traditional algorithms outperform the FI_2 algorithm. At these regions, however, classification accuracy is quite low for all the algorithms as they are strongly affected by noise.

As the level of noise is most likely unknown to the user, using trade classification algorithms on data that suffer from noisy timestamps poses a trade-off between incurring lower accuracy by applying the algorithm at low timestamp precision or risking lower accuracy by leaving the algorithm susceptible to noise at high timestamp precision. The FI_2 eases this trade-off by offering a more stable performance between different degrees of timestamp precision. At those timestamp precisions that proof robust against noise, the FI_2 algorithm again outperforms the others, albeit less pronounced.

5.4 Classification Accuracy and Number of Trades

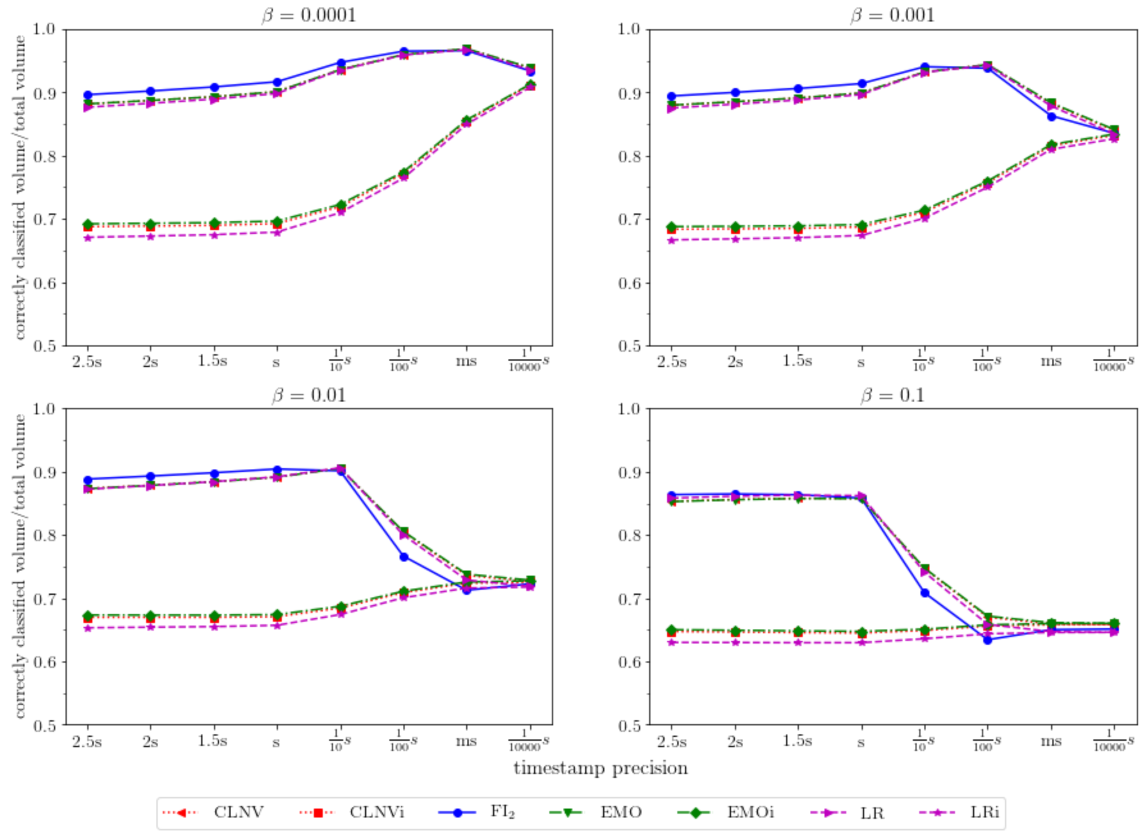
To compare the performance of the algorithm depending on the number of trades to be classified over the given interval I again apply the algorithms to the data timestamped at seconds. Results are presented in Figure 7 for the various degrees

[14]The mean of a exponentially distributed variable is given by β and the q-th percentile by $-\ln(1-q)\beta$. For example, if $\beta = 1/10^3$, the expected delay in the reported trade time is one millisecond and 99% of all trades are expected to have a delay of less than 5 milliseconds. Moreover, for two trades with the second trade following one millisecond after the first one, the probability that the two will be reported in reverse order is 0.1839. More formally, for two trades at time t_1 and t_2 with $t_2 = t_1 + \Delta$ and $\Delta \geq 0$ the probability that the first trade is shifted behind the second one due to noise is given by

$$P(t_1 + \varepsilon_1 > t_2 + \varepsilon_2) = \int_{\Delta}^{\infty} f(\varepsilon_1) F(\varepsilon_1 - \Delta) \, \mathrm{d}\varepsilon_1,$$

with $\varepsilon_i \in \mathbb{R}_{\geq 0}$ and $\varepsilon_i \overset{\text{iid}}{\sim} F$ for some distribution function F with density f. For F being the exponential distribution $\mathrm{Exp}(1/\beta)$ this is given by $\exp\{-\Delta/\beta\}/2$.

Figure 6: Classification Accuracy under Random Trade and Quote Times of Data Structure 2

