Proof's Design for Public-facing TCA: Initial Results for Six Months of Data

July 19, 2022

Abstract

We present Proof's current framework for evaluating our trading performance. We have designed robustness and client privacy checks that enable us to make any stable results public, and so far we can report: 1. Initial evidence suggests our impact model is performing meaningfully well in practice, 2. behavioral metrics like volume curve tracking look good, and 3. short-term markouts for passive fills at IEX using D-Limit show robust improvement over other venues, supporting our current tactical approach. Slippage stats are too noisy at our current sample sizes to report.

1 Introduction

"The thing about numbers is: there are so many of them, you can just pick the ones you like" - this is a joke we often make, but it hits close to home in the realm of transaction cost analyses (TCA). Measuring algo performance is notoriously fraught, for several reasons. First, market noise can drown out any meaningful signal if the sample sizes are not huge. Second, the sheer number of detailed decisions that go into choosing a single metric, setting a time frame, handling outliers, and screening for data quality, etc. creates a wide universe of resulting possible numbers, from which is it all too easy to cherry-pick (consciously or sub-consciously). Third, the number of metrics that one can compute is also large, making it likely that some coincidences will rise to look meaningful. If we run 100 metrics, or even the same metric over 100 individual symbols, for example, and we use the typical notion of "statistical significance" with a 5% threshold, than we should expect about 5 of our results to show up as "significant," even if *none* of them are actually meaningful.

At this point in our evolution as a company, Proof can do very little to address our challenge of low sample sizes. We simply need more clients and more client activity before we can reach the sample sizes that appear to be needed to produce robust numbers for notoriously noisy stats like slippage vs. arrival. We have developed a metric we call "distilled impact" that removes some of the noise in arrival slippage by referencing the contemporaneously price movement in an ETF that tends to be correlated with the symbol we traded. Our distilled impact numbers do show less noise than arrival slippage, but still more noise than our current sample sizes can overcome.

But we can do more to address the other challenges, even now while we have small samples sizes. In fact, now is an excellent time to address these! Some of the best defenses against misleading results in noisy data are:

- commit to your metrics upfront
- make consistent and documented choices in handling details
- perform robustness checks
- be mindful of the total scale of your analyses when interpreting "significance" of individual results
- subject your findings to public scrutiny, and attempt to replicate results over time

This report is intended to set Proof on a firm foundation on all of these fronts: we will detail the design of the analyses we perform on our own trading activity, and provide the subset of results that currently pass our robustness checks. Our data set will be all of Proof's trading activity in the six month period from October 2021 through March 2022.

There is potentially a conflict between public accountability and the privacy of our clients. Ultimately, the results of our analyses are not only a function of Proof's trading logic, but also a function of when,

what, and how our clients trade. Proof has a rigorous process in place to ensure that any statistics released about our aggregate trading activity do not meaningfully compromise the privacy of our individual clients. Stats that merely describe the overall amount of trading activity at Proof, such as the number of active clients we have, the total notional value traded, our venue breakdown, etc. can simply be released. Stats that describe trading performance in some way, such as average slippage vs. a benchmark, markouts, how closely our VWAP algo follows the volume curve, etc. must be vetted before release to ensure that they are not meaningfully skewed by the trading data of any one individual client. The vetting process is:

1. Define a set of ranges for the stat. [E.g. for a stat that represents a percentage, we might specify that we will round to the nearest multiple of 10%, thereby grouping all possible answers that round to the same thing into a "range."]
2. Compute the stat on our full data set with all client data aggregated. Define the range of the result as the value that is a candidate for release.
3. For each client, remove all of that's client's trading data from the dataset. Recompute the stat and the range it belongs to. If the range does not match the value that is a candidate for release, terminate the process and do not release any value. If the range does match, restore that client's data and go on to repeat the check for the next client.
4. Once we have checked that all removals of a single client's data result in a stat that falls within the same candidate range, we may release the range.

