Computation of Implementation Shortfall for Algorithmic Trading by Sequence Alignment

Raymond Chan, Kelvin Kan, and Alfred Ma

Raymond Chan is the Dean in College of Science and a Chair Professor in the Department of Mathematics, City University of Hong Kong. Email: rchan.sci@cityu.edu.hk

Kelvin Kan is a Ph.D. student in the Department of Mathematics, Emory University. Email: kelvin.kan@emory.edu

Alfred Ma is an Adjunct Professor in the Department of Economics and Finance, Hang Seng University of Hong Kong. Email: alfredma@hsu.edu.hk

Raymond Chan's research is supported by HKRGC Grants No. CityU12500915, CityU14306316, HKRGC CRF Grant C1007-15G, and HKRGC AoE Grant AoE/M-05/12.

Abstract

Implementation shortfall measures the difference in performance between paper portfolio and real portfolio, and it is decomposed as a sum of execution cost and opportunity cost. The authors show that the original framework is not directly applicable to algorithmic trading and propose a new framework to compute implementation shortfall and its decomposition. They employ an efficient algorithm inspired by DNA sequence alignment techniques to align the trade records from both portfolios and then compute the implementation shortfall with a breakdown of execution cost and opportunity cost for diagnosis. Their framework is simple, objective, and computationally efficient—the complexity only grows linearly with respect to the numbers of trades of paper and real portfolios. Hence the framework proposed by the authors in this article is applicable to high frequency trading data.

Keywords: Implementation Shortfall, Algorithmic Trading, Algorithmic Trading System, Backtesting, Sequence Alignment

Implementation shortfall, as suggested by Perold (1988), concerns the difference in performance between the paper portfolio and the real portfolio. Perold (1988) breaks down the difference into execution cost, which is the cost of trading, and opportunity cost, which is the cost of not trading. The breakdown is essential to determine the best execution in the sense of Wagner and Edwards (1993), i.e., considering all the implicit transaction costs from the entire process of implementation. After all, as Peter Drucker stated "you can't manage what you can't measure". Accurate measurement is essential for continuous improvement of the implementation.

In the original Perold's framework, we can compute implementation shortfall over periods in which there is no trading in the paper portfolio. This setting works well for traditional portfolio management but not for algorithmic trading because a trading strategy can have lots of trading activities even for a very short period of time.

As an illustration of Perold's framework, Exhibit 1 shows trade records from a paper portfolio and a real portfolio of a particular financial instrument. Since the buying price in

Exhibit 1: An example of paper and real portfolios

Paper portfolio				Real portfolio			
Time Stamp	Price	Volume	Type	Time Stamp	Price	Volume	Type
Jan 8, 2018	10.12	200	Buy	Jan 8, 2018	10.13	100	Buy
				Jan 8, 2018	10.14	50	Buy

the real portfolio is higher than the buying price in the paper portfolio, there is an execution cost for buying the asset and it is computed by $(10.13-10.12)\times 100+(10.14-10.12)\times 50 = 2$. Besides execution cost, there is an opportunity cost caused by failure to buy the financial instrument in the real portfolio where we only bought 150 units instead of 200 units. Hence, there is an opportunity cost of failure to buy the 50 units. Assume that the market closes at 10.20, the opportunity cost is computed by $(10.20-10.12)\times(200-150) = 4$. The profit of the paper portfolio is $(10.20 - 10.12) \times 200 = 16$ while the profit of the real portfolio is $(10.20 - 10.13) \times 100 + (10.20 - 10.14) \times 50 = 10$. The difference $(16 - 10 = 6)$ is the implementation shortfall which is also the sum of the execution cost and opportunity cost $(2 + 4 = 6)$. We

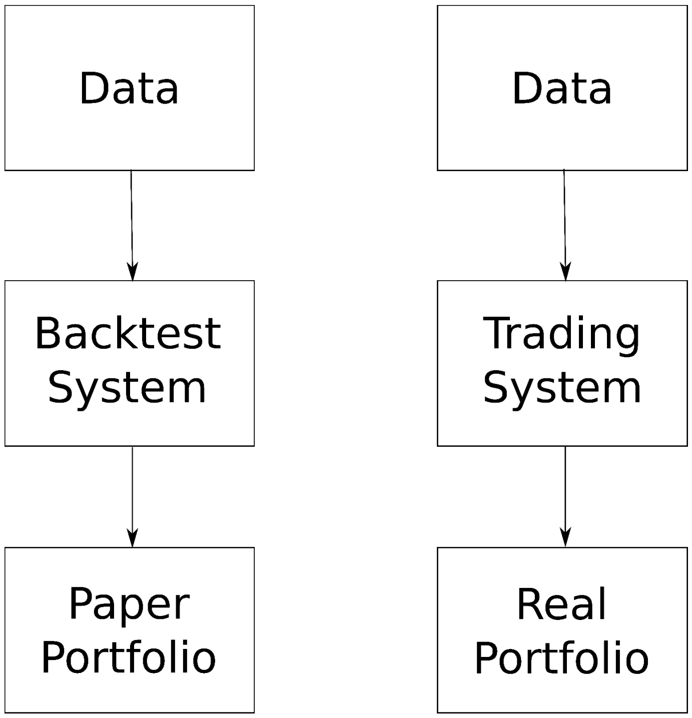

Exhibit 2: Paper and real portfolio from algorithmic trading

then consider algorithmic trading as shown in Exhibit 2. In practice, the paper portfolio is found by a backtest, see Bailey *et al.* (2016) for example. In general, backtesting and trading are implemented in different systems. The paper portfolio is considered the ideal result while real portfolio shows the reality.

