On A/B Testing and the Benefits of Knowing Why

Allison Bishop Sam Markelon Lisa Oakley Adele Shahi

Proof Trading

1 Introduction

There is a compelling argument in trading that as long as something works, you don't need to know *why* it works. In particular, if an algo is producing good trading outcomes, there is no immediate incentive to pick it apart, interrogate its design, and fully explain its operation. There is an intuitive pragmatism in this. Autopsies are for the dead, not the living.

But alongside this human willingness to ignore details as long as things go in your favor, there is a contradictory and seemingly irresistible impulse to meddle. What if we tweaked our routing strategy? What if we tried waiting a bit longer (or a bit shorter) for our posted orders to trade before resorting to taking? Ideas for algo improvements tend to be easy to come up with. The hard part is evaluating which ones actually help.

Almost no idea is going to help performance *universally*. For any change we make, it's probably going to lead to a better outcome some of the time and a worse outcome some of the time. As an algo provider, our goal then is to determine the net effect on our typical trading flow, to see if a potential change is net positive or net negative. The answer here may be highly variable and dependent, and may change as market conditions change and/or the nature of typical flow changes in our client base.

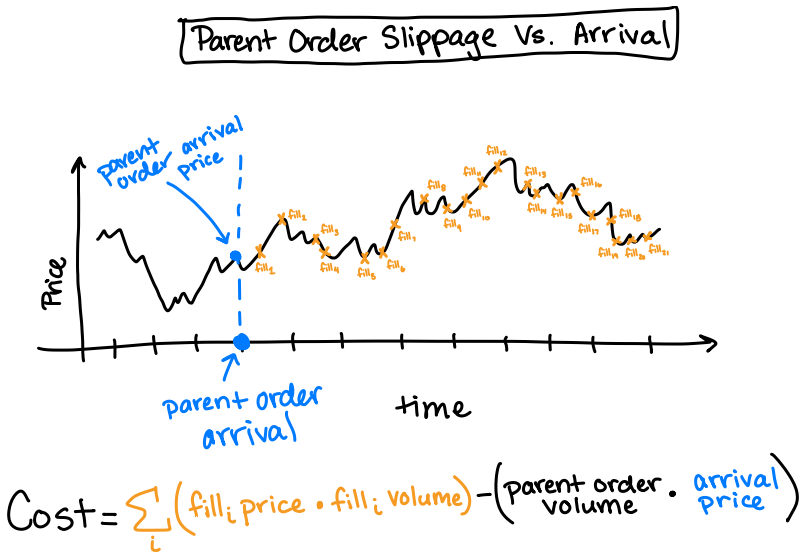
A common approach to estimating the net effect of a particular algo tweak is to run an "A/B test." This involves pitting two versions of the algo head-to-head, randomly assigning incoming orders to be handled by one or the other, and then measuring the overall outcomes. Declaring a winner of an A/B test feels like a natural and obvious procedure: whichever algo produced the better average performance should be the winner!

There is a lot of nuance, however, lurking in the phrase "better average performance." Let's break it down word by word. We'll start with the word "better." How much better should "better" be? Even if the algos were exactly the same, their performance numbers on different orders would be slightly different, and one of them would be "better." Adopting winning results uncritically when there is no meaningful advantage could lead to algo whiplash (frequently introducing and rolling back features upon re-testing) and algo bloat (adding complexity to the code and operations that is not justified by performance). It may also give us a false sense of progress that we are "improving" performance when really we are just treading water. For this reason, we need a way of assessing how much "better" a performance needs to be in order to represent strong evidence that one algo version meaningfully outperforms another. Conversely, we need to ensure that our A/B testing procedure can produce a nonchalant shrug as an answer when it's called for.

Next let's tackle the word "average." We might assess "average" performance by taking a literal average over orders, perhaps weighted by notional value, by volume, or some other relevant parameter. This may directly represent what we most care about (e.g. our bottomline performance in dollars), but it can also be swung considerably by outliers and noise. A median is more robust, but may fail to capture performance differences in the tails. Ideally, our analyses of A/B testing

results should equip us with a more holistic understanding of the distributions of performance that we can expect from each version of the algo, but this is not something that can come from collapsing performance to a single number in each case.

Finally, let's consider the word "performance." There are a few common choices of benchmarks that we can use to define performance, such as "slippage vs. arrival" which compares the achieved trade prices to the prevailing public price of the symbol at the time the order initiated. This metric, however, is notoriously noisy, especially for orders that trade over relatively long time horizons, as unrelated market forces seep in to influence prices in addition to the potential impact of the order's trading activity.

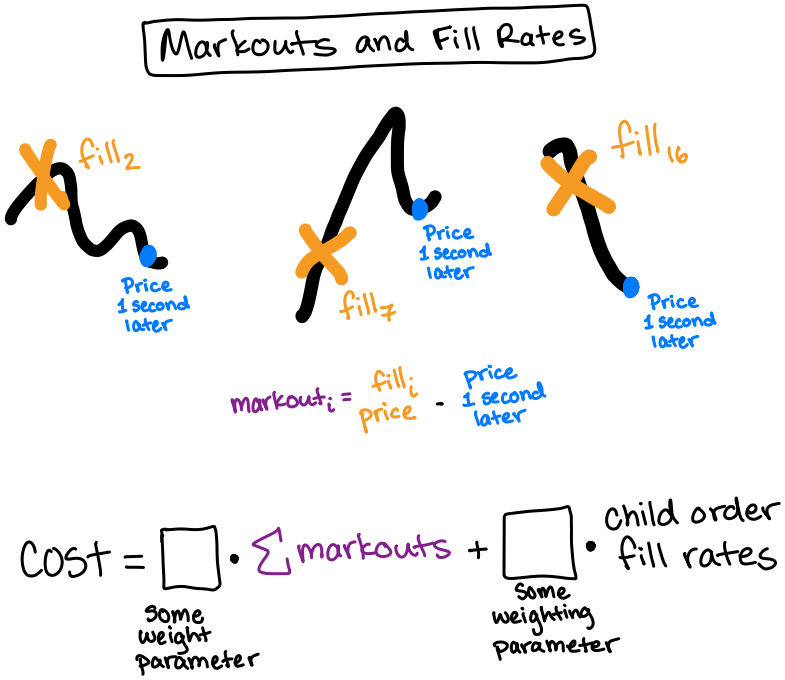


A benchmark like the volume-weighted average price (vwap) of trades in the same symbol across the wider market throughout the order's lifetime can be less noisy, as it accounts for such contemporaneous market forces in the benchmark itself. But this also creates a blind spot, as any impact that the order has on the benchmark gets canceled out by the comparison. In other work, we have investigated ways to remove some extraneous noise from the arrival price benchmark without introducing as much circularity as the vwap benchmark [2, 3]. While such methods can meaningfully reduce noise without the full downsides of vwap, a lot of noise remains.

Given these challenges in assessing parent order behavior, it is tempting to gravitate towards metrics that operate on shorter timescales. For example, we might look at the short-term markouts achieved on fills for algo version A versus algo version B. Such measurements are inherently less noisy, as there is less time for extraneous market forces to obscure the impacts of our design choices, and we have a naturally larger sample size of individual fills as compared to parent orders. However, metrics like short-term markouts alone can be poor proxies for the parent-level performance we care about. For one thing, they fail to capture tradeoffs that play out over greater timescales. Algo A might achieve better markouts on average, for example, by trading more passively throughout the day and then executing more in the closing auction, but this might be bad for parent order performance overall if it is substantially impacting the close. (Even if it is not substantially impacting the close, it might be increasing variance in the parent order performance distribution without any meaningful benefit in expectation).

A natural approach for addressing such issues is to measure multiple short-term quantities, like markouts as well as fill rates, and pick some weighting parameter that is intended to balance the tradeoff between them. This just moves the design challenge onto the selection of that parameter though, which is often somewhat arbitrary. Especially if this parameter is not committed to ahead

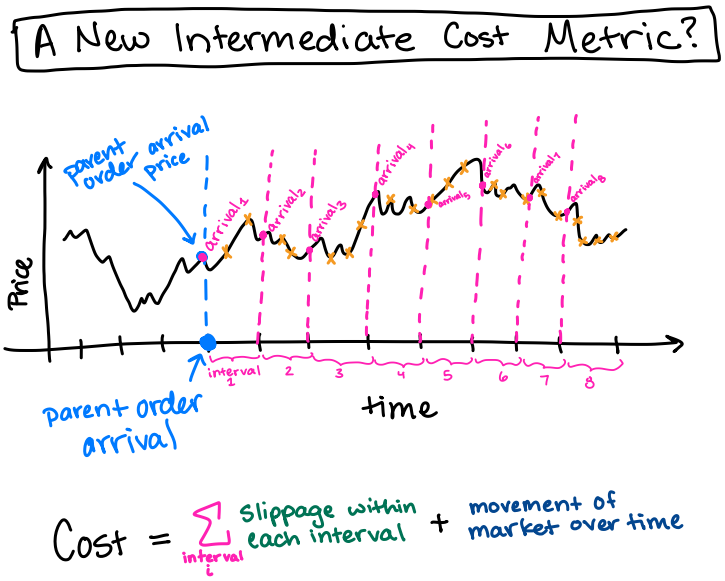
of time, it can end up being a vehicle for the data analyst (consciously or unconsciously) to steer the outcome to whichever version of the algo they prefer.



So what are we to do? Live with a parent level analysis that will require a lot of data and a lot of luck to provide robust and meaningful results? Or live with a lower level analysis that will require some arbitrary choices when we are confronted with inevitable tradeoffs?

As you might guess from the fact that there are many more pages in this paper, we choose neither. Instead, we are going to develop an approach that build models of algo behavior at shorter timescales (where there is less noise) and then extrapolates those models to expected parent level effects (where important tradeoffs are inherently captured). The hope is that we can combine the clarity of shorter timescale effects with the comprehensibility of parent-level calculations. But there is a catch - we need to introduce some fairly strong assumptions in our extrapolation between the differing timescales.