This release process is intended to protect us from releasing statistics that depend heavily on the experience of one particular client. This protects the client's privacy, as well as protecting Proof from relying on numbers that may be misleading because they are not representative of the trading experience across clients. Clients may also opt out of being included in our aggregate analyses at all, as per our data privacy policy. [So far, no clients have taken this route.]

We perform additional robustness checks to help us gauge the level of noise in our aggregate stats. In particular, we look at how much a stat can change based on removing small segments of data. For example, we might gauge the robustness of a performance stat like slippage vs. arrival by removing a small subset of the best performing order or a small subset of the worst performing orders, and see how much the calculated value changes. If a stat exhibits a disconcertingly wide range under this check, we do not rely on that stat in our decision making, nor do we release its value externally. Due to our small trading sizes at this time, many statistics fail at least one of our privacy and robustness checks.

In the following sections, we provide: 1. a summary of the basic stats that describe Proof's overall trading activity, 2. a summary of the suite of analyses we perform to assess our activity and the subsets of results that pass our client privacy and robustness checks, and 3. a more detailed appendix containing full discussion of our each of our metrics.

2 Overall Activity Stats

Here is an overall summary of our trading activity for the time period from October 2021 through March 2022 (inclusive):

- Number of active clients: 6
- Total notional value traded: 1.5 billion
- Notional value traded by our VWAP algo: 1.4 billion
- Notional value traded by our Proof algo: 115 million

Our Proof algo has two components that operate in parallel: a liquidity seeker that employs high min quantities to make large trades in the dark, and an impact minimizer that trades somewhat steadily through a mix of smaller lit and dark orders. In this period, the liquidity seeker traded about 50 million in notional value, while the impact minimization piece traded approximately 65 million in notional value. We must keep in mind throughout that the data for our Proof algo here represents a very small sample size.

2.1 Venue Breakdowns

Let's take a look at how Proof's trading distributes over venues. We will separate our liquidity seeker here from the rest of our flow, as it can be interesting to see more specifically where we are finding relatively larger blocks. Our VWAP algo and the impact minimization component of our Proof algo share a common tactical layer for low-level decision making, so we keep the flow for these together here in one category that we will refer to as the flow for our "algo schedulers." For our purposes here, we round to the nearest 1%, and so we do not list venues that represented less than half of a percent of the total. [For this reason, our displayed values add up to slightly less than 1.]

Here is a breakdown of what percentage of the flow traded by our algo schedulers occurred at each venue, in terms of volume and notional value:

lastMarket	nv	volume
ARCA	0.13	0.03
IEXG	0.45	0.45
NASDAQ	0.21	0.27
NYSE	0.19	0.22

Table 1: Venue Breakdown for Algo Schedulers

The heavy IEX component here is unsurprising, as our posting logic heavily leverages the DLIM order type at IEX. As we will detail below, we have found that this results in improved 1 second markouts for our passive fills.

Here is a breakdown of what percentage of flow traded by our liquidity seeker occurred at each venue, in terms of volume and notional value:

lastMarket	nv	volume
CAES	0.08	0.08
IEXG	0.50	0.46
LEVL	0.03	0.02
SGMT	0.12	0.16
UBSA	0.20	0.23
NASDAQ	0.07	0.06

Table 2: Venue Breakdown for Liquidity Seeker

Our liquidity seeker typically rests dark orders in several venues simultaneously with high minimum quantities, so we expect these proportions to be more a reflection of where there is matching liquidity to be found, rather than a reflection of our tactics. We should remind ourselves here that the sample size so far is fairly small, so there proportions may change dramatically over time. We also expect to connect to a few more venues in the medium-term.