Exhibit 3 lists algorithmic trade records from the paper portfolio and the real portfolio of a particular financial instrument. According to the Perold's framework or other extensions such as Kissell (2006), the implementation shortfall is measured over periods of no trading in the paper portfolio. While they both do not address how the framework can be applied in our example, it is standard to divide the whole period into subperiods with no trading in the paper portfolio and apply the framework over each subperiod. In the example, the first subperiod has to be chosen strictly before the time stamp of the second trade, namely,

Exhibit 3: An example of paper and real portfolios

Paper Portfolio				Real Portfolio			
Time stamp	Price	Volume	Type	Time stamp	Price	Volume	Type
10:23:03.332	10.12	200	Buy	10:23:03.676	10.13	200	Buy
10:23:03.443	10.13	100	Buy	10:23:03.711	10.13	100	Buy
10:23:07.121	10.14	100	Sell	10:23:10.144	10.14	200	Sell
10:23:09.574	10.15	200	Sell				

10:23:03.443. The first trade from the real portfolio does not lie in this subperiod and therefore the implementation shortfall in the first subperiod have only opportunity cost. However, it is more reasonable to conclude that the first two trades in the paper and real portfolio results in execution cost but not opportunity cost.

We can overcome this difficulty by allocating trades in real portfolio to subperiods without only considering the time stamps. However, this flexibility comes with subjectivity and the results vary among different people. In addition, when the number of trades increases, this subjective task is deemed infeasible in practice.

In essence, the problem resembles the DNA sequence alignment problem in bioinformatics. DNA sequence alignment has always been an active research topic in bioinformatics. The pioneering work of Needleman and Wunsch (1970) introduces a simple yet computationally efficient algorithm to globally align two DNA sequences. They propose a dynamic programming approach to avoid repeated calculation in the alignment of subsequences. Since then, many heuristics and variations of Needleman-Wunsch algorithm (NW algorithm) are introduced. The space complexity of NW algorithm is improved by Hirschberg (1975). Smith and Waterman (1981) introduce an algorithm for local sequence alignment algorithm. The time and space complexity of their algorithm are optimized by Gotoh (1982) and Myers and Miller (1988) respectively. Altschul and Erickson (1986) further improve the algorithm of Gotoh (1982) by using affine gap costs instead of proportional to the length of a gap.

While the methods above focused on improving the complexity of the algorithm, some variants focus on aligning two sequences with special structures. Huang and Chao (2003) modify NW algorithm for comparing sequences with intermittent similarities. Huang and Brutlag (2006) extend the work of Huang and Chao (2003) by using multiple parameter sets to compute the optimal alignment. Wallqvist *et al.* (2000) and Litvinov *et al.* (2006) take into account specific features of protein primary structures. Besides aligning two sequences, there are also extensive studies on aligning multiple sequences, see for example, Corpet (1988), Notredame, Higgins, and Heringa (2000) and Katoh *et al.* (2002).

In this paper, we propose an effective and objective two-stage framework for computing the decomposition of implementation shortfall. In the first stage, the trade records from paper portfolio and real portfolio are aligned based on sequence alignment techniques. In the second stage, the implementation shortfall is decomposed as delay cost, market impact, over-trade cost, and under-trade cost. The computational complexity of our framework is linear with respect to the number of trades of each trading sequence. Thus, our framework is applicable to high frequency trading in practice.

Review of Needleman-Wunsch Algorithm

Our trading sequence alignment algorithm is based on the NW algorithm (Needleman and Wunsch 1970), which is designed to globally align two DNA sequences effectively. It uses a dynamic programming approach to avoid repeated calculations in the alignment of subsequences. The computational complexity of the NW algorithm is $O(MN)$, where M and N are the lengths of the two DNA sequences respectively.

The NW algorithm is shown in Algorithm 1, where $\mathbf{p}$ and $\mathbf{q}$ are two DNA sequences to be aligned, p_i and q_j denote the i-th and j-th DNA of $\mathbf{p}$ and $\mathbf{q}$ respectively (e.g., $p_i = A, G, C, \text{or } T$). Matrix A is called the score matrix, $A_{i,j}$ denotes the entry of A at the i-th row and j-th column, $p_i \Theta q_j$ denotes two matched DNAs p_i and q_j being aligned (p_i and q_j are matched if $p_i = q_j$); $p_i \Phi q_j$ denotes two mismatched DNAs p_i and q_j being aligned (p_i

and q_j are mismatched if $p_i \neq q_j$), Δ denotes a missing DNA, $p_i \Theta \Delta$ denotes p_i being aligned with a missing DNA, γ is a parameter penalizing the alignment with a missing DNA, and s is a function evaluating the score of aligning two DNAs.

Since we take the maximum over the scores of the cases in the update of $A_{i,j}$, $A_{i,j}$ stores the score of the best alignment of $p_{1:i}$ with $q_{1:j}$, where $p_{i_1:i_2}$ denotes the subsequence of $\mathbf{p}$ with consecutive DNAs from p_{i_1} to p_{i_2}, and $A_{M,N}$ stores the score of the best alignment of $\mathbf{p}$ with $\mathbf{q}$. After the computation of the score matrix A is completed, we trace back from $A_{M,N}$ to $A_{0,0}$ to construct the best alignment of $\mathbf{p}$ with $\mathbf{q}$. For more details of the NW algorithm, see Needleman and Wunsch (1970).