What results is a framework for transferring insights about trading behavior at a shorter timescale (e.g. seconds, minutes) to trading behavior at a longer timescale (e.g. hours, days), in a way that will more quickly produce an accurate understanding of the longer time scale, as compared to simply measuring the longer time only, *assuming that our strong extrapolation assumptions are true.*



Such a framework is ultimately useful beyond A/B testing, and is in fact a generalization of an approach that is already embedded in some of our algo designs. It also has the potential to be used in TCA, even when there is not a precisely designed A/B test in progress. Such implications we will flesh out later, once we have explained and demonstrated how the framework functions and why it is expected to produce sharper results than a pure parent level metric under certain conditions.

Stepping back, the framework we present here is going to emerge as a natural deconstruction of a child-like investment in the question of "why?" For example:

Q: Why do we think algo A is better than algo B?

A: Because our model of its expected slippage vs. arrival is lower for all common ranges of parent order parameters

Q: but why?

Because the model indicates that it achieves less price impact over short term scales with a slight decrease in fill rate, which ultimately nets out to better overall performance.

Q: but why?

A: Because we designed algo A to post passively more before crossing the spread.

Q: but why?

A: Because we hoped it would result in better parent order performance. Now go brush your teeth and read this TCA report, it will put you right to sleep.

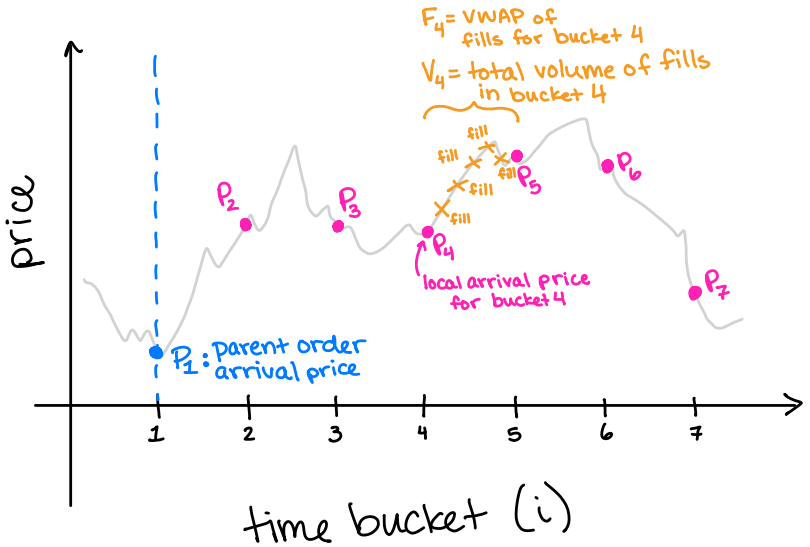
The algo design and evaluation process is basically this just in reverse - we have an idea for a different behavior at a low level timescale (seconds, minutes) that will probably sometimes yield better results and sometimes yield worse results. We can then build a model of behavior at the relevant timescale (second, minutes) and extrapolate this out to project overall results under various assumptions and inputs. We can then evaluate whether the "sometimes" that the new idea is better outweigh the "sometimes" that the new idea is worse, by averaging our extrapolations over an approximation of the distribution of inputs that we expect to encounter in practice. We can do this granularly for different portions of our trading flow, which enables us to ultimately make nuanced design decisions, like changing to the new behavior only when certain conditions are met. The low level modeling step here gives us insight into when/how/why our idea works at this timescale, while the extrapolation step keeps us honest in accounting for tradeoffs at the greater parent order timescale (at the cost of introducing extrapolation assumptions).

Executing this framework for real A/B tests and design processes requires identifying reasonable assumptions about how the market as a stochastic process incorporates and absorbs the influence of trading behaviors at a specified timescale. Such assumptions will lead us to models of shorter term effects and computations of extrapolation. In the rest of this paper, we'll develop an example that we have used for evaluating an A/B test on our "worker" logic, which is the layer of our algo design that handles trading tactics over stretches of a several minutes at a time. The framework will not specifically depend on *what* two tactics are being tested, but is rather general. We will also prove that under our assumptions, this framework *reduces* the noise in our results relative to comparing parent order slippage vs. arrival (even including normalizing adjustments for general market trends, etc.)

2 Modeling Intermediate Timescales and Their Accumulating Effects

We begin our framework by selecting an intermediary timescale: long enough to span the kind of differences in algo tactics that we are interested in, short enough to avoid unnecessary noise. As a running example, let's imagine that we pick 10 minute time buckets as an intermediary scale. We'll then divide the regular trading day into such buckets.

As a basis for understanding evolution of prices across time buckets, we'll sample reference prices (e.g. the NBBO midpoint) at the boundary of each bucket. Formally, we'll define a random variable P_i that denotes the reference price at the beginning of bucket i. We are also interested in the performance of our fills obtained within each bucket, so we will define a random variable F_i to denote the volume-weighted average price for fills we obtain in bucket i, and a random variable V_i to denote the volume that we trade in bucket i. (As a consequence, note that the product $F_i V_i$ corresponds to the notional value we traded in bucket i.)



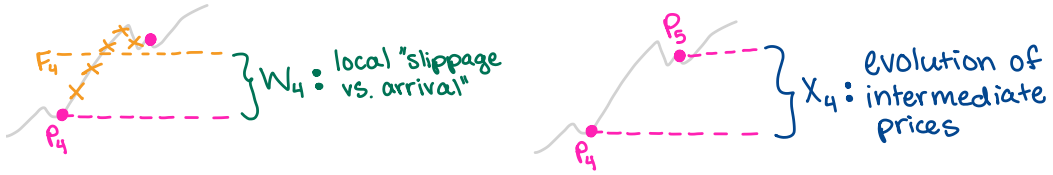
We are interested in assessing how algo tactics affect trading outcomes at a level that is local to our intermediary timescale. For this we will define the random variable W_i, which represents the difference between our average fill price and the local "arrival" price for the bucket:

$$W_i = F_i - P_i. \tag{1}$$

Looking only at W_i values, however, will leave us blind to any reverberating effects of our behavior in bucket i on the next reference price, P_{i+1}. For example, imagine that we trade very aggressively at the beginning of interval i, buying a lot of shares at prices near P_i. The market may then react and set P_{i+1} higher, a reaction that will hurt our overall performance when we have more to trade, but the variables W_{i+1} and onward do not account for this. We therefore introduce another random variable X_i which represents the evolution of the reference prices:

$$P_i = P_{i-1} + X_{i-1}. \tag{2}$$

We now have all the ingredients to express how effects at the bucket time scale accumulate over the lifetime of an order.



We can now break down the total cost of a buy order in terms of these arrival prices, P_i, and the local costs of our fills, W_i, to express the total cost of a buy order as

$$\sum_{i=1}^{t} V_i F_i = \sum_{i=1}^{t} V_i \left(W_i + P_i\right). \tag{3}$$

Parent-level slippage versus arrival in dollars (for buy orders) can then be calculated by subtracting P_1 times the total volume from this to obtain:

$$\sum_{i=1}^{t} V_i \left(W_i + P_i\right) - \sum_{i=1}^{t} V_i P_1 = \sum_{i=1}^{t} \left(V_i W_i + \sum_{j<i} V_i X_j \right). \tag{4}$$

We can convert this into relative price units (e.g. bps) by dividing by P_1 (the parent order arrival price) times the total volume:

$$\frac{\sum_{i=1}^{t} V_i \left(W_i + P_i\right) - P_1 \sum_{i=1}^{t} V_i}{P_1 \sum_{i=1}^{t} V_i} = \frac{\sum_{i=1}^{t} \left(V_i W_i + \sum_{j<i} V_i X_j \right)}{P_1 \sum_{i=1}^{t} V_i} \tag{5}$$

To simplify our expression a bit, let's define $V := \sum_i V_i$. Then our slippage vs. arrival in relative price units can be written as:

$$P_1^{-1} V^{-1} \sum_{i=1}^{t} \left(V_i W_i + \sum_{j<i} V_i X_j \right).$$

Let's give this expression of slippage vs. arrival in relative units a convenient variable name:

$$C := (P_1 V)^{-1} \sum_{i=1}^{t} \left(V_i W_i + \sum_{j<i} V_i X_j \right). \tag{6}$$

Now, we are interested in modeling the expected value of C as a function of order parameters. Building a robust and effective model of the expectation of C will take considerable amounts of

data, so we will crucially need good ways of normalizing our variables across symbols so that we can meaningfully pool data. But before we worry about that - let's take a look at a (criminally) simplified example to see to the heart of why studying W_i and X_i values individually can yield sharper results than merely calculating C empirically.

3 Analyzing Error for the Simplest Case of One Order

Let's think about what happens when we have a single order, and the volume traded in each bucket i is some constant v, resulting in $V = tv$. For simplicity, let's set $P_1 = 1$. We then have:

$$C = \frac{1}{tv} \sum_{i=1}^{t} \left(vW_i + \sum_{j<i} vX_j \right) = \frac{1}{t} \sum_{i=1}^{t} \left(W_i + \sum_{j<i} X_j \right). \tag{7}$$

Let's imagine that each W_i is an independent, identically distributed random variable that has some expected value μ_W and some variance σ_W^2, and each X_i is an independent, identically distributed random variable that has some expected value μ_X and some variance σ_X^2.

We then have:

$$\begin{aligned}
\mathbb{E}[C] &= t^{-1}\mathbb{E}\left[\sum_{i=1}^{t} \left(W_i + \sum_{j<i} X_j \right) \right] \\
&= t^{-1} \sum_{i=1}^{t} \left(\mathbb{E}[W_i] + \sum_{j<i} \mathbb{E}[X_j] \right) \\
&= t^{-1} \sum_{i=1}^{t} \left(\mu_W + \sum_{j<i} \mu_X \right) \\
&= t^{-1} \sum_{i=1}^{t} (\mu_W + (i-1)\mu_X) \\
&= \mu_W + t^{-1}\mu_X \sum_{i=1}^{t} (i-1) \\
&= \mu_W + \frac{(t-1)}{2}\mu_X.
\end{aligned}$$

The final step here comes from the general formula for triangulur numbers:

$$\sum_{i=1}^{t-1} i = \frac{t(t-1)}{2}.$$

We therefore have:

$$\mu_C := \mathbb{E}[C] = \mu_W + \frac{(t-1)}{2}\mu_X \tag{8}$$

3.1 Estimating μ_C Directly at the Parent Order Level

Let's suppose we trade just one parent order, and we compute the empirical value of C that results, and define this as an estimate $\hat{\mu}_C$ for the true expected value of C.