3 Our Suite of Analyses and Summary of Preliminary Results

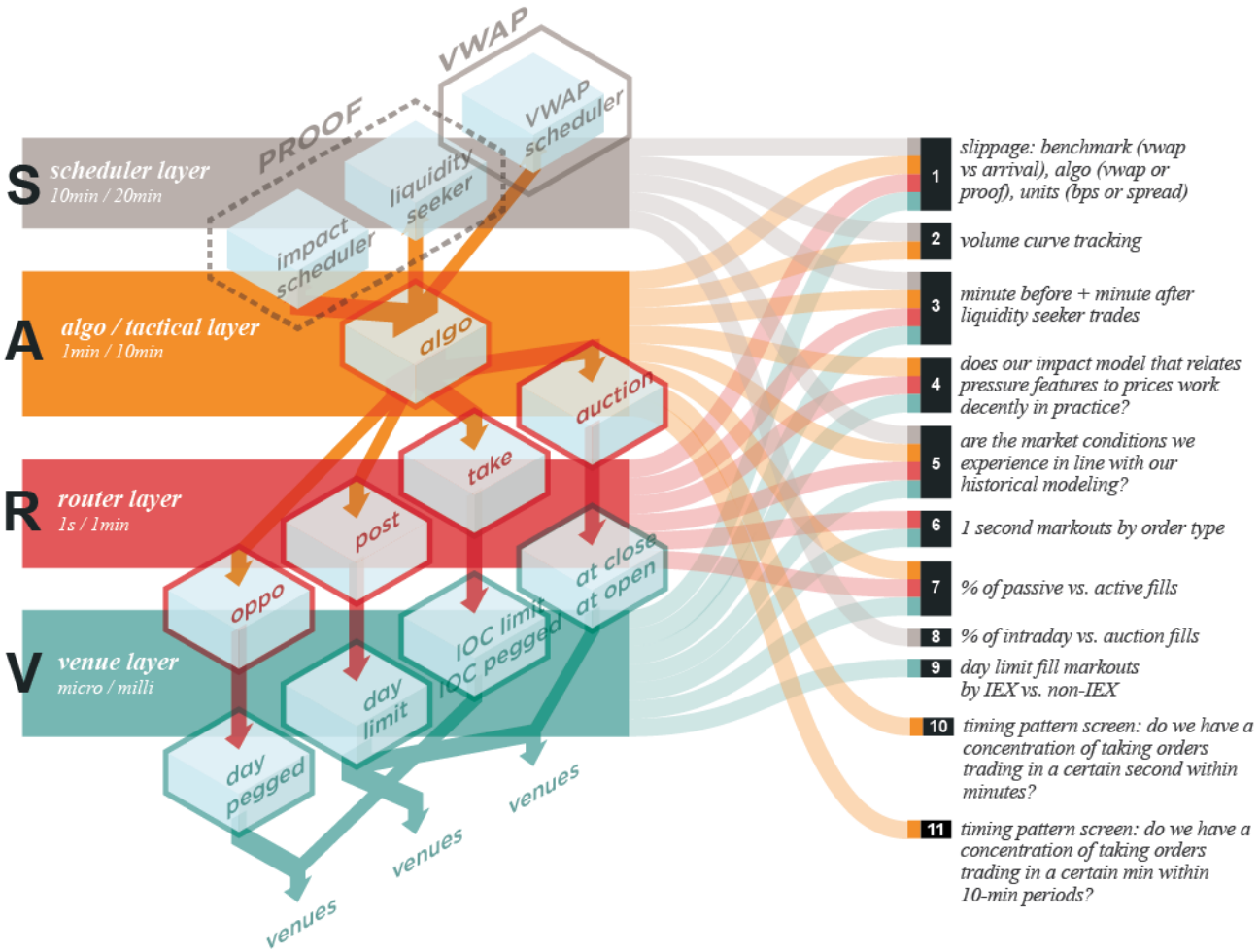
When designing the suite of analyses we perform to evaluate our trading algos, we keep in mind the fact that an "algo" is not really a monolithic thing. There are multiple layers of decision making that translate parent orders into executions, and each layer is roughly responsible for a different time scale. These layers roughly map onto classes in the java code that comprises our trading system. At the top layer, we have schedulers that make decisions about how much we should trade over time intervals that are typically 5-20 minutes long. For our VWAP algo, we use a scheduler that tries to predict and follow the day's volume curve. For our Proof algo, we use a scheduler that aims to minimize our expected impact. The next layer, which we call the "tactical" layer (or the "algo" class in our java code), is responsible for deciding how to trade at a time scales of minutes. It decides how much to post, and how long to wait for passive fills before shuffling some volume to a take router. Below the tactical layer is the router layer, where the detailed logic for posting and taking sits. Our post and take routers, for example, make decisions about what venues to use, and how to manage orders on the timescale of seconds. Each