Algorithm 1 Needleman-Wunsch algorithm

```
1: Initialize:
       $M = \text{length}(p)$, $N = \text{length}(q)$.
       $A_{0,0} = 0$.
       $A_{i,0} = A_{i-1,0} + \gamma, \quad \forall i \in 1, ..., M$.
       $A_{0,j} = A_{0,j-1} + \gamma, \quad \forall j \in 1, ..., N$.
2: for $i = 1 : M$ do
3:     for $j = 1 : N$ do
4:
$$A_{i,j} = \max \begin{cases} A_{i-1,j-1} + s(p_i, q_j), & \text{Case: } p_i \Theta q_j \text{ or } p_i \Phi q_j, \quad (1a) \\ A_{i-1,j} + \gamma, & \text{Case: } p_i \Theta \Delta, \quad (1b) \\ A_{i,j-1} + \gamma, & \text{Case: } \Delta \Theta q_j. \quad (1c) \end{cases}$$
5:     end for
6: end for
7: Trace back from $A_{M,N}$ to $A_{0,0}$ to construct the best alignment of $p$ with $q$.
```

Stage 1: Trading Sequences Alignment

In this section, we describe the alignment algorithm for aligning records from the real portfolio with records from the paper portfolio. For simplicity, we consider one financial asset[1].

1. in some cases, it is helpful to align trades of different assets. The modification is straightforward.

Description and Notation of Trading Sequences

Existing literature has investigated backtesting algorithm on different types of data. For example, Löw, Maier-Paape, and Platen (2015) and Maier-Paape and Platen (2016) investigate backtesting on candle historical data, Hurlin, Colletaz, and Tokpavi (2007) and Dionne, Duchesne, and Pacurar (2009) investigate backtesting on tick-by-tick data, Wang, Rostoker, and Wagner (2009) investigates backtesting on bid-ask tick data. Regardless of the backtest system chosen and the type of data, the trade records of a paper portfolio can be represented by $\mathbf{b}$ with b_i denoting the i-th trade record. Each trade record b_i is a vector of the form:

$$b_i = [t_i^b, p_i^b, v_i^b, y_i^b], \tag{2}$$

where t_i^b is the time when the trade is executed, p_i^b is the trading price, v_i^b is the trading volume and y_i^b is the type of the trade, i.e. buy or sell. Similarly, we use $\mathbf{l}$ to represent the trade records of real portfolio where l_j has the form:

$$l_j = [t_j^l, p_j^l, v_j^l, y_j^l]. \tag{3}$$

Trading Sequence Alignment Algorithm

Each entry in the DNA sequence is represented just by a letter and hence is one-dimensional. However a trade record is of four-dimensional, as shown in (2) and (3). In view of the dimensionality and the possible discrepancies between the two trading sequences, we develop a trading sequence alignment algorithm modified from the NW algorithm. The trading sequence alignment algorithm is presented in Algorithm 2, where A and D are two matrices storing scores of the best alignments and trading volume imbalances respectively, $b_i \Theta l_j$ denotes two matched trade records b_i and l_j being aligned (b_i and l_j are matched if $y_i^b = y_j^l$), $b_i \Phi l_j$ denotes two mismatched trade records b_i and l_j being aligned (b_i and l_j are mismatched if $y_i^b \neq y_j^l$), $b_i \Theta \Delta$ denotes b_i being aligned with a missing trade record, γ_m is a parameter penalizing a trade record being aligned with a missing trade record.

Algorithm 2 Trading sequence alignment algorithm

```
1: Initialize:
       $M = \text{length}(b)$, $N = \text{length}(l)$.
       $A_{0,0} = 0$, $D_{0,0} = 0$.
       $A_{i,0} = A_{i-1,0} + \gamma_m, D_{i,0} = 0, \quad \forall i \in 1, ..., M$.
       $A_{0,j} = A_{0,j-1} + \gamma_m, D_{0,j} = 0, \quad \forall j \in 1, ..., N$.
2: for $i = 1 : M$ do
3:     for $j = 1 : N$ do
4:
```

$$A_{i,j} = \max \begin{cases} A_{i-1,j-1} + m(b_i, l_j), & \text{Case: } b_i\Theta l_j \text{ or } b_i\Phi l_j, & (4a) \\ A_{i,j-1} + g(D_{i,j-1}, b_i, l_j), & \text{Case: } b_i\Theta\{l_{j-1}, l_j\} \text{ or } \Delta\Theta l_j, & (4b) \\ A_{i-1,j} + \gamma_m, & \text{Case: } b_i\Theta\Delta. & (4c) \end{cases}$$

$$D_{i,j} = \begin{cases} V_{b_i} - V_{l_j}, & \text{if } b_i\Theta l_j, & (5a) \\ -V_{l_j} + D_{i,j-1}, & \text{if } b_i\Theta\{l_{j-1}, l_j\}, & (5b) \\ 0, & \text{if } b_i\Phi l_j \text{ or } b_i\Theta\Delta \text{ or } \Delta\Theta l_j. & (5c) \end{cases}$$

```
5:     end for
6: end for
```

Our trading sequence alignment algorithm has a computational complexity of $O(MN)$, where M and N are the lengths of the two trading sequences respectively. So the alignment algorithm is linear with respect to the number of trades in each sequence. This makes our framework applicable to high frequency trading. In the next two subsections, we discuss in detail the update of the matrices D and A respectively.