In other words,

$$\hat{\mu}_C := t^{-1} \sum_{i=1}^{t} \left(W_i + \sum_{j<i} X_j \right), \tag{9}$$

for a single sample of each W_i and X_j.

We can then consider the expected squared error of this estimate, and we start by plugging in the values of $\hat{\mu}_C$ from μ_C from equations (9) and (8):

$$\mathbb{E}\left[(\hat{\mu}_C - \mu_C)^2\right] = \mathbb{E}\left[\left(t^{-1}\sum_{i=1}^{t}\left(W_i + \sum_{j<i} X_j\right) - \left(\mu_W + \frac{(t-1)}{2}\mu_X\right)\right)^2\right]$$

We can then rearrange the terms to zero-center the random variables.

$$= t^{-2}\mathbb{E}\left[\left(\sum_{i=1}^{t}\left((W_i - \mu_W) + \sum_{j<i}(X_j - \mu_X)\right)\right)^2\right]$$

We rewrite the centered random variables as W' and X' for notational simplicity.

$$= t^{-2}\mathbb{E}\left[\left(\sum_{i=1}^{t}\left(W_i' + \sum_{j<i} X_j'\right)\right)^2\right]$$

We expand the square with non-conflicting index variables for clarity.

$$= t^{-2}\mathbb{E}\left[\left(\sum_{i=1}^{t}\left(W_i' + \sum_{j<i} X_j'\right)\right)\left(\sum_{u=1}^{t}\left(W_u' + \sum_{w<u} X_w'\right)\right)\right]$$

Because expectation is linear, we can express this as

$$= t^{-2}\sum_{i=1}^{t}\sum_{u=1}^{t}\left(\mathbb{E}\left[W_i' W_u'\right] + \mathbb{E}\left[W_i' \sum_{w<u} X_w'\right] + \mathbb{E}\left[W_u' \sum_{j<i} X_j'\right] + \mathbb{E}\left[\sum_{j<i} X_j' \sum_{w<u} X_w'\right]\right).$$

Because the random variables are zero-centered and independent, the two middle terms and all non-diagonal terms are zero. This results in a simple simplification for the W_i terms.

$$= t^{-2}\left(\sum_{i=1}^{t}\left(\mathbb{E}\left[(W_i')^2\right]\right) + \sum_{i=1}^{t}\sum_{u=1}^{t}\mathbb{E}\left[\sum_{j<i} X_j' \sum_{w<u} X_w'\right]\right)$$

Continuing to use the fact that all non-diagonal products have expectation zero, we count the number of diagonal terms along the bucket index.

$$= t^{-2}\left(\sum_{i=1}^{t}\left(\mathbb{E}\left[(W_i')^2\right]\right) + \mathbb{E}\left[\sum_{i=1}^{t}\sum_{j<i}(t-j)\left(X_j'\right)^2\right]\right)$$

$$t^{-2} = \left(\sum_{i=1}^{t}\left(\mathbb{E}\left[(W_i')^2\right]\right) + \sum_{i=1}^{t}\sum_{j<i}(t-j)\mathbb{E}\left[\left(X_j'\right)^2\right]\right)$$

$$= t^{-2}\sum_{i=1}^{t}\left(\sigma_W^2 + \sum_{j<i}(t-j)\sigma_X^2\right)$$

$$= t^{-1} \cdot \sigma_W^2 + t^{-2}\sigma_X^2 \sum_{i=1}^{t} \sum_{j<i} (t-j)$$
$$= t^{-1}\sigma_W^2 + t^{-2}\sigma_X^2 \sum_{i=1}^{t-1} i^2$$
$$= \frac{\sigma_W^2}{t} + \frac{(t-1)(2t-1)}{6t}\sigma_X^2$$

In the end, the result of this derivation is that

$$\mathbb{E}\left[(\hat{\mu}_C - \mu_C)^2\right] = t^{-1}\left(\sigma_W^2 + \frac{(t-1)(2t-1)}{6}\sigma_X^2\right)$$

for this extremely simplified case where we assume a single order with constant volume traded in each bucket.

3.2 Estimating μ_C via Components (μ_W and μ_X)

Now let's consider what happens if we look at the behavior of our single order in individual buckets and form estimates of μ_W and μ_X. We can then combine them into an an overall estimate of μ_C. In other words, we can compute

$$\tilde{\mu}_C := \tilde{\mu}_W + \frac{(t-1)}{2}\tilde{\mu}_X \tag{10}$$

where we have

$$\tilde{\mu}_W := \frac{1}{t}\sum_{i=1}^{t} W_i \tag{11}$$

and

$$\tilde{\mu}_X := \frac{1}{t}\sum_{i=1}^{t} X_i. \tag{12}$$

We compute the squared error by plugging in the values of $\tilde{\mu}_C$ and μ_C from equations (10) and (8).

$$\mathbb{E}\left[(\tilde{\mu}_C - \mu_C)^2\right] = \mathbb{E}\left[\left(\left(\tilde{\mu}_W + \frac{(t-1)}{2}\tilde{\mu}_X\right) - \left(\mu_W + \frac{(t-1)}{2}\mu_X\right)\right)^2\right]$$

We can further plug in the values of $\tilde{\mu}_W$ and $\tilde{\mu}_X$.

$$= \mathbb{E}\left[\left(\left(\frac{1}{t}\sum_{i=1}^{t} W_i + \frac{(t-1)}{2}\frac{1}{t}\sum_{i=1}^{t} X_i\right) - \left(\mu_W + \frac{(t-1)}{2}\mu_X\right)\right)^2\right]$$

We can then rearrange the terms to zero-center the random variables.

$$= \mathbb{E}\left[\frac{1}{t^2}\left(\sum_{i=1}^{t}(W_i - \mu_W) + \frac{(t-1)}{2}\sum_{i=1}^{t}(X_i - \mu_X)\right)^2\right]$$

We rewrite the centered random variables for notational simplicity and rearrange the terms.

$$= \mathbb{E}\left[\frac{1}{t^2}\left(\sum_{i=1}^{t} W_i' + \frac{(t-1)}{2}\sum_{i=1}^{t} X_i'\right)^2\right] = \mathbb{E}\left[\frac{1}{t^2}\left(\sum_{i=1}^{t} W_i' + \frac{(t-1)}{2} X_i'\right)^2\right]$$

We expand the square with non-conflicting index variables for clarity.

$$= \frac{1}{t^2}\mathbb{E}\left[\left(\sum_{i=1}^{t} W_i' + \frac{(t-1)}{2} X_i'\right)\left(\sum_{u=1}^{t} W_u' + \frac{(t-1)}{2} X_u'\right)\right]$$

Because the random variables are i.i.d. and zero-centered, all the non-diagonal products are zero in expectation, so after multiplying this out and simplifying a bit, we get:

$$= \frac{1}{t^2}\sum_{i=1}^{t}\left(\mathbb{E}\left[(W_i')^2\right] + \frac{(t-1)^2}{4}\mathbb{E}\left[(X_i')^2\right]\right)$$

$$= t^{-1}\left(\sigma_W^2 + \frac{(t-1)^2}{4}\sigma_X^2\right)$$

Thus, the result of this derivation is that

$$\mathbb{E}\left[\left(\tilde{\mu}_C - \mu_C\right)^2\right] == t^{-1}\left(\sigma_W^2 + \frac{(t-1)^2}{4}\sigma_X^2\right)$$

in the case where we assume equal volume traded in each bucket.

3.3 Comparing Estimation Methods

We now want to compare the expected squared error of the two methods to see how much improvement we get by using the component-level estimations.

The component-wise estimation does have lower expected squared error than the direct estimation:

$$\mathbb{E}\left[\left(\hat{\mu}_C - \mu_C\right)^2\right] = t^{-1}\left(\sigma_W^2 + \frac{(t-1)(2t-1)}{6}\sigma_X^2\right) \geq t^{-1}\left(\sigma_W^2 + \frac{(t-1)^2}{4}\sigma_X^2\right) = \mathbb{E}\left[\left(\tilde{\mu}_C - \mu_C\right)^2\right].$$

The difference here is coming entirely from the X terms, since

$$\frac{(t-1)(2t-1)}{6} \geq \frac{(t-1)(t-1)}{4}$$

is equivalent to

$$\frac{2t-1}{6} \geq \frac{t-1}{4}.$$

Notice that this holds for $t = 1$, and the lefthand side grows by $+\frac{1}{3}$ everytime we increase t by $+1$, wherease the righthand side grows only by $+\frac{1}{4}$, Thus, the amount of absolute increase in the expected squared error due to direct estimation vs. component-wise estimation grows linearly in the number of buckets.

Looking at this more closely, we can see intuitively where the difference for the X terms is coming from. Each X_i term is ultimately contributing $i^2\sigma_X^2$ inside the expected squared error calculation for direct estimation at the parent level, and the sum of these squares over i is adding up to more than the square of the average over i that is happening inside the component-wise estimation:

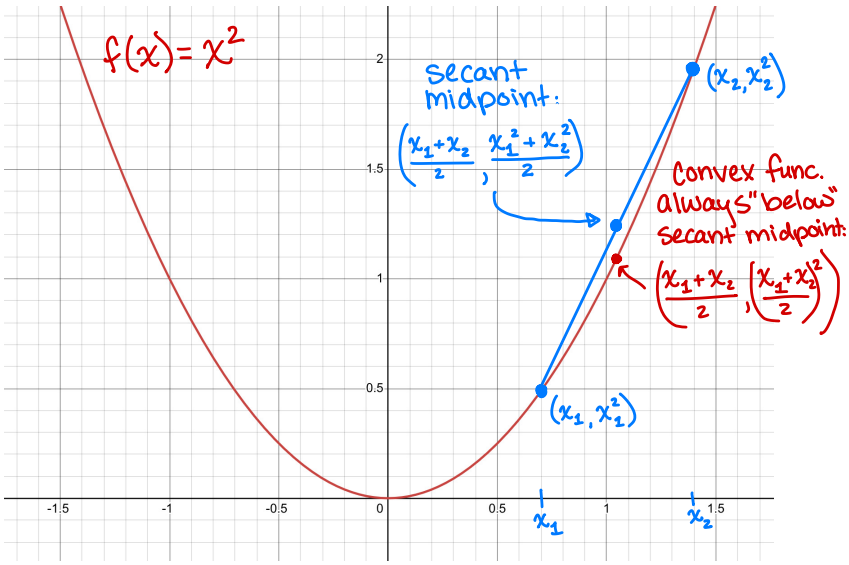
$$\frac{1}{t}\sum_{i=1}^{t-1} i^2 \geq \frac{1}{t^2}\left(\sum_{i=1}^{t-1} i\right)^2$$

Plugging in the formulas for these sums, we can rewrite this as:

$$\frac{(t-1)(2t-1)}{6} \geq \frac{(t-1)^2}{4},$$

as we saw above.