of our router types produces distinct types of child orders, which are then sent to venues. We rely on tools like IEX's DPEG and DLIM order types, IEX's router, Nasdaq's MELO order type and others to pursue favorable outcomes on the microsecond and millisecond time scales.

As we try to assess the performance of our algo, noise accumulates at every time scale, and is greatest at the larger time scales. This is why metrics like 1 second markouts are much less noisy than parent order slippage. Metrics like parent order slippage evaluate the joint affect of *all* of the layers of algo decision making, thus catching all sources of noise as well. Data normalization and corrections for market trends can only make so much of a dent in the overall contribution of noise, making large samples necessary for getting meaningful results here.

Since it is unlikely that we can learn much from parent order slippage metrics at our relatively small sample sizes, we design additional metrics that can give us more tailored insights. Some of these metrics target smaller subsets of our algo layers. Some of them target behaviors beyond price (such as volume). Overall, we expect metrics to be less noisy when they 1) assess shorter time scales, 2) are primarily affected by a smaller number of algo layers, and/or 3) measure outcomes we have more direct control over.

Here is a summary graphic for the analyses we currently perform, indicating which algo layers most heavily affect each metric:



3.1 Results for Q4 2021 through Q1 2022

Here are summary tables of our results on these metrics for the time period covering Q4, 2021 and Q1, 2022. Results that did not pass our client privacy or robustness checks are noted as "unstable" [1].

metric	rounding and units	result
slippage vs. arrival, vwap algo	nearest 10 bps	unstable
slippage vs. arrival, vwap algo	nearest 2∗spread	unstable
slippage vs. arrival, Proof algo	nearest 10 bps	unstable
slippage vs. arrival, Proof algo	nearest 2∗spread	unstable
slippage vs. vwap, vwap algo	nearest 1 bps	unstable
slippage vs. vwap, vwap algo	nearest 0.2∗spread	unstable
slippage vs. vwap, Proof algo	nearest 1 bps	unstable
slippage vs. vwap, Proof algo	nearest 0.2∗spread	unstable
distilled impact, vwap algo	nearest 10 bps	unstable
distilled impact, vwap algo	nearest 2∗spread	unstable
distilled impact, Proof algo	nearest 10 bps	unstable
distilled impact, Proof algo	nearest 2∗spread	unstable

Table 3: Parent Order Slippage Metrics

This table sends a clear though disappointing message: we need more data before we can provide meaningful results on parent order slippage metrics. For further discussion, see appendix A.

metric	rounding and units	result
volume curve tracking error	0.02-wide ranges	0.04- 0.06
1min before liquidity seeker trades	nearest 20% of average price movements	0.8
1min after liquidity seeker trades	nearest 20% of average price movements	unstable
impact model features improvement over default	nearest 0.1	+0.1
real market conditions vs. our impact model	tbd once more data	unstable.

Table 4: Behavioral Metrics at 1 minute plus time scales

Some of the metrics in this table involve shorter time scales and/or target fewer layers of our algo logic, and we are starting to get a few meaningful results here. The volume curve tracking error is a measure of the area between our own trading curve and the market's volume, and it is normalized so that 0 represents perfect tracking, and 1 is the worst possible score. This metric is primarily affected by the decisions of our scheduling layer and our tactical layer over timescales of minutes. We view our current score on this metric as acceptable, though we don't have a sense of an industry-wide standard or other reference point. The dynamic volume prediction model that we have baked into our VWAP algo is intended to result in lower scores for this metric than we would expect to achieve by using a typical 20-day or 30-day average curve, and we ultimately expect this to reduce the variance of our slippage around VWAP.