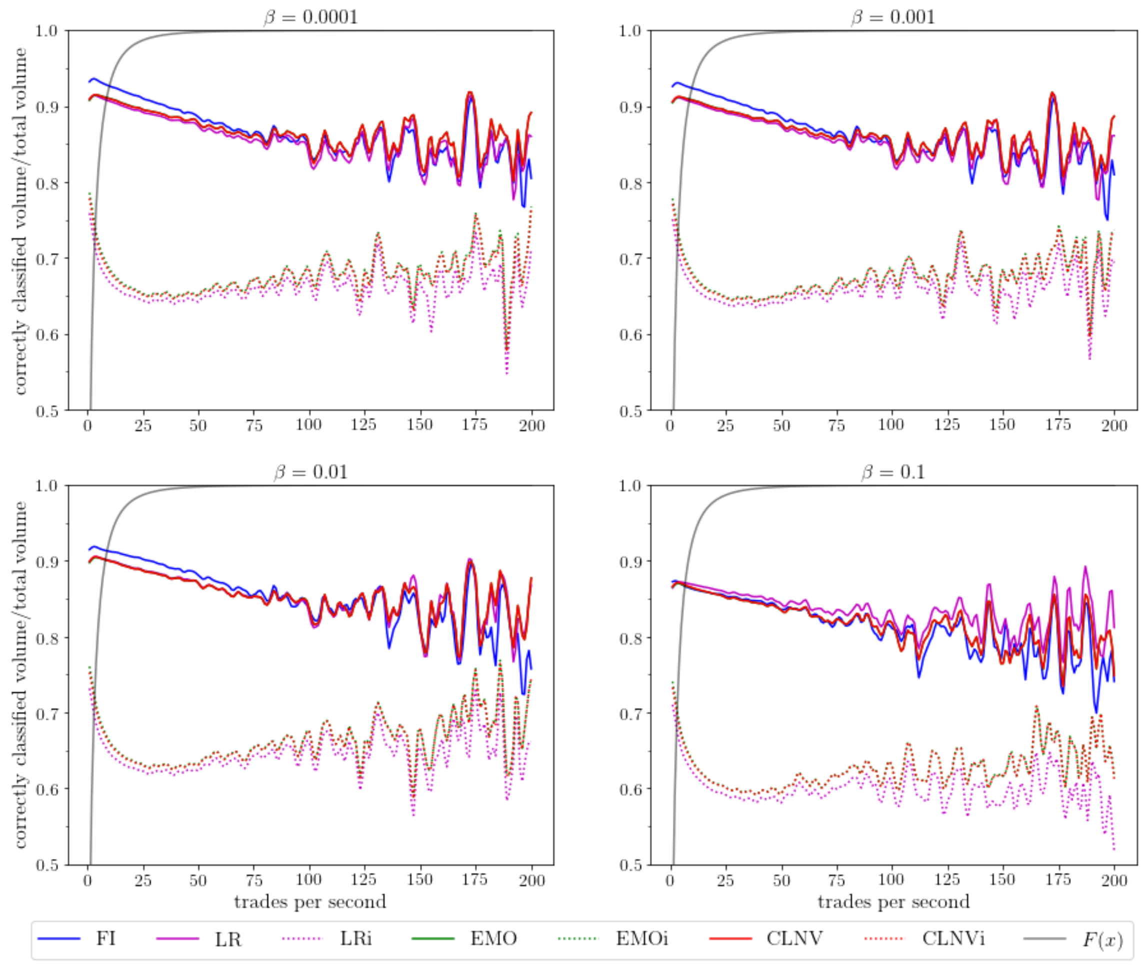


Notes: This figure shows the fraction of correctly classified trading volume (y-axis) for data with noisy quote and trade times (Data Structure 2). The recorded time of trades and quotes equals the actual time plus ε, with $\varepsilon \sim \text{Exp}(1/\beta)$ and $\beta \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. The classification algorithms are apply to the data with reduced timestamp precision ranging from 10^{-4} of a second to 2.5 seconds (x-axis).

of noise. Each data series has been smoothed with a standard Gaussian kernel to obtain a better view on the more volatile region of more than 100 trades per second of which there are relatively few, as can be seen from the grey line which plots the distribution of the number of trades per second.

At very strong noise ($\beta = 0.1$) the performance of the FI$_2$ algorithm and the traditional ones is very similar across the range of number of trades per timestamp, with a slight advantage for the LR algorithm over extremely active trading seconds. At such strong noise, however, it would be advisable to reduce the precision further to intervals of two seconds, at which the FI$_2$ would take a slight advantage. Reducing the noise to more moderate levels the FI$_2$ algorithm offers a more considerable improvement over a range of about up to 75 trades per second. As can be seen from the plot of the distribution of the number of trades per second the vast amount of

Figure 7: Classification Accuracy Depending on Number of Trades Per Second

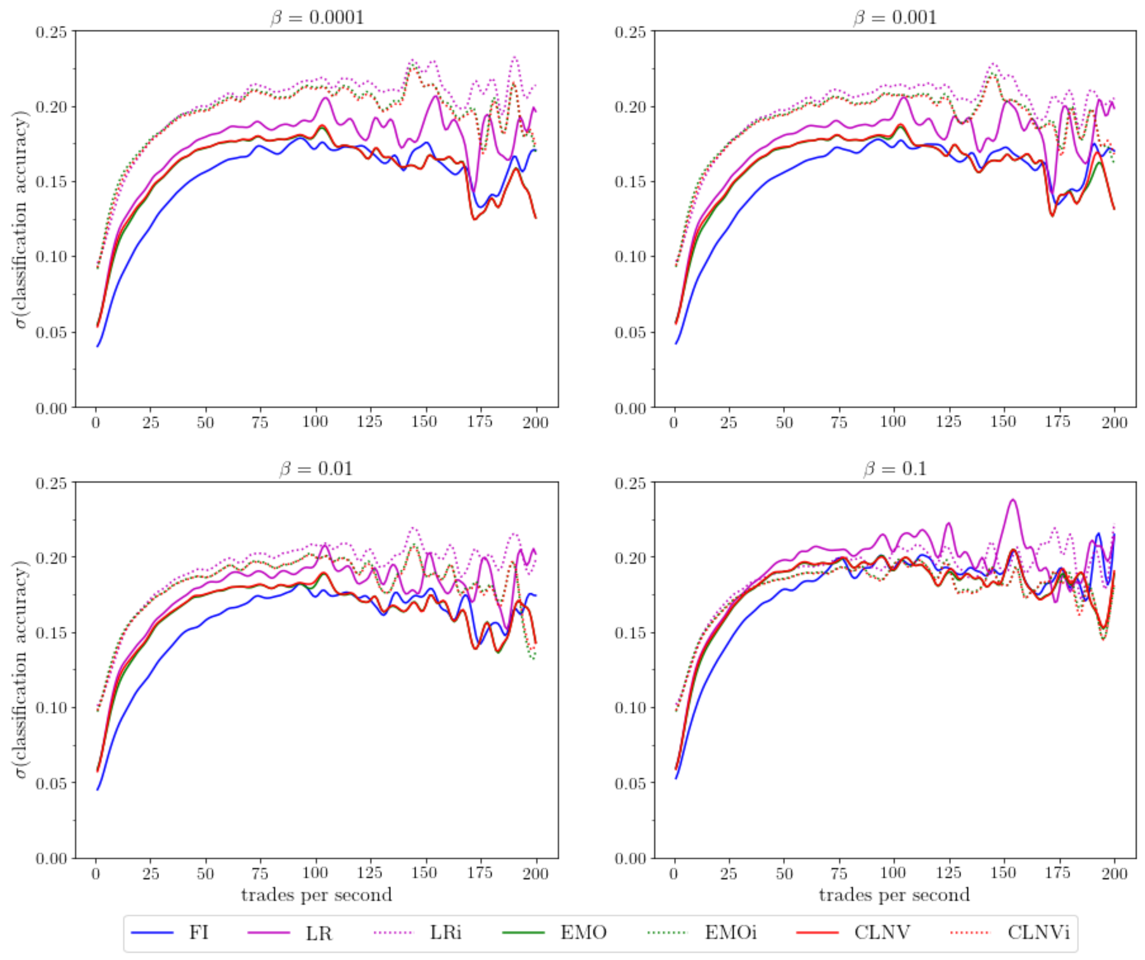


Notes: The Figure shows the correctly classified volume over total volume (y-axis) across seconds with the same number of trades (x-axis). Each data series has been smoothed with a standard Gaussian kernel. Trade and quote timestamp are subject to exponential noise $\varepsilon \sim \text{Exp}(1/\beta)$ with β varying from 10^{-4} to 10^{-1}. The grey line depicts the distribution of the number of trades per second.

trading intervals has less than 75 trades.