In general, an average of squares will be greater than the square of the average. This is because of convexity of the function $f(x) = x^2$:



We are not seeing this effect for the W terms because the coefficients of σ_W^2 are all 1, and when we have a set of equal numbers, the average square and the square of the average are the same. Hence it is the compounding of the effects of X_i over future time buckets that is causing the component-wise estimation to improve the expected squared error.

We might wonder - if the improvement from component-wise estimation is expressed here through the convexity of squaring, will the improvement disappear if we look at absolute error instead of squared error? In other words, is

$$\mathbb{E}\left[|\hat{\mu}_C - \mu_C|\right] \geq \mathbb{E}\left[|\tilde{\mu}_C - \mu_C|\right]?$$

It turns out that this inequality holds true as well. To see why, consider

$$\mathbb{E}\left[\left|\sum_{i=1}^{t}\left(W_i' + \sum_{j<i} X_j'\right)\right|\right] \geq \mathbb{E}\left[\left|\sum_{i=1}^{t} W_i' + \frac{(t-1)}{2}X_i'\right|\right]$$

These quantities are taken from intermediary points in our analyses above, but with the squaring now replaced by an absolute value. The lefthand side can then be rewritten as:

$$\mathbb{E}\left[\left|\sum_{i=1}^{t}(W_i' + (t-i)X_i')\right|\right]$$

Recall that the X_i' variables are zero-centered and i.i.d, so the central theorem tells us about the distribution of sums like $\sum_{i=1}^{t} X_i'$ as t tends to infinity. Their convergence reflects typical cancelation across the X_i' variables, which gets disrupted potentially by unequal coefficients like $t - i$ in the summation $\sum_{i=1}^{t}(t-i)X_i'$. This will result in less expected cancelation and hence a greater absolute value.

To see this more clearly, imagine an extremal case: a sum of i.i.d. X_i' variables with most coefficients equal to 1, except for one "heavy" coefficient, resulting in a final term like $1000X_t'$. Then the weight of this term dominates the overall expression, and the expected absolute value will be close to the expected magnitude of this term alone, masking the cancelation effects of X_i' variables for $i < t$.

These kind of effects will stay at the heart of our analyses as we add complications on top of it that can make things harder to see, which is why have detailed this simplified example despite its (rather ridiculous) assumptions.

4 Modeling Impact Across Orders

Now let's start enhancing our model to relax some of the unrealistic assumptions we've made in our simplest example above. Realistically, we are going to be constructing our understanding of the distributions of variables like W_i, X_i from samples across multiple parent orders, and these parent orders are going to behave differently due to order characteristics (e.g. the total volume of the order), as well as symbol characteristics (e.g. the typical level of price volatility in the symbol, the typical volume curve, and average daily volume/ADV). External market trends that are (relatively) independent of our trading will also affect price evolution during the lifetime of our orders, making it more difficult for us to see causal effects.

To account for such things, let's introduce a few helpful variables. We'll let σ_s denote the standard deviation of relative price movements in a given symbol s, and we'll let μ_s denote the average trend in relative price movements for a symbol s, measured over some time scale greater than our trading (and thus relatively independent of our trading). We'll also let $A_{s,i}$ denote the average volume traded in symbol s in bucket i (one could alternatively use the median, or any similar metric here). This can be computed for using some rolling window over recent historical market data, say 20 or 30 days, for example.

Perhaps it is then reasonable to decompose our variable X_i as

$$X_i := \mu_s + \sigma_s h(V_i/A_{s,i}) + \sigma_s H_i, \tag{13}$$

where $h(V_i/A_{s,i})$ represents the impact of our trading in bucket i on the next reference price, and H_i denotes market noise. By construction here, H_i is intended to be random variable whose mean is 0 and whose standard deviation is 1. The inputs we have chosen for h reflect an assumption that how much we trade in bucket i (as a fraction of the typical volume) controls our impact on the price evolution going forward. We could also parametrize σ_s as a function of i, but we choose not to do so here because it clutters the notation (and introduces potential challenges of estimating σ_s granularly as a function of i from an effectively smaller sample size within each value of i).

We can similarly decompose our W_i variables:

$$W_i := \frac{\mu_s}{2} + \sigma_s g(V_i/A_{s,i}) + \sigma_s G_i, \tag{14}$$

where $g(V_i/A_{s,i})$ represents the impact of our own trading in bucket i on our average fill price in bucket i (normalized by P_1), and G_i represents market noise that is intended to be mean 0 and standard deviation 1. We have added $\frac{\mu_s}{2}$ here because if we simplistically think of our volume within bucket i as uniformly distributed over time and our larger market trend as a linear function of time, than $\frac{\mu_s}{2}$ is the amount of price change we would expect to incur from the larger market forces.

Let's plug these decomposed models back into our expression for C:

$$\begin{aligned} C &:= (VP_1)^{-1} \sum_{i=1}^{t} \left(V_i W_i + \sum_{j<i} V_i X_j \right) \\ &= (VP_1)^{-1} \sum_{i=1}^{t} \left(V_i(\frac{\mu_s}{2} + \sigma_s g(V_i/A_{s,i}) + \sigma_s G_i) + \sum_{j<i} V_i(\mu_s + \sigma_s h(V_j/A_{s,j}) + \sigma_s H_j) \right) \end{aligned} \tag{15}$$

In taking the expectation of this, let's assume that the larger market trend represented by μ_s has expectation 0. We then have:

$$\mu_C := \mathbb{E}[C] = (VP_1)^{-1}\sigma_s \sum_{i=1}^{t} \left(\mathbb{E}[V_i g(V_i/A_{s,i})] + \sum_{j<i} \mathbb{E}[V_i h(V_j/A_{s,j})] \right) \tag{16}$$

We can think of μ_C here as a function of order/symbol parameters $V, P_1, \{A_{s,i}\}, \sigma_s$, where the V_i's are random variables that depend on our algo behavior, and g, h are deterministic functions that also depend on our algo behavior.

In an A/B test context, we are interested in estimating the function $\mu_C(V, P_1, \{A_{s,i}\}, \sigma_s)$ separately for algo version A and algo version B, so that we can discover ranges of parameters for which one version of the algo outperforms the other. We will additionally need robustness checks to give us a sense of whether such outperformance can be alternatively explained by the noise in our data set, and an accounting of any explicit trading fees that affect the all-in performance of A vs. B in addition to slippage vs. arrival.

4.1 Case Study: Linear functions for g, h

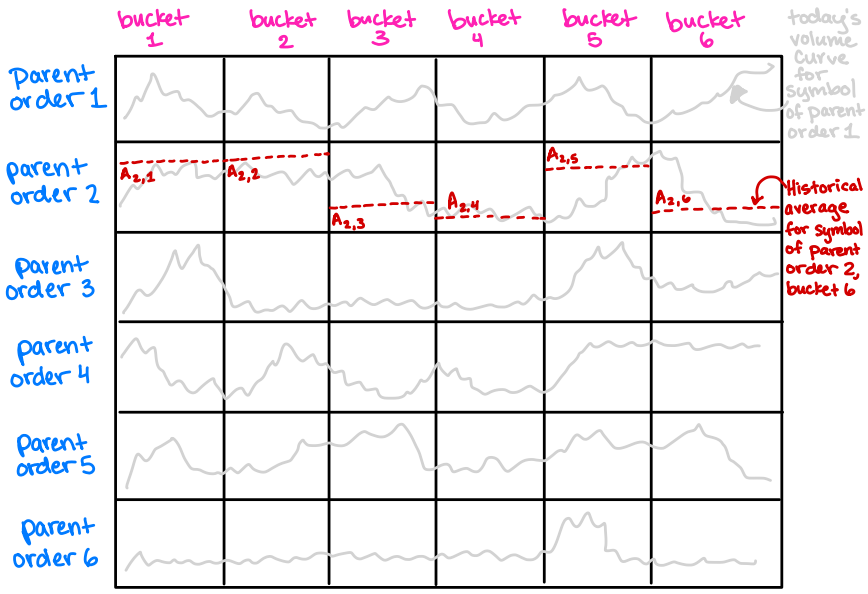
To see how we might go about estimating μ_C as a function of symbol and order parameters, let's consider a particularly simple case where g, h are linear functions of $V_i/A_{s,i}$ and independent of i. More precisely, let's assume $g(x) = \alpha x$ and $h(x) = \beta x$ for some positive constants α and β. In this case:

$$\begin{aligned} \mu_C(V, P_1, \{A_{s,i}\}, \sigma_s) &= (VP_1)^{-1}\sigma_s \sum_{i=1}^{t} \left(\alpha \mathbb{E}\left[\frac{V_i^2}{A_{s,i}}\right] + \beta \sum_{j<i} \mathbb{E}\left[\frac{V_i V_j}{A_{s,j}}\right] \right) \\ &= (VP_1)^{-1}\sigma_s \sum_{i=1}^{t} \left(\frac{\alpha}{A_{s,i}} \mathbb{E}\left[V_i^2\right] + \sum_{j<i} \frac{\beta}{A_{s,j}} \mathbb{E}\left[V_i V_j\right] \right) \end{aligned}$$

Carrying around the multiplier of $(VP_1)^{-1}$ in our expressions is going to get annoying very quickly, and it doesn't really affect the heart of why component-wise estimation may outperform direct estimation. So for the rest of this section, we'll drop it (effectively assuming it is a fixed constant across all of our orders).

4.1.1 Estimating directly

Let's imagine we have a set of n parent orders, indexed by k. For notational simplicity, will use the notation $A_{k,i}$ to mean $A_{s,i}$ and σ_k to mean σ_s for the symbol s specified by parent order k. We can think of our data as a grid where the rows represent parent orders, and the columns represent time buckets of the day.