The metrics here that look at price movements in the minutes before and after our liquidity seeker trades are normalized so that a score of 1 represents perfectly average movements, numbers less than 1 represent movements of less than average magnitude, and numbers greater than 1 represent movements of greater than average magnitude. **Scores greater than 1 on these metrics may indicate some level of information leakage. We pleased here to have a stable result that does not indicate this.**

The metric here that compares our impact model features to a default is intended to test that our model has predictive power for the real trading conditions we experience, rather than just in our testing on historical market data. Positive scores here indicate good evidence of predictive power, and hence our stable result of +0.1 shows we are beating our baseline in a stable way.

[1] We note that individual clients receive detailed reports about their own trading data, and can request any raw data or analyses they want when it comes to their own data, as we view them as owners of that data. The robustness and privacy checks here apply solely to our use of aggregate data across clients (though we do insist on giving individual clients the results of robustness checks alongside the metrics we provide, so they have a proper context for interpreting their results).

More details on all of these metrics can be found in appendix B.

order type	Tif	rounding and units	result
Limit	DAY	nearest 0.1∗spread	-0.1
Limit	IOC	nearest 0.1∗spread	-0.3
Pegged	DAY	nearest 0.1∗spread	unstable
Pegged	IOC	nearest 0.1∗Spread	0.0

Table 5: 1 second markouts by order type and tif

To evaluate our router and venue layers, we look at 1 second markouts to mid for each fill. We have set the signs so that positive markouts represent buying lower than the later midpoint, or selling higher. Our markouts are normalized by spread, and then aggregated over fills with weighting by volume. We can see here that our sample sizes are sufficient to get stable results for most categories of 1 second markouts.

Exchange	rounding and units	result
non-IEX	nearest 0.1∗spread	-0.2
IEX	nearest 0.1∗spread	0.0

Table 6: 1 second markouts for Day Limit orders, IEX vs. non-IEX

If we dig in a bit deeper into our Day Limit orders, we can see a stark and stable difference between our outcomes on IEX vs. other exchanges. This is a large component in our improved markouts for passive vs. active orders overall, given how much of our trading occurs on IEX. We believe this difference also validates our IEX-heavy trading distribution. It is worth noting here that **the improvement we see on Day Limit markouts for IEX (+0.2 of the spread) is much larger than the differences in access fees/rebates across exchanges.**

metric	rounding and units	result
percentage of intra-day shares passively filled	nearest 10%	unstable
percentage of shares filled in auctions	nearest 10%	10%

Table 7: Other Metrics on Fills

The percentage of shares that we fill in auctions is controlled solely by our high level scheduling logic, and this is unsurprisingly more stable than our percentage of passive vs. active fills throughout the trading day, which is influenced by decisions at our tactical layer, our router layer, and market dynamics at venues.

More details on markouts and other metrics on fills can be found in appendix C.

metric	rounding and units	result
second	probability of mode frequency by chance	no stable pattern detected
minute mod 10	probability of mode frequency by chance	no stable pattern detected

Table 8: Screens for Unintended Patterns in timing of IOC Limit trades

Finally, we also perform screens to catch any unintended patterns in trades whose timing we directly control: trades where we cross the spread to take. We have added randomization at our tactical layer to avoid, say, always taking in the first second of a minute, or always taking in minute 9 out of a 10-minute interval. Currently, **these screens seem to validate that our current level of randomization suffices stamp out any noticeable patterns of this form.** More details can be found in appendix D.

Summary Overall, this report may feel like a musical festival where the headliner has canceled, as the slippage stats were all over the place and unable to make an appearance. But we don't really see it

that way. We see it as the first iteration of something that is starting small, but has a strong foundation to grow and improve over time. There are encouraging signs here: our impact model is showing positive predictive power under our real trading conditions, and our 1-second markouts validate our IEX-heavy posting logic. There are no troubling signs of information leakage in the price movements around our liquidity seeker's trades, or in the timing of our spread-crossing trades.