Another advantage of the FI algorithm is to provide less volatile classification accuracies at the same or improved level of accuracy. Figure 8 provides the standard deviations across the classification accuracies of timestamps with the same number of trades. Quite generally the FI_2 algorithm provides less volatile classification accuracies. At around 100 trades or more per timestamp the CLNV and EMO standard deviations converge to that of the FI_2 algorithm, while the LR algorithm remains more volatile.

Figure 8: Standard Deviation of Classification Accuracy for given Number of Trades Per Second



Notes: The Figure shows the standard deviation of the fraction of correctly classified volume (y-axis) for a given number trades per second (x-axis). Standard deviation is measured over stock-days, i.e. classification accuracies over seconds with the same number of trades have been aggregated within each stock-day. Each data series has been smoothed with a Gaussian kernel with standard deviation of 2. Trade and quote timestamp are subject to exponential noise $\varepsilon \sim \text{Exp}(1/\beta)$ with β varying from 10^{-4} to 10^{-1}.

6 Portfolio Optimization under Transaction Costs

To demonstrate the value of the improved classification accuracy by the FI algorithm, I will apply the competing algorithms (excluding the interpolation method) to the estimation of transaction costs, which are in turn used in a portfolio optimization exercise assuming an investor with mean-variance utility function.[15] The differences

[15]I would like to thank Evangelos Benos for the suggestion of the application to portfolio optimization.

in the investor's utility obtained under the different transaction cost estimates represent the sure return that the investor would be willing to give up to use one estimate rather than the other.

The optimization problem is given by the quadratic program

$$\max_{x^+,x^-} x'\mu - k'(x^+ + x^-) - \gamma x'\mathbb{V}x/2 \tag{1}$$

$$\text{s.t.}$$

$$x^+, x^- \geq 0 \tag{2}$$

$$x^+ - x^- + x_0 = x \tag{3}$$

$$\mathbf{1}'x = 1 \tag{4}$$

$$x \geq 0 \tag{5}$$

where x is a vector of the assets' weights in the portfolio, μ is a vector of expected returns, k is a vector of proportional transaction costs, x^+ and x^- are vectors of increases and decreases in asset allocations, respectively, γ is the investor's risk aversion, $\mathbb{V}$ is the assets' covariance matrix and x_0 is the initial asset allocation.

6.1 Transaction Cost Estimation

To estimate the proportional transaction costs for a given asset I will conduct a simple price impact regression. First, I sort transactions into groups of successive trades with the same trade direction as identified by the respective algorithm. Each group represents a single buy or sell order. For each order I then compute the execution cost

$$p_i = o_i \sum_{t=1}^{\tau_i} (p_{it} - m_i) v_{it}, \tag{6}$$

where o_i is the trade direction of the i-th order (1 for a buy, -1 for a sell order), $\{p_{it}, v_{it}\}_i^{\tau_i}$ are the transaction prices (in log) and volumes of all trades belonging to the i-th order, and m_i is the mid-quote (also in log) at the time of the order. The mid-quote is given by the mid-point of the ask and bid as quoted at the beginning of the order execution at the given timestamp precision.

Finally, I regress the execution costs on the total volume of the order

$$p_i = \beta_0 + \beta_1 \sum_{t=1}^{\tau_i} v_{it} + \epsilon_i. \tag{7}$$

The least-squares estimator of β_1 provides the estimate for the proportional transaction costs.

Table 7 and 8 in the Appendix provide the estimated transaction costs based on the true trade initiator label, as well as the biases and root-mean-square errors (RMSE) from the estimates obtained from the inferred trade initiator labels. The FI algorithm outperforms the traditional algorithms often with bias improvements by an order of magnitude. Exceptions are the LR algorithm under Data Structure 1 at higher than millisecond precision, where it performs slightly better than the FI algorithm, and the traditional algorithms at higher than second or 10-th of a second timestamp precision at strong noise.

6.2 Simulation Setup

I will allocate the investor a portfolio of 100 randomly drawn assets on a randomly drawn day. The initial allocation is drawn from a Dirichlet distribution, $Dir(\alpha)$, with $\alpha = (5, \cdots, 5)'$. To focus the exercise on transaction costs the expected return is assumed to be zero. The covariance matrix is estimated using Ledoit and Wolf's (2004) estimator from intra-day data on the selected day using 60 minutes returns.[16] The estimation of the covariance matrix is not vital for this exercise only in so far as that its quadratic form determines the trade-off between rebalancing (and thus incurring transaction costs) and not rebalancing. Given that this exercise would be pointless if the investor would choose not to rebalance the portfolio at all, I will scale the estimated matrix by 1 to 100. γ is fixed at 1.

To focus the exercise on the ability of the different algorithms to result in accurate transaction cost estimates and not let the results be driven by particular forecasting models, I will conduct the exercise in-sample. That is, I will compare the attained utility under the transaction costs estimated from trade initiators identified by the different algorithms on the selected day against the utility obtained from using the transaction costs estimated from the true initiator label on the same day.

[16]Pooter et al. (2008) show that lower sampling frequencies are the preferred choice compared to the common 5 minute sampling scheme when estimating covariance matrices from intra-day data in the context of mean-variance portfolio optimization with daily rebalancing.

6.3 Optimization Results

Figure 9 presents the simulation results for Data Structure 1 (DS1) and 2 (DS2) over 1000 runs at a timestamp precision of seconds. The graph shows the differences in the investor's utility if the asset allocation were derived under the knowledge of the true trade initiator compared to being estimated by the respective algorithm, which we may interpret as an algorithm fee,

$$\Delta_{algo} \equiv U_{k,\mathbb{V},\gamma}(x^+_{opt}, x^-_{opt}) - U_{k,\mathbb{V},\gamma}(x^+_{algo}, x^-_{algo}), \tag{8}$$

where $U_{k,\mathbb{V},\gamma}(\cdot)$ is the investor's mean-variance utility function evaluated at given transaction costs, return covariance matrix and risk aversion. x^+_{algo}, x^-_{algo} is the optimal solution to the program in (1) estimating k based on algorithm $algo \in$ {FI,LR,EMO,CLNV}, and x^+_{opt}, x^-_{opt} is the optimal solution based on the true trade initiator. Fees are shown in basis points, annualized assuming 253 trading days.