We might be interested in, say, the expected value of C over the distribution of order parameters represented by our empirical data, as well as over the randomness inherent in the market. We might estimate this directly over our set of parent orders by computing:

$$\begin{aligned}\hat{\mu}_C &:= \frac{1}{n}\sum_{k=1}^{n}\sigma_k\left(\sum_{i=1}^{t}V_{k,i}\left(\alpha(V_{k,i}/A_{k,i})+G_{k,i}\right)+\sum_{j<i}V_{k,i}\left(\beta(V_{k,j}/A_{k,j})+H_{k,j}\right)\right)\\ &= \frac{1}{n}\sum_{k=1}^{n}\sigma_k\sum_{i=1}^{t}\left(\frac{\alpha}{A_{k,i}}V_{k,i}^2+V_{k,i}G_{k,i}+\sum_{j<i}\frac{\beta}{A_{k,j}}V_{k,i}V_{k,j}+V_{k,i}H_{k,j}\right),\end{aligned}$$

where $V_{k,i}$ denotes our volume traded in time bucket i for order k, and $A_{k,i}$ denotes the historical average market volume traded in time bucket i for the symbol specified in order k.

The astute reader will notice that we have removed the μ_s parameters reflecting longer term market trends. We are assuming for (implausible) simplicity here that we can separately compute such parameters perfectly and thus subtract them from our observations. The order level parameter of σ_k is also annoying to carry around, so we will simplify our further expressions by setting $\sigma_k = 1$ across orders. Just to emphasize - these simplifications are not things we would assume in practice, they are merely conveniences that we can employ to make our expressions here more digestible for pedagogical purposes, and you'll have to trust us (or verify for yourself) that such simplifications do not invalidate the points we are ultimately making.

Then we have:

$$\mathbb{E}\left[(\mu_C-\hat{\mu}_C)^2\right] = \mathbb{E}\left[\left(\frac{1}{n}\sum_{k=1}^{n}\frac{\alpha}{A_{k,i}}(V_{k,i}^2-\mathbb{E}\left[V_{k,i}\right]^2)+V_{k,i}G_{k,i}+\sum_{j<i}\frac{\beta}{A_{k,j}}(V_{k,i}V_{k,j}-\mathbb{E}\left[V_{k,i}V_{k,j}\right])+V_{k,i}H_{k,j}\right)^2\right] \tag{17}$$

4.1.2 Estimating with components

Now let's imagine instead that we want to form an overall estimate by separately estimating the quantities like α and β. We can form an estimate $\tilde{\alpha}$ of α, using ordinary least squares regression, taking each $W_{k,i}$ as an opportunity to extract a noisy sample of $\alpha \frac{V_{k,i}}{A_{k,i}}$.

More precisely, let's assume that we have separately computed (implausibly) correct values for μ_k and σ_k (the μ_s, σ_s parameters for the symbol s corresponding to order k), and can therefore compute:

$$\sigma_k^{-1}\left(W_{k,i} - \frac{\mu_k}{2}\right) = g\left(\frac{V_{k,i}}{A_{k,i}}\right) + G_{k,i} = \alpha\frac{V_{k,i}}{A_{k,i}} + G_{k,i}.$$

We can then choose $\tilde{\alpha}$ to minimize the quantity:

$$\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\alpha\frac{V_{k,i}}{A_{k,i}} + G_{k,i} - \tilde{\alpha}\frac{V_{k,i}}{A_{k,i}}\right)^2.$$

This leads us to set:

$$\tilde{\alpha} := \alpha + \frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}G_{k,i}}{A_{s,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{s,i}}\right)^2}. \tag{18}$$

Analogously, we can set:

$$\tilde{\beta} := \beta + \frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}H_{k,i}}{A_{s,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{s,i}}\right)^2}. \tag{19}$$

We can then define our component-wise estimate as:

$$\tilde{\mu}_C := \frac{\tilde{\alpha}}{n}\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}^2}{A_{k,i}} + \tilde{\beta}\sum_{j<i}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right) \tag{20}$$

Thus,

$$\begin{aligned}\tilde{\mu}_C - \mu_C = &\frac{1}{n}\sum_{k=1}^{n}\frac{\alpha}{A_{k,i}}(V_{k,i}^2 - \mathbb{E}\left[V_{k,i}\right]^2) + \sum_{j<i}\frac{\beta}{A_{k,j}}(V_{k,i}V_{k,j} - \mathbb{E}\left[V_{k,i}V_{k,j}\right])\\ &+ \frac{1}{n}\left(\frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}G_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right)\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}^2}{A_{k,i}}\right)\\ &+ \frac{1}{n}\left(\frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}H_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right)\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\sum_{j<i}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right)\end{aligned}$$

The terms written on the top line here (the ones involving the real values of α and β) also appear inside the squaring inside the expectation in expression (17) above.

4.1.3 Comparing estimation methods

If we assume that $G_{k,i}$ and $H_{k,i}$ values are independent of everything else, than if we expand the square inside $\mathbb{E}\left[(\hat{\mu}_C - \mu_C)^2\right]$ and $\mathbb{E}\left[(\tilde{\mu}_C - \mu_C)^2\right]$, we see that the contribution from the common real α, β term is really just the expectation of its own square (the cross terms involving it have

expectation 0 because G, H's are independent of it and 0-centered). Thus, if we are interested in the *difference* between $\mathbb{E}\left[(\hat{\mu}_C - \mu_C)^2\right]$ and $\mathbb{E}\left[(\tilde{\mu}_C - \mu_C)^2\right]$, it suffices to compare

$$\mathbb{E}\left[\left(\sum_{k=1}^{n}\sum_{i=1}^{t} V_{k,i}G_{k,i} + \sum_{j<i} V_{k,i}H_{k,j}\right)^2\right] \tag{21}$$

versus

$$\mathbb{E}\left[\left(\left(\frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}G_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right)\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}^2}{A_{k,i}}\right) + \left(\frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}H_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right)\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\sum_{j<i}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right)\right)^2\right] \tag{22}$$

Let's further assume that G's and H's are independent of each other, so that the cross terms here don't contribute in expectation when we expand the square, and we there are comparing:

$$\mathbb{E}\left[\left(\sum_{k=1}^{n}\sum_{i=1}^{t} V_{K,i}G_{k,i}\right)^2\right] + \mathbb{E}\left[\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\sum_{j<i} V_{k,i}H_{k,j}\right)^2\right] \tag{23}$$

versus

$$\mathbb{E}\left[\left(\frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}G_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right)^2\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}^2}{A_{k,i}}\right)^2\right] + \mathbb{E}\left[\left(\frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}H_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right)^2\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\sum_{j<i}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right)^2\right] \tag{24}$$

We'll look at this in pieces. First let's consider:

$$\begin{aligned}\mathbb{E}\left[\left(\sum_{k=1}^{n}\sum_{i=1}^{t} V_{k,i}G_{k,i}\right)^2\right] &= \mathbb{E}\left[\sum_{k=1}^{n}\sum_{\ell=1}^{n}\sum_{i=1}^{t}\sum_{a=1}^{t} V_{k,i}V_{\ell,a}G_{k,i}G_{\ell,a}\right] \\ &= \sum_{k=1}^{n}\sum_{\ell=1}^{n}\sum_{i=1}^{t}\sum_{a=1}^{t}\mathbb{E}\left[V_{k,i}V_{\ell,a}\right]\mathbb{E}\left[G_{k,i}G_{\ell,a}\right].\end{aligned}$$

Here we have assumed that the centered noise variables $G_{k,i}$ are independent of our trading volumes $V_{k,i}$. Assuming the noise variables are also independent of each other, this reduces to only the diagonal terms:

$$\mathbb{E}\left[\left(\sum_{k=1}^{n}\sum_{i=1}^{t} V_{K,i}G_{k,i}\right)^2\right] = \sum_{k=1}^{n}\sum_{i=1}^{t}\mathbb{E}\left[V_{k,i}^2\right],$$

as $\mathbb{E}\left[G_{k,i}^2\right] = 1$ due to our normalization.

By performing an analogous expansion of the square for the sum involving $G_{k,i}$ terms in the numerator, we similarly have:

$$\mathbb{E}\left[\left(\frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}G_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right)^2\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}^2}{A_{k,i}}\right)^2\right] = \mathbb{E}\left[\frac{\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}^2}{A_{k,i}}\right)^2}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right]$$

At this point, it helps to recall the Cauchy-Schwarz inequality, which we can state in the form

$$|x \cdot y|^2 \leq (x \cdot x)(y \cdot y),$$

for any real-valued vectors x and y of the same length. Here, we can view the $V_{k,i}$'s as the coordinates of a real-valued vector, and $\frac{V_{k,i}}{A_{k,i}}$ as the coordinates of a real-valued vector, therefore obtaining:

$$\mathbb{E}\left[\frac{\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}^2}{A_{k,i}}\right)^2}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right] \leq \mathbb{E}\left[\sum_{k=1}^{n}\sum_{i=1}^{t}V_{k,i}^2\right] = \sum_{k=1}^{n}\sum_{i=1}^{t}\mathbb{E}\left[V_{k,i}^2\right]$$

.