It is our intention that our suite of analyses will evolve with our algos, and that as we collect more data, more of these analyses will produce stable and actionable results. Here we have laid a foundation that we feel is fit to build on, and we welcome feedback on what else we should explore!

Appendices

A Slippage Metrics And Robustness Checks

Our slippage metrics in bps are computed by taking the average price we achieved, subtracting the benchmark price, and then dividing by the benchmark price. The sign of the subtraction is flipped for sell orders, so more positive slippage represents a less favorable outcome. We aggregate over parent orders with weighting by notional value.

Our spread-normalized slippage metrics are computed by taking the difference between our average price and the benchmark price, then dividing by the time-weighted average spread for the specified symbol over the 20 calendar days preceding the trade. Aggregations over parent orders are again performed with weighting by notional value.

We view slippage vs. arrival as an uncorrupted metric, in the sense that our own trading decisions in handling the order do not affect the arrival price. However, slippage vs. arrival is notoriously noisy. Slippage vs. vwap is much less noisy, due to the fact that the VWAP benchmark is also affected by the same extraneous market forces that influence our trade prices, allowing some of the noise to cancel out. However, it is a somewhat circular metric, in the sense that our own trading decisions influence the VWAP benchmark.

Due to these challenges with arrival and VWAP as benchmarks, we have developed a third metric, which we call *distilled slippage*. The goal of "distilling" is to control for some of the wider market effects without introducing the full circularity of relying on contemporaneous trading activity in the same symbol that we may heavily influence. We currently do this as follows. For each symbol, we compute its correlations with a curated set of ETFs (around 50 or them) to find one whose price movements are most correlated to the symbol of interest, according to historical market data. Once our symbol has an associated proxy ETF, we can use the (normalized) price movements in this proxy ETF as a model for how the symbol might have behaved without our trading activity. From this model combined with the arrival price in a given symbol, we can estimate what we think the VWAP would have been, if it had been a pure consequence of wider market forces. We can then compare this to the average price we achieved in our trading. For more information, see our most Distilled Impact whitepapers.

To assess the impact of noise on our slippage metrics, we also consider how easily they are swayed by the removal of a small set of trades. In particular, we look at what happens if we remove the best performing trades (the top 5% in notional value) and also what happens if we remove the worst performing trades (the worst 5% in notional value). This shows us the range in the slippage metrics we can induce through small omissions in our underlying data set. If these ranges are dramatically wide, this is a indication that outliers and chance are having a considerable impact on our computed averages.

Unfortunately, none of our slippage metrics passed our robustness and client privacy checks for this data set. The values varied widely as we removed the worst or best orders, and also when we removed single clients. As expected, our distilled impact metric exhibited less noise than our slippage vs. arrival, and more than our slippage vs. VWAP. However, even slippage vs. VWAP was unstable across all dimensions of robustness and privacy checks. In order to release meaningful slippage metric at the parent order level, we will need to collect considerably more data and/or considerably improve our distillation techniques.

B Behavioral Metrics for our Algos

We can generally think of an algo design as driven by a choice of goals, as well as a theory as to a good methodology to achieve those goals. In the case of VWAP, our goals are to reduce slippage vs. vwap, as

well as to reduce variance around slippage vs. vwap, and to do all of this without "gaming" the metric or moving the benchmark considerably through our own behavior. Our current theory is that closely tracking the volume curve is a good way to balance these goals.