Trading Volume Imbalance Matrix D

Trading volume imbalance matrix D stores the trading volume imbalances of the alignment from the two trading sequences. In real portfolio, one trading order could be executed as multiple trade records due to insufficient volume in one bid-ask tick. In contrast, in paper portfolio, one trading order can always be assumed executable as only one trade record. Hence, one record in paper portfolio can correspond to multiple live trading records. So our algorithm should allow one b_i being aligned with multiple l_j's. The matrix D stores the trading volume imbalances of the alignment of $\mathbf{b}$ and $\mathbf{l}$. More specifically, $D_{i,j}$ stores the trading volume imbalance of the last match of the best alignment of $b_{1:i}$ and $l_{1:j}$. In (4b), we then can compute the reward of reducing the volume imbalance using the value of $D_{i,j}$.

There are three cases in the update of $D_{i,j}$, see (5a)–(5c). The update of $D_{i,j}$ depends on which case we have in the update of $A_{i,j}$. We explain the cases separately in the following:

1. For (5a), two matched trade records b_i and l_j are aligned. If $D_{i,j} > 0$ then there are more trading volume in b_i than l_j, and vice versa. If $D_{i,j} = 0$, then there is no trading volume imbalance.

2. For (5b), one live trading record l_j is added to the alignment containing b_i and l_{j-1}. Since the trading volume imbalance in the alignment of b_i and l_{j-1} is already stored in $D_{i,j-1}$, we calculate $D_{i,j}$ by subtracting the trading volume of l_j from $D_{i,j-1}$.

3. For (5c), two mismatched trade records b_i and l_j are aligned or a trade record is aligned with a missing trade record. In either case, there is no trading volume imbalance, so we set $D_{i,j}$ to be 0.

Score Matrix A

Score matrix A stores the scores of the best alignment of the two trading sequences. There are 3 cases in the update of $A_{i,j}$, see (4a)–(4c). The maximum score among the three cases is used to update $A_{i,j}$. We explain the cases separately in the following:

1. For (4a), two trade records b_i and l_j are aligned. If the two trade records are matched, we assign a score with its magnitude depending on the similarity between the two trade records; if they are mismatched, we assign a penalty. The following function $m(b_i, l_j)$

is used to evaluate the score:

$$m(b_i, l_j) = \begin{cases} 1 - \lambda_t |t_i^b - t_j^l| - \lambda_p \dfrac{|p_i^b - p_j^l|}{p_i^b + p_j^l} - \lambda_v \dfrac{|v_i^b - v_j^l|}{|v_i^b + v_j^l|}, & \text{if } y_i^b = y_j^l, \quad (6a) \\ -\infty, & \text{if } y_i^b \neq y_j^l, \quad (6b) \end{cases}$$

where λ_t, λ_p and λ_v are parameters of the weights on time, price, and volume respectively. Recall that the y_i, t_i, p_i, and v_i are defined in (2) and (3). Since it is irrational to align two mismatched trade records, the output of m is $-\infty$ if the two trade records are mismatched, i.e. $y_i^b \neq y_j^l$. Hence two mismatched records will not be aligned. If b_i and l_j are matched, $m(b_i, l_j)$ will return a value that depends on how similar the trade records b_i and l_j are, i.e. the more similar they are, the larger the $m(b_i, l_j)$.

2. In case (4b), the j-th live trading record l_j is added to the alignment of b_i with l_{j-1}, or l_j is aligned with a missing trade record. A function g is used to evaluate the score in such case and the function is given by the following:

$$g(D_{i,j-1}, b_i, l_j) = \begin{cases} h(D_{i,j-1}, b_i, l_j) & \text{if } D_{i,j-1} > 0 \text{ and } y_i^b = y_j^l, \quad (7a) \\ \gamma_m, & \text{if } D_{i,j-1} \leq 0 \text{ or } y_i^b \neq y_j^l, \quad (7b) \end{cases}$$

where the function h is given by:

$$h(D_{i,j-1}, b_i, l_j) = \max \begin{cases} -\gamma_t |t_i^b - t_j^l| - \gamma_p \dfrac{|p_i^b - p_j^l|}{p_i^b + p_j^l} + \gamma_v f(D_{i,j-1}, v_j^l), & (8a) \\ \gamma_m, & (8b) \end{cases}$$

with

$$f(D_{i,j-1}, v_j^l) = \begin{cases} \dfrac{v_j^l - D_{i,j-1}}{D_{i,j-1}}, & \text{if } v_j^l > D_{i,j-1}, \quad (9a) \\ 0, & \text{if } v_j^l \leq D_{i,j-1}, \quad (9b) \end{cases}$$

where γ_t, γ_p and γ_v are the weights on time, price, and volume imbalance respectively. We recall γ_m is the weight of alignment with a missing trade record.

For (7a), it is under the condition that $D_{i,j-1} > 0$, i.e. there are more backtesting volumes than live trading volumes in the last alignment. When this happens, we would like to align l_j with the alignment in order to reduce the volume imbalance if the backtesting trade record and the live trading record are not too dissimilar. Hence for (8a), we evaluate the time and price similarity between b_i and l_j and we use the function f in (9) to evaluate the reward of reducing the order imbalance. For (9a), the volume imbalance will be overcompensated as $v^l_j > D_{i,j-1}$. We then assign a penalty which is the ratio of the overcompensation to the volume imbalance. For (9b), if the volume v^l_j is less than or equal to the volume imbalance $D_{i,j-1}$, we do not put any penalty to encourage reducing the volume imbalance. However, if l_j and b_i are too dissimilar in the sense that the score is lower than l_j being aligned with a missing trade record, we will align l_j with a missing trade record instead. After that, we then take the maximum over the score of (8a), which is the score of aligning l_j with the last alignment, and the score of (8b), which is the score of aligning l_j with a missing trade record. Note that we can keep adding live trading records to the last alignment and eventually have an alignment with two or more l_j's.