In fact, we can use the Cauchy-Schwarz inequality again to establish that

$$\mathbb{E}\left[\left(\frac{\sum_{k=1}^{n}\sum_{i=1}^{t}\frac{V_{k,i}H_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right)^2\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\sum_{j<i}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right)^2\right] \leq \mathbb{E}\left[\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\sum_{j<i}V_{k,i}H_{k,j}\right)^2\right].$$

For this, we need to use the trick of expanding the square for $H_{k,i}$ terms like we did for $G_{k,i}$ terms. This reduces our desired inequality to the equivalent form:

$$\mathbb{E}\left[\frac{\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\sum_{j<i}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right)^2}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right] \leq \sum_{k=1}^{n}\sum_{j=1}^{t}\sum_{i,\ell>j}\mathbb{E}\left[V_{k,i}V_{k,\ell}\right]$$

Now if we squint closely enough at the numerator of the lefthand side here, we can see the interior of its square as the inner product of two vectors indexed over k, j values, one with entries $\frac{V_{k,j}}{A_{k,j}}$ and one with entries $\sum_{i>j}V_{k,i}$. Now applying Cauchy-Schwarz, we have:

$$\mathbb{E}\left[\frac{\left(\sum_{k=1}^{n}\sum_{i=1}^{t}\sum_{j<i}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right)^2}{\sum_{k=1}^{n}\sum_{i=1}^{t}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}\right] \leq \mathbb{E}\left[\sum_{k=1}^{n}\sum_{j=1}^{t}\left(\sum_{i>j}V_{k,i}\right)^2\right]$$

By expanding the interior square, we observe:

$$\mathbb{E}\left[\sum_{k=1}^{n}\sum_{j=1}^{t}\left(\sum_{i>j}V_{k,i}\right)^2\right] = \sum_{k=1}^{n}\sum_{j=1}^{t}\sum_{i,\ell>j}\mathbb{E}\left[V_{k,i}V_{k,\ell}\right].$$

Thus, we have established that for this case (with its many assumptions!), component-wise estimation achieves lesser expected square error than direct estimation.

4.2 Case Study: Piecewise linear functions for g, h

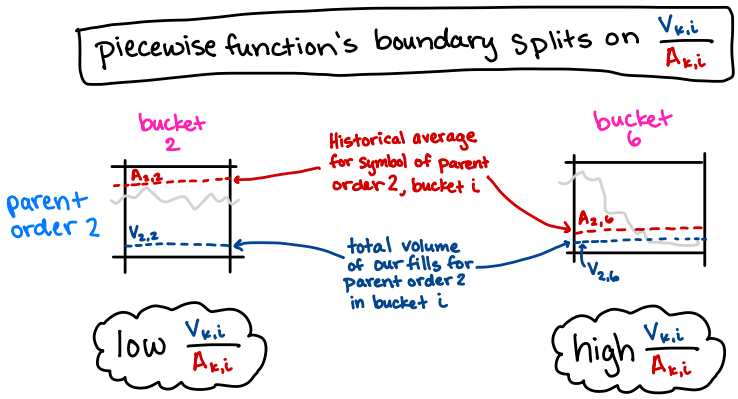
We want to complicate this function a bit, as it is clear that a linear function may not be the best model of impact. Luckily, there is a fairly simple (if notationally complicated) extension of our reasoning which demonstrates that our findings still hold for a piecewise linear function. To illustrate our point, we will use a piecewise function with a single boundary point (two pieces) for both the g and h functions, but our arguments generalize to a function with any number of boundaries. More specifically, we define

$$g(x) = \begin{cases} \alpha_1 x & \text{if } x < L_g \\ \alpha_2 x & \text{if } x \geq L_g \end{cases} \tag{25}$$

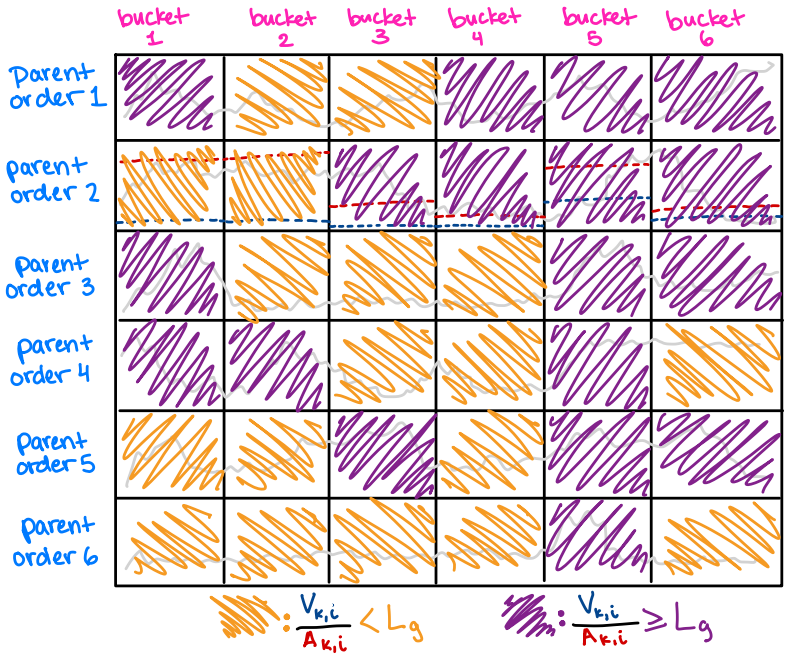
and

$$h(x) = \begin{cases} \beta_1 x & \text{if } x < L_h \\ \beta_2 x & \text{if } x \geq L_h, \end{cases} \tag{26}$$

for positive constants $\alpha_1, \alpha_2, \beta_1, \beta_2$ and thresholding values L_g and L_h. In particular, our piecewise functions are boundaried with respect to the ratio of our trading volume and the historical average trading volume for that symbol and bucket.



At a high level, we can think about this boundary as dividing up the grid of data for each of these piecewise functions, and treating all of our computations separately over these disjoint sets.



Making these sub-functions piecewise means that computing the mean of our cost function will require breaking down the summations into their corresponding sides of the boundaries (letting $(VP_1)^{-1}$ be a constant and setting $\sigma_s = 1$ as we did in the last section for notational simplicity):

$$\mu_C(\{A_{s,i}\}) =$$

$$\sum_{\substack{i\in\{1,\dots,t\} \text{ s.t.} \\ \frac{V_i}{A_{s,i}}<L_g}} \left(\alpha_1 \mathbb{E}\left[\frac{V_i^2}{A_{s,i}}\right] + \beta_1 \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_j}{A_{s,j}}<L_h}} \mathbb{E}\left[\frac{V_i V_j}{A_{s,j}}\right] + \beta_2 \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_j}{A_{s,j}}\geq L_h}} \mathbb{E}\left[\frac{V_i V_j}{A_{s,j}}\right] \right)$$

$$+ \sum_{\substack{i\in\{1,\dots,t\} \text{ s.t.} \\ \frac{V_i}{A_{s,i}}\geq L_g}} \left(\alpha_2 \mathbb{E}\left[\frac{V_i^2}{A_{s,i}}\right] + \beta_1 \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_j}{A_{s,j}}<L_h}} \mathbb{E}\left[\frac{V_i V_j}{A_{s,j}}\right] + \beta_2 \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_j}{A_{s,j}}\geq L_h}} \mathbb{E}\left[\frac{V_i V_j}{A_{s,j}}\right] \right)$$

$$= \sum_{\substack{i\in\{1,\dots,t\} \text{ s.t.} \\ \frac{V_i}{A_{s,i}}\geq L_g}} \left(\frac{\alpha_1}{A_{s,i}} \mathbb{E}\left[V_i^2\right] + \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_j}{A_{s,j}}<L_h}} \frac{\beta_1}{A_{s,j}} \mathbb{E}\left[V_i V_j\right] + \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_j}{A_{s,j}}\geq L_h}} \frac{\beta_2}{A_{s,j}} \mathbb{E}\left[V_i V_j\right] \right)$$

$$+ \sum_{\substack{i\in\{1,\dots,t\} \text{ s.t.} \\ \frac{V_i}{A_{s,i}}<L_g}} \left(\frac{\alpha_2}{A_{s,i}} \mathbb{E}\left[V_i^2\right] + \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_j}{A_{s,j}}<L_h}} \frac{\beta_1}{A_{s,j}} \mathbb{E}\left[V_i V_j\right] + \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_j}{A_{s,j}}\geq L_h}} \frac{\beta_2}{A_{s,j}} \mathbb{E}\left[V_i V_j\right] \right)$$

This equation is notationally quite a bit more complicated than in the linear case. However, it is really a direct extension of that case, splitting each summation into regions based on the relevant thresholds.

4.2.1 Estimating Directly

Similarly to the linear case, we can estimate the expected cost directly over parent orders for the single-boundary piecewise linear function.

$$\hat{\mu}_C := \frac{1}{n}\sum_{k=1}^{n} \left(\sum_{\substack{i\in\{1,\dots,t\} \text{ s.t.} \\ \frac{V_{k,i}}{A_{k,i}}<L_g}} \alpha_1 \frac{V_{k,i}^2}{A_{k,i}} + V_{k,i}G_{k,i} + \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_{k,j}}{A_{k,j}}<L_h}} \beta_1 \frac{V_{k,j}V_{k,i}}{A_{k,j}} + V_{k,i}H_{k,j} + \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_{k,j}}{A_{k,j}}\geq L_h}} \beta_2 \frac{V_{k,j}V_{k,i}}{A_{k,j}} + V_{k,i}H_{k,j} \right.$$

$$\left. + \sum_{\substack{i\in\{1,\dots,t\} \text{ s.t.} \\ \frac{V_{k,i}}{A_{k,i}}\geq L_g}} \alpha_2 \frac{V_{k,i}^2}{A_{k,i}} + V_{k,i}G_{k,i} + \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_{k,j}}{A_{k,j}}<L_h}} \beta_1 \frac{V_{k,j}V_{k,i}}{A_{k,j}} + V_{k,i}H_{k,j} + \sum_{\substack{j<i \text{ s.t.} \\ \frac{V_{k,j}}{A_{k,j}}\geq L_h}} \beta_2 \frac{V_{k,j}V_{k,i}}{A_{k,j}} + V_{k,i}H_{k,j} \right)$$

4.2.2 Estimating with Components

When doing the component-wise estimation for the parameters for the piecewise-linear case, we note that our impact function will also be piecewise, and in fact will be piecewise on the same boundary:

$$\sigma_k^{-1}\left(W_{k,i}-\frac{\mu_k}{2}\right)=\begin{cases}\alpha_1\frac{V_{k,i}}{A_{k,i}}+G_{k,i} & \text{if } \frac{V_{k,i}}{A_{k,i}}<L_g\\ \alpha_2\frac{V_{k,i}}{A_{k,i}}+G_{k,i} & \text{if } \frac{V_{k,i}}{A_{k,i}}\geq L_g\end{cases} \tag{27}$$