If our slippage vs. vwap is low and our variance is low, then we can be confident we are doing a good job on our goals of reducing slippage and variance around this benchmark. But if our slippage is high and/or our variance is high - this alone cannot tell us how much of the problem lies in our theory vs. in our instantiation of that theory. This is where a supplemental metric like the area of our tracking error vs. the volume curves can be clarifying. Looking at our error in tracking the volume curves gives a view into how closely we are instantiating our theory in the case of the vwap algo. This can be useful information as we prioritize research into advancing our theory vs. improving our instantiation. For example, research into strategies like under-trading when spreads are relatively high and over-trading when spreads are low would fall into the category of advancing our theory, while research into improving our prediction model for real-time volume curves would fall into the category of improving our instantiation of our theory. Here we will analyze our tracking error for the VWAP algo, and after this, we will develop some supplemental metrics to evaluate the quality of our theories and instantiations for the Proof algo.

B.1 Volume Curves

We designed our VWAP algo to try to track closely to the day's volume curve. A natural way to measure our effectiveness at this is to compute the area between the cumulative volume curves of our own trading activity and the true volume curves in each symbol and day that we trade. A lower value means there is not much area of discrepancy between the curves, and hence we are tracking closely to the overall market volume. A higher value means there is some considerable deviation. The values are scaled so that the highest possible value is 1.

Here we state the average of these values over VWAP parent orders in Q4 2021 and Q1 2022, with weighting by notional value. For replaces, we treat the replaced order as a new order for this analysis, as changes to the quantity would otherwise distort our tracking of the volume curves. We exclude orders where < 1000 shares were filled, as we expect trading for such small orders to be quite chunky. This will result in high values for this metric for understandable reasons, so we remove these from our data set to get a better understanding of the behavior of the larger orders.

For the purposes of client privacy checks, we rounded the NV-weighted average area between our trading curves and the true volume curves into ranges of the form 0-0.02, 0.02-0.04, 0.04-0.06, etc. With this rounding, the range of **0.04-0.06** was stable across the removal of each individual client from the data set. In the future, we may hope to make these ranges smaller (say 0.01 wide instead of 0.02 wide).

Internal investigation of examples Internally at Proof, we also visually inspect cases of VWAP parent orders over 1000 shares that have a deviation greater than 0.1 in area from the real volume curve for the order's lifetime. Generally, these fall into relatively benign categories: 1. cases where there was a large block of trading that we didn't participate in, resulting in a dramatic jump of the real volume curve, 2. cases where our model at least outperformed the historical curve, even though the deviation for both was high, and 3. cases where the order was canceled or trading was halted before the order's original lifetime completed.

The cases we have seen so far do not appear to be concerning or worthy of specific further investigation. Next time, however, we do plan to update our metrics and screening tools to either exempt orders that are canceled or evaluate them with updated lifetimes that reflect their cancelation. These are a small subset of orders that are unlikely to cause much difference in aggregated stats, but when we screen for visual examples, they are beginning to be prevalent enough to be distracting.

B.2 Evaluating theory vs. practice for the behavior of the Proof algo

Like VWAP, we can think of the Proof algo as having a set of goals and a set of working theories as to how those goals can be achieved. The most natural high level goal is to reduce slippage vs. arrival, as well as reducing the variance around this benchmark. Because slippage vs. arrival is extremely noisy, we can substitute this goal with reducing distilled slippage and the variation around this metric instead.

Each component of the Proof algo can be viewed as instantiating a different theory as to how to reduce distilled impact. The liquidity seeker component operates on the theory that relatively large blocks matched in the dark and traded at the midpoint should not have a significant impact on price. The impact scheduler component operates on the theory that our impact model is an effective model

of our real trading environment, and hence minimizing its predicted impact results in minimizing real impact.

It is likely useful to devise supplemental metrics that can help us directly evaluate either the quality of these theories and/or the quality of our instantiations of them.

B.2.1 Evaluating market behavior around our liquidity seeker's trades

We will evaluate our theory and our implementation of it all together for the liquidity seeker by examining the price dynamics just before and just after our liquidity seeker's trades. There are two phenomena in particular we may want to look for, both representing forms of information leakage. On one hand, we may be concerned about information leakage that occurs *before* a particular block is traded. If someone has effectively "sniffed out" that we are present as a large buyer, for example, we may worry that the price will be pushed up as we go to buy a block, and then will depress again afterwards. In other words, we are worried about buying at local highs and selling at local lows. If present, we would expect this phenomenon to lead to price changes that are above average in magnitude in the periods before and after our blocks, with the market moving away from us as we approach a block trade, and moving into us after.