Under DS1, at its maximum difference the fee the investor incurs from using the LR algorithm is 45.82 bps compared to 12.65 bps for the FI algorithm, a difference of around 33.17 bps per annum. For the EMO and CLNV algorithm the maximum fee difference to the FI algorithm is roughly 26 bps.

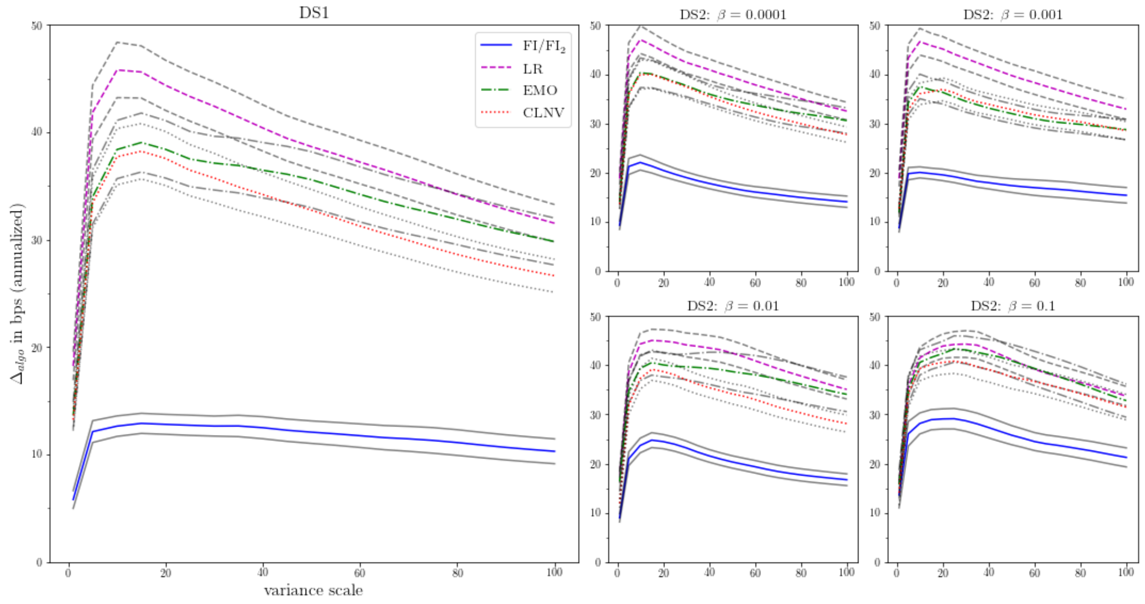
The improvements of the FI algorithm remain significant even under DS2 with strong noise at the timestamp. With $\beta = 0.1$, despite the overall similar level of classification accuracy, the maximum fee difference to the CLNV algorithm is around 12 bps and 16 bps if compared to the LR algorithm.

7 Comparison to BVC

The Bulk-Volume classification algorithm (BVC) of Easley et al. (2012) is not directly targeted at estimating the trade initiator as it is believed that the trade initiator only insufficiently reflects the trading intention in today's markets. Instead it is designed to better link the order imbalance with the information contained in the order flow. The measure has become a popular choice in the VPIN literature.

Despite the different focus of the BVC algorithm, the output produced by it can directly by taken as a substitute for the order imbalance constructed from the trade initiator label. So even if the interest is in the traditional trade initiator, in an application involving the order imbalance the choice may still fall on the BVC algorithm if it happens to produce more accurate results.

Figure 9: Trade Classification Algorithm Fees



Notes: This Figure depicts the algorithm fee defined in (8) which is the maximum sure return a mean-variance utility investor would give up to optimize the portfolio knowing the true trade initiator instead of inferring it using the respective algorithm. The optimization problem is defined in (1) with μ set to zero and k estimated from a price impact regression with the knowledge of the true trade initiator or it being inferred using one of the algorithms. γ is fixed to 1 and the covariance matrix is scaled from 1 to 100 (x-axis). The optimization is solved for 1000 randomly selected days at which the investor is given a universe of 100 randomly selected assets with random initial portfolio weights. Results are presented for Data Structure 1 (DS1) and 2 (DS2). Under DS2 timestamps are subject to noise, $\varepsilon \sim \text{Exp}(1/\beta)$. The data is timestamped to seconds. Grey lines show 2*standard errors around the fees.

The BVC algorithm splits the trading day into intervals of equal size (either by volume or time) and takes the last price from each interval to compute between-interval returns, Δp_τ. These returns are then standardized and mapped onto a zero to one scale by transforming it using a cumulative density function of a symmetric distribution, F_{df}, such as the normal or t-distribution. The estimated buyer-initiated volume is then given

$$\hat{V}_{B,\tau} = V_\tau F_{df}\Big(\frac{\Delta p_\tau}{\sigma_{\Delta p}}\Big), \tag{9}$$

where V_τ is the trading volume over the τ-th interval.

The performance of the BVC algorithm has been tested against the LR algorithm in earlier studies (Chakrabarty et al., 2015; Panayides et al., 2019), with the finding that the LR algorithm outperforms BVC. I add to this literature the comparison under noisy timestamps to which the BVC algorithm may prove more robust given that it does not rely on an accurate matching of trades to quotes. However, to the extent that end-interval prices are affected by prices being out of order the accuracy of the BVC algorithm is impacted as well.

For this exercise I will focus on the FI and LR algorithm applied to Data Structure 1 and 2 at a timestamp precision of seconds. The BVC algorithm is applied at each stock-day individually using different interval sizes of both time and volume type.[17] The distribution F_{df} is taken to be the t-distribution with df either set to 0.25 as in Easley et al. (2016) or estimated from the between-interval returns. The accuracy of the signed volume across all intervals of a given stock-day is evaluated following Chakrabarty et al. (2015) by

$$Ac^{algo} = \frac{\sum_\tau \sum_{j\in\{B,S\}} \min(\hat{V}^{algo}_{j,\tau}, V_{j,\tau})}{\sum_\tau V_\tau}, \tag{10}$$

where $V_{S,\tau} = V_\tau - V_{B,\tau}$ is the seller initiated volume. The estimated buyer- and seller-initiated volumes,$\hat{V}^{algo}_{B,\tau}$ and $\hat{V}^{algo}_{S,\tau}$, for the BVC algorithm are as defined above. The buyer-initiated volume computed from the trade initiator flag is given by $V_{B,\tau} = \sum_{i=1}^{I_\tau} v_i \mathbb{1}_{\{o_i=1\}}$, where $\sum_i^{I_\tau} v_i = V_\tau$, o_i is the trade initiator (1 for buyer-initiated, -1 for seller initiated), and $\mathbb{1}$ is the indicator function.