Therefore, we can still use ordinary least squares to fit the functions, and we are able to do the estimates for the different pieces of the functions separately. Let's start with fitting $\tilde{\alpha}_1$ to minimize

$$\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}<L_g}}\left(\alpha_1\frac{V_{k,i}}{A_{k,i}}+G_{k,i}-\tilde{\alpha}_1\frac{V_{k,i}}{A_{k,i}}\right)^2, \tag{28}$$

where $\tilde{\alpha}_1$ is an estimate of α_1. Then we have

$$\tilde{\alpha}_1:=\alpha_1+\frac{\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}<L_g}}\frac{V_{k,i}G_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}<L_g}}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}. \tag{29}$$

We can do the same to find $\tilde{\alpha}_2$, $\tilde{\beta}_1$, and $\tilde{\beta}_2$:

$$\tilde{\alpha}_2:=\alpha_2+\frac{\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}\geq L_g}}\frac{V_{k,i}G_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}\geq L_g}}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}, \tag{30}$$

$$\tilde{\beta}_1:=\beta_1+\frac{\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}<L_g}}\frac{V_{k,i}H_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}<L_g}}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2},\text{ and} \tag{31}$$

$$\tilde{\beta}_2:=\beta_2+\frac{\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}\geq L_g}}\frac{V_{k,i}H_{k,i}}{A_{k,i}}}{\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}\geq L_g}}\left(\frac{V_{k,i}}{A_{k,i}}\right)^2}. \tag{32}$$

Therefore, our component-wise estimate will be

$$\begin{aligned}\tilde{\mu}_C:=&\frac{\tilde{\alpha}_1}{n}\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}<L_g}}\left(\frac{V_{k,i}^2}{A_{k,i}}+\tilde{\beta}_1\sum_{\substack{j<i\text{ s.t.}\\ \frac{V_{k,j}}{A_{k,j}}<L_h}}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}+\tilde{\beta}_2\sum_{\substack{j<i\text{ s.t.}\\ \frac{V_{k,j}}{A_{k,j}}\geq L_h}}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right)\\ &+\frac{\tilde{\alpha}_2}{n}\sum_{k=1}^{n}\sum_{\substack{i\in\{1,\ldots,t\}\text{ s.t.}\\ \frac{V_{k,i}}{A_{k,i}}\geq L_g}}\left(\frac{V_{k,i}^2}{A_{k,i}}+\tilde{\beta}_1\sum_{\substack{j<i\text{ s.t.}\\ \frac{V_{k,j}}{A_{k,j}}<L_h}}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}+\tilde{\beta}_2\sum_{\substack{j<i\text{ s.t.}\\ \frac{V_{k,j}}{A_{k,j}}\geq L_h}}\frac{1}{A_{k,j}}V_{k,i}V_{k,j}\right)\end{aligned}$$

4.2.3 Comparing estimation methods

If we were to compute the expected squared error for each of the above estimation methods, we would see that the expected squared error for the component-wise estimation is again dominated by the expected squared error for the direct estimation. This follows from the same tricks we employed for linear functions, namely expanding the square, leveraging the independence assumptions to kill cross terms, and employing the Cauchy-Schwarz inequality. It is prohibitively more painful to write down in this case, as we have to carry around the infrastructure of the sums split by the thresholds of L_g and L_h, but the same proof essentially operates in parallel within each linear piece. This phenomenon extends to an arbitrary number of linear pieces, rather than just the two we have notated here.

5 Additional Considerations for Applying a Component-wise Estimation Framework in Practice

It would be natural to feel some trepidation in applying this framework to an A/B test analysis in practice, as the proofs of superior estimation quality in the previous section are quite intricate and notationally overwhelming. As soon as a practical situation deviates from the simplifying assumptions we made there (and it will!), we might be worried that adjustments we make to the modeling process to mold it more closely to our use case will end up erasing the gains we have proven for simplified cases.

For this reason, it would be nice to have a more general proof, but alas, such a proof has eluded us thus far. We suspect that a general proof might be derivable directly from or in the spirit of the Rao-Blackwell-Kolmogorov theorem [4, 5, 6], a broadly applicable theorem of statistics that establishes the benefit of conditioning an estimator on a "sufficient statistic." We were unable to formulate a direct translation between the structures of that theorem and our modeling context, but we suspect that one may exist (and would love to hear from you if you can see why or why not!).

Nonetheless, if we squint at the core of the proofs of superior estimation quality that we provided for specific model structures in the previous sections, we can see a deep and general level of commonality. Weighted averages with uneven weights contribute more to estimation error than averages with even weights, and this fact is ultimately rather intuitive. Uneven weights can get in the way of zero-centered errors canceling each other out, thus making effective sample size morally smaller and estimation quality lower.

Uneven weights arise in many forms inside an overall calculation of parent order slippage versus arrival, whether due to differing symbol parameters and volume curves, compounding of effects over sequences of time buckets within a trading day, or other sources. We certainly wouldn't want to just blindly replace all such weights with uniform ones. Sometimes this would be undesireable because it would change what we are ultimately measuring: e.g. we really *don't* want to treat all orders as equally important to our performance when some represent a lot of notional value and others don't. Sometimes it's not fully clear what "uniformizing" the weights would mean, as many sources of weighting are implicit in the computation, not to mention the fact that "uniformizing" is not a word.

Hence it makes intuitive sense to build a modeling framework that allows us to isolate important model components, re-weight/normalize our observations in estimating them individually, and then plug them back into an overall weighted calculation that still authentically reflects our total priorities. And it makes sense that we should expect lower errors in expectation from such a framework, subject to the quality of our modeling assumptions and normalization techniques.

Another comforting realization is that error distributions can be simulated under a wide variety of assumptions, and such simulations can empirically demonstrate reduced estimation error in cases beyond what we have provided proofs for. Simulations can also help us to gain confidence that the

results we obtain through such a framework are meaningful. For example, let's suppose we use this framework to estimate the expected costs of algo versions A and B, as functions of relevant order parameters. Suppose we estimate that the expected costs are lower for A in some range of order parameters. This alone should not be treated as definitive, as the chance of the numbers coming out exactly the same for A and B is very small, even if *A and B are actually not different at all!* Therefore, we don't just want to know, "is the estimated cost for A lower than the estimated cost for B?" We further want to know, "is the estimated cost for A lower than the estimated cost for B by an amount that seems unlikely to arise from noise/chance in our experiment if there is no actual difference between the true expected costs of A and B?" We can address this better question by simulation and see if we can get a similar magnitude of result in a hypothetical world where the labels for A and B are randomly scrambled.

All of which is to say - there are a lot of nuances to using this in practice that we have not fully covered in our abstract writeup so far in this whitepaper, but such issues are generally addressable and the framework's flexibility is a strength in this sense.

But perhaps the best way to reason about the usefulness of our framework in practice is to... well... use it in practice. So here we provide an account of how this framework was employed to analyze results from an A/B test that Proof ran to compare two potential settings of our tactical posting behavior.

5.1 The A and the B

The two worlds we are comparing here concern two variations of our "worker" logic. The worker logic is the layer of our algo stack that handles how and where small pieces of a larger parent order are traded over an interval of several minutes. Skipping over a number of intricacies, a key part of the worker logic is our posting behavior within an interval. That is, where and how we post orders at the bid price when buying (or ask price when selling). Version A of our of algo is our default world in which we post orders split between two standard venues. Version B of our algo posts between these two standard venues and additionally an *inverted exchange*. (For any readers who are not market structure nerds, an inverted exchange is where takers receive a rebate and makers pay a fee, which is the opposite of the usual model.) Therefore, the objective of our experiment is to ascertain the difference in estimated parent order performance between posting only to our standard venues and posting to our standard venues with the addition of the inverted exchanged. We collected data over roughly a nine month period in 2025. We executed approximately half of our orders using algo engine A and half of our orders using algo engine B. [Aside: this probably sounds like a long data collection period - it is! The time period of data collection needed to produce meangingful models depends greatly on the complexity of what we're modeling, the level of noise, and the amount of data that is produced each day. As Proof's business grows, hopefully we will have more orders per day on average, enabling much faster experimentation. In parallel, we also hope to continue improving our noise mitigation techniques.]

We will next describe the decisions we made to be able to compare orders across different symbols and time and ultimately compute our estimated cost metric to compare between A and B.

5.2 Refining the relevant volume variables

Recall that in Section 4 we decomposed our X_i variables as

$$X_i := \mu_s + \sigma_s h(V_i/As, i) + \sigma_s H_i, \tag{33}$$

and our W_i variables as

$$W_i := \frac{\mu_s}{2} + \sigma_s g(V_i/A_{s,i}) + \sigma_s G_i. \tag{34}$$

Essentially we have decomposed both the impact we have and the fill price we get during a particular interval as a function of the volume we traded during the interval and some component of

market noise. There are a number of complexities we need to handle when moving from our abstract formalization to the real world as it pertains to computing these intermediate values.

The first to handle is simply the notion of an interval. Our time bucket abstraction from the prior sections implicitly assumed that buckets are of the same length, however the *natural* interval imposed by our worker logic does not abide by this. That is, what one would consider a single execution of the worker logic does not run in a fixed and discrete interval, but rather a varied and random interval. Moreover, parent level orders do not begin and end at the same time during a day. To ameliorate this, we simply impose that we break all parent orders into 39 discrete 10-minute buckets over the entirety of the trading day, 9:30 AM to 4:00 PM (omitting the auctions). This ensures that we are comparing over the same bucket definition across all orders. Of course, this could mean that our common definition of bucket could contains parts of a one or more, or zero executions of our worker logic. However, this is gracefully handled by how we apply our modeling assumptions.

To compute the term $V_i/A_{s,i}$ we need as input to h and g, we first simply compute V_i as the total number of shares we traded in a given bucket. During the process of feature design, we found that we achieved better results from our model by decomposing V_i into two components: the amount of fills we got from posting V_i^p and the amount of fills we got from taking V_i^t, such that $V_i = V_i^p + V_i^t$. Our worker logic, in a single interval, attempts to post all of the shares assigned to that interval by the scheduler (a higher layer of our algo stack), and will cross the spread (take) as needed if we do not get enough fills from posting after a set amount of time (also determined by the scheduler). Again, our common definition of a bucket may include multiple (potentially partial) intervals of the worker logic or potentially none at all if the order has not yet started or is limited away. However, this does not matter in terms of our definitions of g, h. They are simply linear functions of the variables V_i^p, V_i^t defined over a common bucket length and not inherently tied to any particular worker logic. [Aside: this is a purposeful disconnect, as we would want our framework to be able to compare worker logic possibilities that operate on different intervals, so keeping the framework as agnostic to the worker level details as possible is desirable.]