The case (7b) is when the live trading volume is not larger than the backtesting volume or when the type of trading does not match, i.e. $y^b_i \neq y^l_j$. In such a case, the function g will just output the score of l_j being aligned with a missing trade record.

3. For (4c), the i-th trade record b_i from paper portfolio is aligned with a missing trade record. Hence, a score of γ_m penalizing this alignment is assigned. Recall that we assumed a trading order can always be executed as only one trading record in paper portfolio, hence there is no multiple backtesting trade records alignment.

Stage 2: Breakdown of Implementation Shortfall

Given the alignment results obtained in the first stage, one can break down the implementation shortfall into its execution cost and opportunity cost. The breakdown is similar to the implementation shortfall breakdown approach proposed in Perold (1988). Yet our proposed method first aligns the two sequences with multiple transactions on both live trading and backtesting while Perold's approach assumes the trading period lies between two transactions of the backtesting sequence.

In this section, we present the derivation of the implementation shortfall breakdown. After the first stage, we obtain sets of aligned trade records where each set of aligned trade records contains aligned backtesting trade records and live trading records. Note that each set of aligned trade records can contain multiple live trading records but only one backtesting trade record. Let $P^l_{i,j}$ and $V^l_{i,j}$ be the price and volume of the j-th trade record of the i-th aligned set of trade records in live trading respectively, P^b_i and V^b_i be the price and volume of the i-th set of aligned trade records in backtesting respectively, N be the total number of sets of aligned trade records, and M_i be the number of live trading records in the i-th set of aligned trade records.

The implementation shortfall S can be expressed as the market-to-market profit and loss difference between real and paper portfolios:

$$S = \sum_{i=1}^{N+1} \Big(\sum_{j=1}^{M_i} P^l_{i,j} V^l_{i,j} - P^b_i V^b_i \Big). \tag{10}$$

Here, for simplicity, we assume all positions are unwound in one trade record at the end of the evaluation period for both portfolios, hence we set $M_{N+1} = 1$. In particular, $P^l_{N+1,1}$ and P^b_{N+1} are both equal to the asset price at the end of the period; and $V^l_{N+1,1}$ and V^b_{N+1} are equal to the trading volumes needed to entirely unwind the positions in the live trading portfolio and backtesting portfolio respectively at the end of the period, i.e. $V^l_{N+1,1} = -\sum_{i=1}^{N} \sum_{j=1}^{M_i} V^l_{i,j}$ and $V^b_{N+1} = -\sum_{i=1}^{N} V^b_i$.

We remark that in (10), the volumes are set to 0 for all missing trade records, and their price are set to the corresponding price in the aligned trade records. Applying some rearrangements to R.H.S. of (10), we obtain:

$$\begin{aligned} S &= \sum_{i=1}^{N+1} \left(\sum_{j=1}^{M_i} P_{i,j}^l V_{i,j}^l - \sum_{j=1}^{M_i} P_i^b V_{i,j}^l + \sum_{j=1}^{M_i} P_i^b V_{i,j}^l - P_i^b V_i^b \right) \\ &= \sum_{i=1}^{N+1} \sum_{j=1}^{M_i} \left(P_{i,j}^l - P_i^b \right) V_{i,j}^l + \sum_{i=1}^{N+1} P_i^b \left(\sum_{j=1}^{M_i} V_{i,j}^l - V_i^b \right). \end{aligned} \tag{11}$$

Putting $V_{N+1,1}^l = -\sum_{i=1}^{N} \sum_{j=1}^{M_i} V_{i,j}^l$, $V_{N+1}^b = -\sum_{i=1}^{N} V_i^b$, and $P_{N+1,1}^l = P_{N+1}^b$ into R.H.S. of (11), then applying some simplifications and rearrangements, we have:

$$S = \underbrace{\sum_{i=1}^{N} \sum_{j=1}^{M_i} \left(P_{i,j}^l - P_i^b \right) V_{i,j}^l}_{\text{Execution cost}} + \underbrace{\sum_{i=1}^{N} \left(P_i^b - P_{N+1}^b \right) \left(\sum_{j=1}^{M_i} V_{i,j}^l - V_i^b \right)}_{\text{Opportunity cost}}. \tag{12}$$

The breakdown is interpreted as follows. In the first term, $(P_{i,j}^l - P_i^b)$ is the cost of trading at the price of $P_{i,j}^l$ instead of P_i^b, and $V_{i,j}^l$ is the trading volume in which this cost is involved. In the second term, $(\sum_{j=1}^{M_i} V_{i,j}^l - V_i^b)$ is the volume of an unexecuted trade and $(P_i^b - P_{N+1}^b)$ is the net return of the unexecuted trade.