Tables 2 and 3 provide the average classification accuracies across all stock-days for time and volume intervals, respectively. The algorithms are applied to the data timestamped to the second under both data structures with timestamp noise governed by the Exponential distribution, $\varepsilon \sim \text{Exp}(1/\beta)$, with β ranging from 0.0001 to 0.1.

The results show that the BVC algorithm cannot compete with the FI or the LR algorithm if the estimation target is the trade initiator. While the BVC algorithm performs better than in Chakrabarty et al. (2015) it is outperformed considerably by the other two under both data structures.

[17]Trades are added to a volume interval until it has reached its specified size. Trades are never split between intervals, i.e. volume intervals will typically be greater than their specified size by a fraction of a trade. Empty time intervals are dropped. The first trade on each stock-day is used for the first price of the day and omitted from the construction of the intervals.

Table 2: Comparison to BVC - time intervals

β		interval size in seconds									
		1	5	10	30	60	120	300	600	1800	3600
Panel A: Data Structure 1											
	$BVC_{0.25}$	62.30	65.76	68.23	73.51	77.34	80.97	84.73	86.60	88.07	88.34
	$BVC_{\hat{df}}$	66.12	69.42	71.31	74.28	75.98	77.30	78.14	78.10	77.13	76.65
	FI	95.34	95.47	95.58	95.85	96.11	96.45	96.96	97.37	97.96	98.24
	LR	89.51	89.86	90.12	90.83	91.48	92.30	93.55	94.52	95.91	96.55
Panel B: Data Structure 2											
0.0001	$BVC_{0.25}$	62.31	65.77	68.23	73.51	77.34	80.97	84.73	86.60	88.07	88.34
	$BVC_{\hat{df}}$	66.11	69.42	71.31	74.28	75.98	77.30	78.14	78.10	77.13	76.65
	FI_2	92.82	93.07	93.25	93.71	94.14	94.67	95.50	96.16	97.10	97.53
	LR	89.38	89.74	90.01	90.73	91.39	92.22	93.48	94.46	95.87	96.52
0.0010	$BVC_{0.25}$	62.32	65.77	68.24	73.51	77.34	80.97	84.73	86.60	88.07	88.34
	$BVC_{\hat{df}}$	66.10	69.41	71.31	74.28	75.98	77.30	78.14	78.10	77.13	76.65
	FI_2	92.23	92.59	92.80	93.30	93.77	94.36	95.25	95.95	96.96	97.42
	LR	88.87	89.34	89.64	90.38	91.07	91.93	93.23	94.25	95.71	96.39
0.0100	$BVC_{0.25}$	62.20	65.71	68.19	73.48	77.32	80.96	84.73	86.60	88.07	88.34
	$BVC_{\hat{df}}$	65.89	69.26	71.19	74.23	75.95	77.29	78.14	78.10	77.13	76.64
	FI_2	90.30	91.56	91.91	92.58	93.14	93.83	94.86	95.65	96.76	97.27
	LR	87.24	88.60	89.02	89.89	90.63	91.55	92.93	94.00	95.53	96.24
0.1000	$BVC_{0.25}$	61.74	65.49	68.03	73.41	77.28	80.94	84.72	86.59	88.07	88.34
	$BVC_{\hat{df}}$	65.20	68.84	70.89	74.08	75.87	77.25	78.13	78.09	77.12	76.65
	FI_2	77.90	86.64	88.06	89.70	90.73	91.85	93.40	94.53	96.05	96.72
	LR	75.86	84.63	86.07	87.79	88.89	90.14	91.91	93.24	95.08	95.90

Notes: This table shows the classification accuracy defined in (10) averaged over all stock-days. The algorithms are applied to the data timestamped to the second and under both data structures defined in Data Structure 1 and 2. Under Data Structure 2 trade and quote timestamps are subject to Exponential noise, $\varepsilon \sim \text{Exp}(1/\beta)$ with β varying from 0.0001 to 0.1. The BVC algorithm is applied using the t-distribution with the degrees of freedom either set to 0.25 ($BVC_{0.25}$) or estimated from the intra-day interval returns ($BVC_{\hat{df}}$). Days are split into equally sized time intervals.

8 Conclusion

This paper offers a number of practical recommendation where the trade initiator has to be inferred from the data. First, traditional algorithms perform reasonably well even in very active markets *if* combined with the traditional quote matching rule of using the last quotes from before the time of the trade. The reliability of traditional algorithms has been questioned in light of the increase in trading and quotation frequency. I do not find, however, that the algorithms perform material worse than their historical performance documented in other studies. Secondly, the

Table 3: Comparison to BVC - volume intervals

β		volume intervals in 1000 shares 1	5	10	20	30	40	50	100	150	200
Panel A: Data Structure 1											
	$\text{BVC}_{0.25}$	73.54	82.88	85.49	87.31	88.01	88.40	88.59	88.81	88.69	88.53
	$\text{BVC}_{\hat{df}}$	72.73	77.96	78.59	78.51	78.14	77.83	77.53	76.27	75.37	74.66
	FI	95.42	96.57	97.12	97.62	97.88	98.05	98.17	98.49	98.67	98.78
	LR	90.58	92.96	94.08	95.10	95.63	95.98	96.22	96.87	97.18	97.38
Panel B: Data Structure 2											
0.0001	$\text{BVC}_{0.25}$	72.35	82.65	85.39	87.26	87.99	88.38	88.58	88.80	88.69	88.53
	$\text{BVC}_{\hat{df}}$	71.02	77.61	78.44	78.46	78.10	77.81	77.51	76.27	75.37	74.65
	FI_2	86.08	93.49	95.03	96.12	96.62	96.92	97.10	97.63	97.88	98.02
	LR	83.78	91.64	93.48	94.85	95.49	95.88	96.14	96.84	97.17	97.37
0.0010	$\text{BVC}_{0.25}$	71.27	82.23	85.19	87.17	87.93	88.34	88.55	88.79	88.68	88.53
	$\text{BVC}_{\hat{df}}$	69.36	76.95	78.12	78.30	78.00	77.73	77.46	76.24	75.36	74.65
	FI_2	79.46	91.10	93.85	95.55	96.22	96.63	96.89	97.52	97.82	97.98
	LR	77.52	89.33	92.31	94.26	95.07	95.57	95.90	96.71	97.09	97.30
0.0100	$\text{BVC}_{0.25}$	70.70	81.97	85.02	87.09	87.88	88.30	88.52	88.78	88.68	88.52
	$\text{BVC}_{\hat{df}}$	68.46	76.54	77.86	78.17	77.93	77.68	77.41	76.22	75.34	74.64
	FI_2	75.98	89.63	92.88	95.01	95.87	96.34	96.65	97.39	97.70	97.89
	LR	74.29	87.97	91.41	93.75	94.73	95.30	95.67	96.59	96.99	97.23
0.1000	$\text{BVC}_{0.25}$	70.37	81.78	84.92	87.03	87.84	88.28	88.49	88.76	88.67	88.52
	$\text{BVC}_{\hat{df}}$	67.94	76.25	77.71	78.10	77.85	77.64	77.37	76.21	75.32	74.65
	FI_2	73.56	87.96	91.65	94.04	95.03	95.59	95.93	96.82	97.20	97.43
	LR	72.20	86.61	90.47	93.06	94.15	94.80	95.20	96.25	96.71	96.99