Finally, we need to compute $A_{s,i}$, the denominator of our term that is taken as inputs to h, g. This term represents the average number of shares that were traded in the interval for that symbol in recent days. We compute this from historical market data with a 20-day trailing window. This means that $V_i/A_{s,i}$ is similar to a participation rate, except that the $A_{s,i}$ denominator is historical instead of contemporaneous. We would expect that things would behave roughly similarly if we used the actual market volume traded in that interval, but we think the historical expectation is a cleaner choice as it is independent of our own trading that day.

5.3 De-noising through removal of longer term market trends

We are still left to compute μ_s, σ_s, the average trend and standard deviation of relative price movements in a given symbol s measured over some time scale greater than our trading period for that symbol. The idea is that due to this difference in time scales, this trend will be relatively independent of our trading. Overall, we want to use these values to separate the effect of our own behavior from that of a broader and persistent market trend. If a symbol is moving consistently in one direction over a longer time period, we do not want to give ourselves too much credit when the price is moving into us or punish ourselves too much when the price is moving against us, as there is evidence that the driving forces extend beyond our influence.

For an order with a given symbol s that traded on a day d, we find the trading week w that the order traded. For any trading day d that falls within a trading week (Monday through Friday) we compute the long-term trend over that week. That is, weekends serve as a barrier, and an order for a symbol s that trades on a Tuesday is de-noised using the same trend for an order that trades that same symbol s three days later on a Friday, as opposed to computing this trend over rolling calendar weeks. [Note: we are documenting all of our exact choices here for completeness, but we

expect that there are many similarly reasonable choices one could make, and the hope is that the framework and its results are relatively robust to these things.]

To compute the average and standard deviation of the week long trend, we again break each day in the week into discrete 10-minute buckets over the entirety of the trading day, 9:30 AM to 4:00 PM, using the auction prices as our first and last price benchmarks for the day. For each time bucket in each day we compute the price deltas as the difference between the NBBO at the end and the NBBO at the start of an interval. We collect all such price deltas over the week – 195 of them for a typical trading week without holidays. We denote this set as Δ.

We originally tried to compute the simple mean and standard deviation of Δ to obtain μ_s, σ_s, however we observed that our distribution of price deltas was heavily skewed with a number of extreme outliers. This led us to creating models for estimates of μ_W and μ_X that were not stable. Instead, we landed on using the median of Δ as a more robust estimator of μ_s and the median absolute deviation (MAD) as a more robust estimator of σ_s. MAD is defined as the median of the absolute differences from the data set's median. For our data set $\Delta = \{\delta_1, \delta_2, \ldots, \delta_m\}$ we have

$$\text{MAD}(\Delta) = \text{median}(|\delta_i - \text{median}(\Delta)|).$$

5.4 Modeling our expected costs as piecewise linear functions

After computing $\{(V_i^p, V_i^t, A_{s,i}\}, \mu_s, \sigma_s$ for all of the parent orders in our data set, we are left to estimate μ_W and μ_X via estimations of g, h. As considered in Section 4.2, we model g, h as piecewise linear functions of $V_i^p/A_{s,i}$ and $V_i^t/A_{s,i}$, where the boundaries of the pieces are dependent on the value of $(V_i/A_{s,i})$ in a particular bucket.

We breakdown $V_i/A_{s,i}$ into six brackets defined in Table 1 and separate our buckets into these brackets. For extremal brackets 0 and 5, we simply estimate a constant for both g, h. We do this because the data in these brackets is quite noisy and when we try to run a full linear regression we get unstable results, e.g., computing the slippage as negative for a large number of data points in these brackets. For bracket 0 we estimate 0, as we expect to have little impact when trading such small amounts, thus a prediction of 0 is sensible. For bracket 5 we estimate the empirical median value of $(X_i - \mu_s)/\sigma_s$ and $(W_i - (\mu_s/2))/\sigma_s$ – the median of our empirical observations of h, g in this bracket, respectively. This serves as a blunt yet robust estimator of g, h for these especially heavy order buckets.

For the rest of the brackets $1 - 4$, we run a separate linear regression on independent variables $V_i^p/A_{s,i}, V_i^t/A_{s,i}$ (the separation into fills that were posted versus from crossing the spread as explained in Section 5.2) while enforcing no intercept. We train a single model over a random selection of 60% of the entire data set, that is data points belonging to both A and B. We do this because we observed that training separate models on A and B essentially yielded the same result, and the real difference existed in the distribution of $V_i^p/A_{s,i}, V_i^t/A_{s,i}$ between the two strategies. Thus, combining the models simply served to boost the sample size and presumably improve model quality.

Bracket	$V_i/A_{s,i}$ **range**
0	0%-3%
1	3%-6%
2	6%-9%
3	9%-12%
4	12%-15%
5	15%+

Table 1: $(V_i/A_{s,i})$ Brackets

We hold out 40% of the data set as a test set, where we measure the improvement of our estimations of g, h as percentage improvement over a baseline of predicting 0 for all brackets when compared to the empirically observed values of g, h. A baseline of predicting 0 represents a world in which our activity has no impact on the future price of a symbol or the price of our own fills. While rudimentary and unrealistic, it gives a steady anchor to ensure our models are stable and sensible. Our results show that our models achieve modest but stable improvement over this baseline as a whole and across all non-trivial brackets (that is, omitting bracket 0 where our models estimates a constant 0) for both models of g, h.

5.5 Our results

With our models g, h ready to go, we are ready to compute our metrics for all parent orders and compare the results between A and B. To gain a more granular understanding of the estimated cost differences between A and B, we break down parent orders by an order difficultly parameter. Our order difficulty parameter can be understood as the participation rate over the lifetime of an order. We further bracket parent orders by this order difficulty parameter in an analogous way to the way we bracketed $(V_i/A_{s,i})$ on a per bucket level.

Looking at the notional-weighted-median estimated cost for each order difficulty bracket, we find that B has less estimated cost than A across all non-trivial brackets. The upshot being that the change to the worker logic that posts on the inverted exchange appears to provide a stable and relatively significant lower estimated cost across all order difficulties as compared to our (previously) standard worker logic. We present our results adhering to our client privacy checks [1] in Table 2. The range reported is taken from the difference in notional-weighted-median estimated cost between A and B over a number of trials of random splits of our training and testing data set (see below for a discussion on robustness testing) applied to our client privacy check procedure.

Participation Rate Bracket	**NW Median Slippage Improvement Range (bps)**
0/%-1%	0-5
1%-3%	0-5
3%-5%	0-5
5%-7%	0-20
7%+	5-20

Table 2: Estimated range of notational weighted median slippage improvement of strategy B versus A over different participation rate brackets. We report ranges to adhere to our client privacy policy.

One thing to consider, however, is whether this cost savings from posting to an inverted exchange survives net of venue fees. Recall, that with an inverted exchange, we are paying fees to post there as opposed to getting rebates to post at other venues. We examined the venue cost for orders in our data set between orders in A and B. While B on average did incur more venue costs across all order difficulty brackets, the estimated cost savings in using strategy B was at least an order of magnitude larger for a typical order.

We also wanted to ensure that our results are robust. We did this in two ways. First, we retrained our model using 100 different random samplings of our train and test split and recomputed our metric. In all trials, our result that B has less estimated cost than A across all brackets holds. This gives us a strong level of robustness across the random sampling of our train vs. test split. Secondly, we wanted to ensure that the magnitude of the difference between A and B could not be easily explained by chance in the assignment of parent orders to A vs. B in a hypothetical world where there is actually no difference in expected costs between A and B. To assess this, we randomly split our entire data set with artificial labels A* and B*, i.e., each parent order was labeled with a random strategy label independent of what strategy it was actually executed under. We then retrained our

model and ran our metric over 100 trials of these random splits with artificial labels. The results were as expected, with A* and B* each having lower expected cost about half the time across order difficulty brackets over all trials, and the margins of victory typically being smaller than we observed for our actual A/B test.

5.6 Further Exploration

We are currently exploring if there are any other useful order level features that we could integrate into our model without sacrificing the stability and robustness of our results. We have to be very careful about how we go about this. The more we cut our data around additional features, the smaller the sample size is in each partition. This can lead to models of g, h that are not robust, and reported costs from our metric that are not stable. With that said, we are currently investigating if integrating a measure of liquidity as measured by the ADV of a symbol allows us to see liquidity-dependent effects on the impact of our posting decisions.

We are also planning to incorporate this framework into our TCA analyses, as we have demonstrated here that modeling our expected parent order slippage this way can yield a better estimation of our true underlying behavior than raw weighted averaging. Right now the normalization and longer term trend removal techniques here are distinct from our distilled impact approach in TCA, which is similarly intended to reduce the noise in estimated expected parent order slippage, but does so by comparing to the price behavior in a correlated ETF over the same time period of the order's trading. There may be a way to combine the de-noising benefits of our distilled impact approach with the de-noising techniques here, perhaps leading to even further improvements in estimation quality. In addition, the parameters we estimate for functions like g, h here can become their own metrics in TCA analyses to be tracked over time, perhaps leading to insights for algo design that are harder to infer from only parent order slippage.

References

[1] Bishop, A. "Aggregation isn't a privacy guarantee. Here's what we do instead." (2022) `https://medium.com/p/dd0127730cd5`

[2] Bishop, A. "Distilled Impact." (2020). `https://prooftrading.com/docs/distilled-impact.pdf`

[3] Bishop, A. and Schoenbauer, M. "Refining Proxy Symbol Selection for Distilled Impact." (2021) `https://prooftrading.com/docs/distilled-impact-v1.pdf`.

[4] Blackwell, D. "Conditional expectation and unbiased sequential estimation" in Annals of Mathematical Statistics. 18 (1): 105-110. (1947). doi:10.1214/aoms/1177730497

[5] Kolmogorov, A. N. "Unbiased estimates" in Izvestiya Akad. Nauk SSSR. Ser. Mat. 14: 303-326. (1950).

[6] Rao, C. Radhakrishna. "Information and accuracy attainable in the estimation of statistical parameters" in Bulletin of the Calcutta Mathematical Society. 37 (3): 81–91. (1945).