The execution cost can be further broken down to delay cost, which is the cost caused by delayed execution, and market impact. The opportunity cost can be further broken down to over-trade cost and under-trade cost. The breakdown is given as follows:

$$\begin{aligned} S =& \underbrace{\sum_{i=1}^{N} \left(P_{i,1}^l - P_i^b \right) V_{i,1}^l}_{\text{Delay cost}} + \underbrace{\sum_{i=1}^{N} \sum_{j=2}^{M_i} \left(P_{i,j}^l - P_i^b \right) V_{i,j}^l}_{\text{Market impact}} \\ &+ \underbrace{\sum_{i=1}^{N} \left(P_i^b - P_{N+1}^b \right) \left(\sum_{j=1}^{M_i} V_{i,j}^l - V_i^b \right) \mathbb{1}_{|\sum V_{i,j}^l| > |V_i^b|}}_{\text{Over-trade cost}} + \underbrace{\sum_{i=1}^{N} \left(P_i^b - P_{N+1}^b \right) \left(\sum_{j=1}^{M_i} V_{i,j}^l - V_i^b \right) \mathbb{1}_{|\sum V_{i,j}^l| < |V_i^b|}}_{\text{Under-trade cost}}. \end{aligned} \tag{13}$$

Here $\mathbb{1}_{|\sum V^l_{i,j}|>|V^b_i|}$ denotes an indicator function, i.e. $\mathbb{1}_{|\sum V^l_{i,j}|>|V^b_i|} = 1$ if $|\sum V^l_{i,j}| > |V^b_i|$ and $\mathbb{1}_{|\sum V^l_{i,j}|>|V^b_i|} = 0$ if $|\sum V^l_{i,j}| \leq |V^b_i|$. The cost caused by the first trade record of each aligned set of trade record in live trading will be considered as delay cost. This is in the same spirit as in Hendershott, Jones, and Menkveld (2013), i.e. the first trade record is not affected by market impact, thus any cost is attributed to time delay. After the first trade is executed, the subsequent trades will be affected by the market impact caused by the first trade. Hence, the trades after the first trade are attributed to market impact.

It is also possible to compute the implementation shortfall breakdown for each set of aligned trade records. It is done by simply extracting the i-th term in (13). Let S_i be the implementation shortfall for the i-th set of aligned trade records. Thus S_i can be expressed as:

$$\begin{aligned} S_i = &\underbrace{\left(P^l_{i,1} - P^b_i\right) V^l_{i,1}}_{\text{Delay cost}} + \underbrace{\sum_{j=2}^{M_i} \left(P^l_{i,j} - P^b_i\right) V^l_{i,j}}_{\text{Market impact}} \\ &+ \underbrace{\left(P^b_i - P^b_{N+1}\right)\left(\sum_{j=1}^{M_i} V^l_{i,j} - V^b_i\right) \mathbb{1}_{|\sum V^l_{i,j}|>|V^b_i|}}_{\text{Over-trade cost}} + \underbrace{\left(P^b_i - P^b_{N+1}\right)\left(\sum_{j=1}^{M_i} V^l_{i,j} - V^b_i\right) \mathbb{1}_{|\sum V^l_{i,j}|<|V^b_i|}}_{\text{Under-trade cost}} . \end{aligned} \tag{14}$$

Diagnosis of Algorithmic Trading Implementation

One purpose of computing the implementation shortfall is to improve the existing algorithmic trading systems. Large delay cost reflects inefficiency in software or hardware throughout the whole algorithmic trading implementation from market data feed to order execution. Market impact cost can be reduced by employing optimal execution strategies, see for example Bertsimas and Lo (1998).

If the opportunity cost is large, it is likely due to incorrect implementation of the trading system in a live trading environment. High under-trade cost could be due to using wrong type of trades (e.g., fill-or-kill, fill-and-cancel, etc.) during order execution. Over-trade should be rare in algorithmic trading implementation. High over-trade cost should give attention to potential human intervention during live trading. Our framework provides a valuable tool for continuous monitoring of algorithmic trading strategies. For more detailed discussions on diagnosis of algorithmic trading implementation, see Harris (2008, Chapter 6), Chan (2009, Chapter 3) and Pardo (2008, Chapter 6).

Experimental Results

In this section, we apply our alignment algorithm on illustrative live trading and backtesting data and show how one can calculate the implementation shortfall based on the results of our alignment algorithm. Exhibit 4 shows the trading sequences generated by backtesting and live trading respectively. Exhibit 5 shows an alignment result generated by our algorithm, where the end of evaluation period price is set to be 28390, i.e., $P^l_{N+1,1} = P^b_{N+1} = 28390$, and parameters $\lambda_t = 1$, $\lambda_p = 2$, $\lambda_v = 1$, $\gamma_t = 1$, $\gamma_p = 2$, $\gamma_v = 1$ and $\gamma_m = 1$. Exhibit 6 shows another alignment result, where the parameters are the same as in Exhibit 5 but $\gamma_t = 3$, i.e. we put more penalty on the time difference between trade records for multiple trade records alignment. With more penalty on the time difference, we obtain different alignment results (marked by bold-face fonts) and hence produce a different implementation shortfall breakdown. Each row of Exhibit 5 and Exhibit 6 contains one set of aligned trade records.