Notes: This table shows the classification accuracy defined in (10) averaged over all stock-days. The algorithms are applied to the data timestamped to the second and under both data structures defined in Data Structure 1 and 2. Under Data Structure 2 trade and quote timestamps are subject to Exponential noise, $\varepsilon \sim \text{Exp}(1/\beta)$ with β varying from 0.0001 to 0.1. The BVC algorithm is applied using the t-distribution with the degrees of freedom either set to 0.25 ($\text{BVC}_{0.25}$) or estimated from the intra-day interval returns ($\text{BVC}_{\hat{df}}$). Days are split into volume intervals.

interpolation of timestamps as suggested by Holden and Jacobsen (2014) to counter the problem of relatively imprecise timestamps cannot be advocated based on the results in this paper. This is an important finding given that the method has gained considerable following but its reliability has never been tested directly. Also the Bulk-Volume classification algorithm of Easley et al. (2012), which could substitute for the traditional algorithms in applications involving the order imbalance, cannot be recommended. Third, even though the Lee and Ready (1991) algorithm is the default choice for trade classification—possibly partly due to being automatically supplied by data vendors, partly due to its simplicity—the similar simplistic algorithms of

Chakrabarty et al. (2007) and Ellis et al. (2000) tend to perform better and may be preferred in certain applications.

However, typical data sources will contain information which this paper has shown can be used to obtain more accurate and robust estimates of the trade initiator, including over a wide range of order book activity levels. This paper provides two versions of an algorithm, depending on the granularity and accuracy of the data structure, that outperform the traditional algorithms.[18] The paper further shows that the outperformance transmits to applications that need the trade initiator to be identified, such as the estimation of transaction costs.

The algorithm is arguably more complex, but the versions presented in this paper are available online and can easily be employed by anyone familiar with the increasingly popular `python` language.[19] It should also be noted that the increased complexity does not come at any noteworthy computational costs.

[18]A third version for an intermediate data structure is available online.

[19]https://github.com/jktis

Table 4: Traditional Trade Classification Algorithms

Variables: p_i – transaction price of the i^{th} trade; a_i, b_i – ask, bid price corresponding to the i^{th} trade; o_i – trade initiator; $i = 1, \dots, I$

Tick-Test

```
for $i = 2 : I$ do
    $j = 0$
    while $i - j > 0$ do
        $j = j + 1$
        if $p_i > p_{i-j}$ then
            $o_i$ = buyer
            break
        else if $p_i < p_{i-j}$ then
            $o_i$ = seller
            break
```

LR

```
for $i = 1 : I$ do
    $m_i = (a_i + b_i)/2$
    if $p_i > m_i$ then
        $o_i$ = buyer
    else if $p_i < m_i$ then
        $o_i$ = seller
    else
        apply tick-test
```

EMO

```
for $i = 1 : I$ do
    if $p_i = a_i$ then
        $o_i$ = buyer
    else if $p_i = b_i$ then
        $o_i$ = seller
    else
        apply tick-test
```

CLNV

```
for $i = 1 : I$ do
    $\underline{a} = 0.7a_i + 0.3b_i$
    $\overline{b} = 0.3a_i + 0.7b_i$
    if $\underline{a} < p_i \leq a_i$ then
        $o_i$ = buyer
    else if $b_i \leq p_i < \overline{b}$ then
        $o_i$ = seller
    else
        apply tick-test
```

Table 5: Summary Statistics of NASDAQ's Transaction and Quote Data

	mean	std	min	25%	median	75%	max
T	6776.01	7647.55	17	1673	4397	9247.75	106407
V	1131.88	2347.50	1.37	165.53	467.49	1160.76	58115.01
V/T	129.02	87.20	44.65	95.50	108.31	129.33	3573.15
$\%\{V \geq 100\}$	77.34	11.90	21.30	70.51	79.02	86.62	97.61
$\%\{V = 100\}$	62.02	11.83	18.98	54.79	63.26	70.23	91.88
P	59.01	54.25	4.48	32.07	48.22	69.96	623.37
$\#Q$	104.10	102.26	1.16	28.62	70.39	150.38	908.74

Notes: This table provides summary statistics to the following variables computed for each stock-day: T – Number of trades, V – Trading volume in 1000 shares, V/T Volume per trade (stock-day average), $\%\{V \geq 100\}$ – Percentage of trades with volume greater or equal to 100 shares, $\%\{V = 100\}$ – Percentage of trades with trading volume equal to 100 shares, P – Price per share (stock-day average), $\#Q$ – Number of quote changes (including changes only in the volume at the quotes) in 1000.

Table 6: Classification Accuracy at Different Timestamp Precisions

	timestamp precision 10^{-i} of a second with $i =$					
	0 (s)	1	2	3 (ms)	4	9 (ns)
Panel A: total volume						
FI	95.02	96.97	97.95	98.34	98.24	98.18
EMO	90.14	93.63	96.08	97.61	98.17	98.19
CLNV	90.13	93.64	96.08	97.61	98.18	98.20
LR	89.84	93.56	96.02	97.51	98.09	98.09
EMOi	75.26	78.87	84.30	92.10	97.49	97.61
CLNVi	74.92	78.58	84.07	91.96	97.46	97.58
LRi	73.79	77.81	83.52	91.59	97.29	97.41
Panel B: average volume						
FI	94.52	96.47	97.38	97.71	97.53	97.44
	(2.40)	(1.91)	(1.91)	(2.04)	(2.05)	(2.09)
EMO	89.39	92.69	94.92	96.55	97.42	97.42
	(3.40)	(3.01)	(2.85)	(2.54)	(2.07)	(2.10)
CLNV	89.49	92.75	94.93	96.51	97.45	97.43
	(3.42)	(2.99)	(2.86)	(2.61)	(2.07)	2.11
LR	89.27	92.64	94.77	96.29	97.24	97.19
	(3.57)	(3.10)	(3.02)	(2.78)	(2.28)	(2.34)
EMOi	76.87	80.41	84.97	91.39	96.57	96.69
	(6.78)	(6.08)	(4.85)	(3.26)	(2.40)	(2.41)
CLNVi	75.70	79.40	84.15	90.89	96.48	96.60
	(6.44)	(5.91)	(4.97)	(3.68)	(2.56)	(2.57)
LRi	73.36	77.59	82.77	90.02	96.09	96.21
	(5.85)	(5.51)	(5.01)	(4.14)	(2.96)	(2.97)