Conclusion

We blend an efficient algorithm in bioinformatics and a classic implementation shortfall to create a useful tool for evaluating algorithmic trading implementation. The framework is applicable to high frequency trading since the complexity only grows linearly with respect to the number of trades in each trading sequence. Moreover, it is simple to implement and it provides an objective way to analyze implementation shortfall on algorithmic trading. The breakdown of costs resulted from the framework also gives astute insights on the algorithmic

Exhibit 4: Live trading and backtesting results

Backtesting				Live trading			
Time Stamp	Price	Volume	Type	Time Stamp	Price	Volume	Type
10:00:00.122	28387	1	Buy	10:00:03.008	28387.5	2	Buy
10:00:02.811	28387	2	Buy	10:00:04.154	28389	−2	Sell
10:00:04.115	28389	−4	Sell	10:00:04.417	28387	−2	Sell
10:00:11.899	28394	5	Buy	10:00:08.593	28390	−1	Sell
10:00:13.563	28396	−1	Sell	10:00:12.018	28394	2	Buy
10:00:15.552	28400	7	Buy	10:00:12.080	28395	2	Buy
10:00:16.237	28398	4	Buy	10:00:14.026	28393	−1	Sell
				10:00:15.833	28402	5	Buy
				10:00:16.005	28405	3	Buy
				10:00:16.737	28400	2	Buy

trading implementation.

References

Altschul, S. F., and B. W. Erickson. 1986. “Optimal sequence alignment using affine gap costs.” *Bulletin of mathematical biology* 48 (5-6): 603–616.

Bailey, D. H., J. Borwein, M. López de Prado, and Q. J. Zhu. 2016. “The probability of backtest overfitting.” *Journal of Computational Finance* 20 (4): 39–69.

Bertsimas, D., and A. Lo. 1998. “Optimal control of execution costs.” *Journal of Financial Markets* 1 (1): 1–50.

Chan, E. 2009. *Quantitative trading: how to build your own algorithmic trading business.* Hoboken, N.J.: John Wiley.

Corpet, F. 1988. “Multiple sequence alignment with hierarchical clustering.” *Nucleic acids research* 16 (22): 10881–10890.

Exhibit 5: Alignment and implementation shortfall breakdown on the paper trading and real trading data with $\mu_t = 1$.

	Paper trading				Real trading				Delay Cost	Market Impact	Over-trade Cost	Under-trade Cost
	Time Stamp	Price	Volume	Type	Time Stamp	Price	Volume	Type				
1	10:00:00.122	$P_1^b = 28387$	$V_1^b = 1$	Buy		$P_{1,1}^r = 28387$	$V_{1,1}^r = 0$		0	0	0	3
2	10:00:02.811	$P_2^b = 28387$	$V_2^b = 2$	Buy	10:00:03.008	$P_{2,1}^r = 28387.5$	$V_{2,1}^r = 2$	Buy	1	0	0	0
3	10:00:04.115	$P_3^b = 28389$	$V_3^b = -4$	Sell	10:00:04.154	$P_{3,1}^r = 28389$	$V_{3,1}^r = -2$	Sell	0	4	0	0
					10:00:04.417	$P_{3,2}^r = 28387$	$V_{3,2}^r = -2$	Sell				
4		$P_4^b = 28390$	$V_4^b = 0$		10:00:08.593	$P_{4,1}^r = 28390$	$V_{4,1}^r = -1$	Sell	0	0	0	0
5	10:00:11.899	$P_5^b = 28394$	$V_5^b = 5$	Buy	10:00:12.018	$P_{5,1}^r = 28394$	$V_{5,1}^r = 2$	Buy	0	2	0	-4
					10:00:12.080	$P_{5,2}^r = 28395$	$V_{5,2}^r = 2$	Buy				
6	10:00:13.563	$P_6^b = 28396$	$V_6^b = -1$	Sell	10:00:14.026	$P_{6,1}^r = 28393$	$V_{6,1}^r = -1$	Sell	3	0	0	0
7	10:00:15.552	$P_7^b = 28400$	$V_7^b = 7$	Buy	10:00:15.833	$P_{7,1}^r = 28402$	$V_{7,1}^r = 5$	Buy	10	15	10	0
					10:00:16.005	$P_{7,2}^r = 28405$	$V_{7,2}^r = 3$	Buy				
8	10:00:16.237	$P_8^b = 28398$	$V_8^b = 4$	Buy	10:00:16.737	$P_{8,1}^r = 28400$	$V_{8,1}^r = 2$	Buy	4	0	0	-16
	Total profit and loss = -111				Total profit and loss = -143				Total= 18	Total= 21	Total= 10	Total= -17

Exhibit 6: Alignment and implementation shortfall breakdown on the paper trading and real trading data with $\mu_t = 3$.