Notes: This table shows the percentage of correctly classified trading volume by the FI algorithm and the traditional algorithms using the last quotes from before the time of the trade (EMO, CLNV and LR) and using the interpolated time of trades and quotes (EMOi, CLNVi and LRi) as suggested by Holden and Jacobsen (2014). The algorithms are applied to the data with reduced timestamp precisions (10^{-i} of a second for $i = 0, \dots, 4$) and using the original precision of nanoseconds (10^{-9} of a second). These correspond to a median number of quote changes at the time of trades ranging from 17 (for $i = 0$) to 1 (for $i = 9$). Panel A shows the percentage of correctly classified volume summed over the entire sample. Panel B shows the average of correctly classified volume over the 19842 stock-days with the standard deviations in brackets.

Table 7: Estimated Transaction Costs - DS1

timestamp precision 10^{-i} of a second for $i =$		Bias				RMSE			
	$\hat{\beta}_1$	FI	CLNV	EMO	LR	FI	CLNV	EMO	LR
0 (s)	0.0580	0.0033	0.0163***	0.0155***	0.0204***	0.0224	0.0403	0.0374	0.0445
1	0.0598	0.0020	0.0116***	0.0110***	0.0127***	0.0207	0.0325	0.0319	0.0321
2	0.0608	0.0016	0.0079***	0.0075***	0.0076***	0.0200	0.0288	0.0272	0.0281
3 (ms)	0.0620	0.0022	0.0047***	0.0045***	0.0041***	0.0199	0.0230	0.0226	0.0223
4	0.0628	0.0025	0.0027	0.0026	0.0020***	0.0199	0.0201	0.0199	0.0198
9 (ns)	0.0629	0.0025	0.0026	0.0025	0.0018***	0.0218	0.0220	0.0199	0.0216

Notes: This table shows the results from the price impact regression outlined in Section 6 for Data Structure 1. $\hat{\beta}_1$ is the average estimator of β_1 in the regression $p_i = \beta_0 + \beta_1 v_i + \epsilon_i$ across all stock-days where p_i is the execution cost of the i-th order on a given stock-day calculated using the true trade initiator label, and v_i is the total volume of the order. Estimators are multiplied by 100 to be comparable to percentage returns. Execution costs are calculated at different timestamp precisions. The Bias columns show the difference between $\hat{\beta}_1$ and the estimator obtained from the respective algorithm. Asterisks mark significant differences in the bias of the FI estimator compared to the respective alternative estimator using a two-sided Welch t-test (***: p-value< 0.01). RMSE is the root-mean-square error.

Table 8: Estimated Transaction Costs - DS3 (Part I)

timestamp precision in seconds	$\hat{\beta}_1$	Bias				RMSE			
		FI	CLNV	EMO	LR	FI	CLNV	EMO	LR
Panel A: $\beta = 0.0001$									
2s	0.0575	0.0081	0.0184***	0.0176***	0.0242***	0.0282	0.0403	0.0402	0.0467
1.5s	0.0580	0.0072	0.0174***	0.0167***	0.0223***	0.0269	0.0384	0.0388	0.0454
s	0.0582	0.0062	0.0164***	0.0156***	0.0205***	0.0270	0.0413	0.0381	0.0450
$\frac{1}{10}$s	0.0599	0.0027	0.0117***	0.0111***	0.0128***	0.0249	0.0336	0.0324	0.0332
$\frac{1}{100}s$	0.0610	0.0011	0.0080***	0.0076***	0.0076***	0.0233	0.0299	0.0282	0.0291
ms	0.0616	0.0010	0.0047***	0.0047***	0.0038***	0.0246	0.0274	0.0253	0.0273
Panel B: $\beta = 0.001$									
2s	0.0576	0.0081	0.0185***	0.0173***	0.0245***	0.0294	0.0412	0.0407	0.0480
1.5s	0.0581	0.0071	0.0175***	0.0163***	0.0226***	0.0306	0.0395	0.0389	0.0454
s	0.0583	0.0064	0.0166***	0.0153***	0.0208***	0.0282	0.0414	0.0386	0.0448
$\frac{1}{10}$s	0.0599	0.0028	0.0119***	0.0109***	0.0131***	0.0276	0.0359	0.0332	0.0354
$\frac{1}{100}$s	0.0604	0.0001	0.0076***	0.0068***	0.0067***	0.0276	0.0317	0.0304	0.0317
ms	0.0578	−0.0028	0.0007***	0.0009***	−0.0022*	0.0348	0.0343	0.0322	0.0337

Notes: Continued on next page.

Table 9: Estimated Transaction Costs - DS3 (Part II)

timestamp precision in seconds		Bias				RMSE			
	$\hat{\beta}_1$	FI	CLNV	EMO	LR	FI	CLNV	EMO	LR
Panel C: $\beta = 0.01$									
2s	0.0578	0.0069	0.0165***	0.0152***	0.0243***	0.0299	0.0413	0.0415	0.0488
1.5s	0.0582	0.0060	0.0155***	0.0144***	0.0226***	0.0320	0.0392	0.0394	0.0454
s	0.0585	0.0049	0.0146***	0.0134***	0.0207***	0.0297	0.0396	0.0395	0.0446
$\frac{1}{10}$s	0.0595	−0.0007	0.0088***	0.0078***	0.0106***	0.0317	0.0366	0.0360	0.0367
$\frac{1}{100}$s	0.0557	−0.0084	−0.0025***	−0.0028***	−0.0048***	0.0401	0.0404	0.0371	0.0397
ms	0.0525	−0.0076	−0.0058***	−0.0058***	−0.0096***	0.0422	0.0427	0.0404	0.0422
Panel D: $\beta = 0.1$									
2s	0.0576	0.0029	0.0126***	0.0113***	0.0223***	0.0305	0.0413	0.0410	0.0478
1.5s	0.0579	0.0015	0.0114***	0.0100***	0.0198***	0.0312	0.0401	0.0401	0.0440
s	0.0579	−0.0001	0.0098***	0.0085***	0.0168***	0.0322	0.0402	0.0397	0.0428
$\frac{1}{10}$s	0.0540	−0.0111	−0.0044***	−0.0048***	−0.0056***	0.0405	0.0394	0.0389	0.0389
$\frac{1}{100}$s	0.0494	−0.0100	−0.0078***	−0.0078***	−0.0111**	0.0426	0.0420	0.0396	0.0430
ms	0.0484	−0.0088	−0.0079**	−0.0079**	−0.0114***	0.0440	0.0439	0.0414	0.0452