	Paper trading				Real trading				Delay Cost	Market Impact	Over-trade Cost	Under-trade Cost
	Time Stamp	Price	Volume	Type	Time Stamp	Price	Volume	Type				
1	10:00:00.122	$P_1^b = 28387$	$V_1^b = 1$	Buy		$P_{1,1}^r = 28387$	$V_{1,1}^r = 0$		0	0	0	3
2	10:00:02.811	$P_2^b = 28387$	$V_2^b = 2$	Buy	10:00:03.008	$P_{2,1}^r = 28387.5$	$V_{2,1}^r = 2$	Buy	1	0	0	0
3	10:00:04.115	$P_3^b = 28389$	$V_3^b = -4$	Sell	10:00:04.154	$P_{3,1}^r = 28389$	$V_{3,1}^r = -2$	Sell	0	4	0	0
					10:00:04.417	$P_{3,2}^r = 28387$	$V_{3,2}^r = -2$	Sell				
4		$P_4^b = 28390$	$V_4^b = 0$		10:00:08.593	$P_{4,1}^r = 28390$	$V_{4,1}^r = -1$	Sell	0	0	0	0
5	10:00:11.899	$P_5^b = 28394$	$V_5^b = 5$	Buy	10:00:12.018	$P_{5,1}^r = 28394$	$V_{5,1}^r = 2$	Buy	0	2	0	−4
					10:00:12.080	$P_{5,2}^r = 28395$	$V_{5,2}^r = 2$	Buy				
6	10:00:13.563	$P_6^b = 28396$	$V_6^b = -1$	Sell	10:00:14.026	$P_{6,1}^r = 28393$	$V_{6,1}^r = -1$	Sell	3	0	0	0
7	**10:00:15.552**	$\mathbf{P_7^b = 28400}$	$\mathbf{V_7^b = 7}$	**Buy**	**10:00:15.833**	$\mathbf{P_{7,1}^r = 28402}$	$\mathbf{V_{7,1}^r = 5}$	**Buy**	10	0	0	−20
8	**10:00:16.237**	$\mathbf{P_8^b = 28398}$	$\mathbf{V_8^b = 4}$	**Buy**	**10:00:16.005**	$\mathbf{P_{8,1}^r = 28405}$	$\mathbf{V_{8,1}^r = 3}$	**Buy**	21	0	0	−8
9		$\mathbf{P_9^b = 28400}$	$\mathbf{V_9^b = 0}$		**10:00:16.737**	$\mathbf{P_{9,1}^r = 28400}$	$\mathbf{V_{9,1}^r = 2}$	**Buy**	0	0	20	0
	Total profit and loss = −111				Total profit and loss = −143				Total= 35	Total= 6	Total= 20	Total=−29

Dionne, G., P. Duchesne, and M. Pacurar. 2009. "Intraday Value at Risk (IVaR) using tick-by-tick data with application to the Toronto Stock Exchange." *Journal of Empirical Finance* 16 (5): 777–792.

Gotoh, O. 1982. "An improved algorithm for matching biological sequences." *Journal of molecular biology* 162 (3): 705–708.

Harris, M. 2008. *Profitability and Systematic Trading: A Quantitative Approach to Profitability, Risk, and Money Management.* Hoboken, N.J.: John Wiley.

Hendershott, T., C. M. Jones, and A. J. Menkveld. 2013. "Implementation shortfall with transitory price effects." In *High Frequency Trading: New Realities for Trades, Markets, and Regulators,* edited by D. Easley, M. López de Prado, and M. OfffdfffdfffdHara, 185fffdfffdfffd–206. Risk Books.

Hirschberg, D. S. 1975. "A linear space algorithm for computing maximal common subsequences." *Communications of the ACM* 18 (6): 341–343.

Huang, X., and D. L. Brutlag. 2006. "Dynamic use of multiple parameter sets in sequence alignment." *Nucleic Acids Research* 35 (2): 678–686.

Huang, X., and K.-M. Chao. 2003. "A generalized global alignment algorithm." *Bioinformatics* 19 (2): 228–233.

Hurlin, C., G. Colletaz, and S. Tokpavi. 2007. *Irregularly Spaced Intraday Value at Risk (ISIVaR) Models: Forecasting and Predictive Abilities.* Technical report. Hyper Articles en Ligne.

Katoh, K., K. Misawa, K.-i. Kuma, and T. Miyata. 2002. "MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform." *Nucleic acids research* 30 (14): 3059–3066.

Kissell, R. 2006. “The expanded implementation shortfall.” *Journal of Trading* 1 (3): 6–16.

Litvinov, I., M. Y. Lobanov, A. Mironov, A. Finkelshtein, and M. Roytberg. 2006. “Information on the secondary structure improves the quality of protein sequence alignment.” *Molecular Biology* 40 (3): 474–480.

Löw, R., S. Maier-Paape, and A. Platen. 2015. “Correctness of Backtest Engines.” *arXiv preprint arXiv:1509.08248.*

Maier-Paape, S., and A. Platen. 2016. “Backtest of Trading Systems on Candle Charts.” *A Professional Journal Published by The International Federation of Technical Analysts* 16:10–17.

Myers, E. W., and W. Miller. 1988. “Optimal alignments in linear space.” *Bioinformatics* 4 (1): 11–17.

Needleman, S. B., and C. D. Wunsch. 1970. “A general method applicable to the search for similarities in the amino acid sequence of two proteins.” *Journal of molecular biology* 48 (3): 443–453.

Notredame, C., D. G. Higgins, and J. Heringa. 2000. “T-Coffee: A novel method for fast and accurate multiple sequence alignment.” *Journal of molecular biology* 302 (1): 205–217.

Pardo, R. 2008. *The evaluation and optimization of trading strategies.* Hoboken, N.J.: John Wiley.

Perold, A. F. 1988. “The implementation shortfall: Paper versus reality.” *The Journal of Portfolio Management* 14 (3): 4–9.

Smith, T. F., and M. S. Waterman. 1981. “Identification of common molecular subsequences.” *Journal of molecular biology* 147 (1): 195–197.

Wallqvist, A., Y. Fukunishi, L. R. Murphy, A. Fadel, and R. M. Levy. 2000. “Iterative sequence/secondary structure search for protein homologs: comparison with amino acid sequence alignments and application to fold recognition in genome databases.” *Bioinformatics* 16 (11): 988–1002.

Wang, J., C. Rostoker, and A. Wagner. 2009. “A high performance pair trading application.” In *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on,* 1–8. IEEE.