Notes: This table shows the results from the price impact regression outlined in Section 6 for Data Structure 2. $\hat{\beta}_1$ is the average estimator of β_1 in the regression $p_i = \beta_0 + \beta_1 v_i + \epsilon_i$ across all stock-days where p_i is the execution cost of the i-th order on a given stock-day calculated using the true trade initiator label, and v_i is the total volume of the order. Estimators are multiplied by 100 to be comparable to percentage returns. Execution costs are calculated at different timestamp precisions and degrees of timestamp noise, $\varepsilon \sim \text{Exp}(1/\beta)$. The Bias columns show the difference between $\hat{\beta}_1$ and the estimator obtained from the respective algorithm. Asterisks mark significant differences in the bias of the FI estimator compared to the respective alternative estimator using a two-sided Welch t-test (*: p-value< 0.1, **: p-value< 0.05, ***: p-value< 0.01). RMSE is the root-mean-square error.

References

Angel, J.J., Harris, L.E., Spatt, C.S., 2011. Equity trading in the 21st century. The Quarterly Journal of Finance 1, 1–53.

Angel, J.J., Harris, L.E., Spatt, C.S., 2015. Equity trading in the 21st century: An update. The Quarterly Journal of Finance 5, 1–39.

Bernile, G., Hu, J., Tang, Y., 2016. Can information be locked up? Informed trading ahead of macro-news announcements. Journal of Financial Economics 121, 496–520.

Bessembinder, H., 2003. Issues in assessing trade execution costs. Journal of Financial Markets 6, 233–257.

Bicu-Lieb, A., Chen, L., Elliott, D., 2020. The leverage ratio and liquidity in the gilt and gilt repo markets. Journal of Financial Markets 48, 100510.

Brennan, M.J., Huh, S.W., Subrahmanyam, A., 2018. High-frequency measures of informed trading and corporate announcements. The Review of Financial Studies 31, 2326–2376.

Brogaard, J., Carrion, A., Moyaert, T., Riordan, R., Shkilko, A., Sokolov, K., 2018. High frequency trading and extreme price movements. Journal of Financial Economics 128, 253–265.

Chakrabarty, B., Li, B., Nguyen, V., Van Ness, R.A., 2007. Trade classification algorithms for electronic communications network trades. Journal of Banking & Finance 31, 3806–3821.

Chakrabarty, B., Moulton, P.C., Shkilko, A., 2012. Short sales, long sales, and the Lee–Ready trade classification algorithm revisited. Journal of Financial Markets 15, 467–491.

Chakrabarty, B., Pascual, R., Shkilko, A., 2015. Evaluating trade classification algorithms: Bulk volume classification versus the tick rule and the Lee-Ready algorithm. Journal of Financial Markets 25, 52–79.

Chordia, T., Hu, J., Subrahmanyam, A., Tong, Q., 2017. Order flow volatility and equity costs of capital. Management Science Forthcoming.

Easley, D., de Prado, M.L., O'Hara, M., 2016. Discerning information from trade data. Journal of Financial Economics 120, 269–285.

Easley, D., de Prado, M.M.L., O'Hara, M., 2012. Flow toxicity and liquidity in a high-frequency world. Review of Financial Studies 25, 1457–1493.

Ellis, K., Michaely, R., O'Hara, M., 2000. The accuracy of trade classification rules: Evidence from Nasdaq. Journal of Financial and Quantitative Analysis 35, 529–551.

Finucane, T.J., 2000. A direct test of methods for inferring trade direction from intra-day data. Journal of Financial and Quantitative Analysis 35, 553–576.

Fong, K.Y., Holden, C.W., Trzcinka, C.A., 2017. What are the best liquidity proxies for global research? Review of Finance 21, 1355–1401.

Henker, T., Wang, J.X., 2006. On the importance of timing specifications in market microstructure research. Journal of Financial Markets 9, 162–179.

Holden, C.W., Jacobsen, S., 2014. Liquidity measurement problems in fast, competitive markets: expensive and cheap solutions. The Journal of Finance 69, 1747–1785.

Hu, J., 2014. Does option trading convey stock price information? Journal of Financial Economics 111, 625–645.

Hu, J., 2017. Option listing and information asymmetry. Review of Finance Forthcoming.

Huang, R.D., Stoll, H.R., 1996. Dealer versus auction markets: A paired comparison of execution costs on NASDAQ and the NYSE. Journal of Financial economics 41, 313–357.

Kremer, S., Nautz, D., 2013. Causes and consequences of short-term institutional herding. Journal of Banking & Finance 37, 1676–1686.

Ledoit, O., Wolf, M., 2004. A well-conditioned estimator for large-dimensional covariance matrices. Journal of Multivariate Analysis 88, 365 – 411.

Lee, C., Ready, M.J., 1991. Inferring trade direction from intraday data. The Journal of Finance 46, 733–746.

Lee, C.M., Radhakrishna, B., 2000. Inferring investor behavior: Evidence from torq data. Journal of Financial Markets 3, 83–111.

Muravyev, D., 2016. Order flow and expected option returns. The Journal of Finance 71, 673–708.

Odders-White, E.R., 2000. On the occurrence and consequences of inaccurate trade classification. Journal of Financial Markets 3, 259–286.

O'Hara, M., 2015. High frequency market microstructure. Journal of Financial Economics 116, 257–270.

Panayides, M.A., Shohfi, T.D., Smith, J.D., 2019. Bulk volume classification and information detection. Journal of Banking & Finance 103, 113–129.

Perlin, M., Brooks, C., Dufour, A., 2014. On the performance of the tick test. The Quarterly Review of Economics and Finance 54, 42–50.

Peterson, M., Sirri, E., 2003. Evaluation of the biases in execution cost estimation using trade and quote data. Journal of Financial Markets 6, 259–280.

Piwowar, M.S., Wei, L., 2006. The sensitivity of effective spread estimates to trade–quote matching algorithms. Electronic Markets 16, 112–129.

Pooter, M.d., Martens, M., Dijk, D.v., 2008. Predicting the daily covariance matrix for S&P 100 stocks using intraday data-but which frequency to use? Econometric Reviews 27, 199–229.

Rosenthal, D.W., 2012. Modeling trade direction. Journal of Financial Econometrics 10, 390–415.

Theissen, E., 2001. A test of the accuracy of the Lee/Ready trade classification algorithm. Journal of International Financial Markets, Institutions and Money 11, 147–165.

Vergote, O., 2005. How to match trades and quotes for nyse stocks? Center for Economic Studies Discussion Paper Series .