On the other hand, we may be concerned about information leakage that occurs *from* a particular block being traded. If the block trade itself (or the mechanisms leading up to it) provide an indication of our presence, we might expect a price movement in the aftermath of the trade. In particular, we would expect the market to move away from us after a block trade. Such impact would be undesirable as we will typically have more left to trade.

It is worth noting that both of these hypothetical scenarios involve post-trade movement, but in opposite directions. In some sense, this means *any* post-trade price movement can be taken as possible evidence of information leakage. However, what unites these scenarios and separates them from coincidence is that we would expect a higher *magnitude* of price movement than is typical for this symbol over this day.

So to look for these kind of phenomena, we'll examine price movements for the minute before as well as the minute after each trade our liquidity seeker makes. To examine the minute before, we'll compute the ratio of the NBBO midpoint at the time of trade over the prevailing NBBO midpoint one minute earlier. Then we'll take the absolute value of the logarithm of this value.

To see what this means, we can write the midpoint price at the time of the trade as P_t and the midpoint price one minute earlier as P_{t-1}. We are then viewing the relationship between these prices as $P_t = e^x P_{t-1}$ for some value of x, and we are computing the value of x that makes this true. What's nice about this formulation is that relative price increases and decreases over time can be represented as additional multipliers of e raised to powers, and the powers accumulate additively in the exponent. A price returning to a value then corresponds to these exponents adding up to 0. Taking weighted averages of these exponent values thus makes sense, a feature that is less true if we left out the logarithm. As a basic example, consider a price that moves from \$1 to \$2 over the course of 1 minute, then moves back from \$2 to \$1 over the course of the next minute. In this case, our relative movement in first minute is $\frac{2}{1} = 2$, and our relative movement in the second minute is $\frac{1}{2}$. If we averaged these, we'd get $\frac{5}{4}$. If we take the absolute values of the logarithms, however, we get $\log 2$ and $\log 2$, which average to $\log 2$. For our purposes here, this feels like a better representation, as we do view the price decrease from \$2 to \$1 as equivalent to the increase from \$1 to \$2 if we don't care about the direction of the movement.

Next to provide context, for each symbol/day pair, we will repeat this calculation for each minute of the trading day and take the notional value weighted average of the result. (Here, the notional value weighting refers to the notional value traded in the market for each minute in this day/symbol, not solely our own trading.) This gives us an average value for this calculation for each symbol over the day we were trading. Since we want to get a sense of whether our blocks happened at particularly suspicious moments (moments where prices were moving more than they typically do), we will divide by this daily average value and get a ratio for each of our trades. This ratio will be 1 if the price movement in the minute before our trade was exactly "average" in magnitude for minutes of trading in that symbol on that day. This ratio will be greater than 1 if the minute before our trade was a more drastic price movement than average, and less than 1 if the minute before our trade was a less drastic price movement than average.

Once we have these ratios for each trade that our liquidity seeker made, we can get some indication of our overall performance on this metric by taking a notional value weighted average of these values (weighting each ratio by the notional value of the trade it represents). The nv-weighted average ratio computed by comparing our price movements in the minute before a trade to the day/symbol averages is stable over removing one client if we round to the nearest 20%. The stable rounded value is 0.80. This

indicates that (by this averaging at least), we are trading in relatively stable moments as opposed to relatively volatile ones.

While we were of course hoping *not* to see a value greater than 1 that could indicate information leakage problems, this low value seems a bit strange at first. Why would our liquidity seeker be trading in moments that are meaningfully *less* volatile than average, rather than just average? A simple explanation could be: these trades are less likely to take place during the more volatile periods around the opening and closing auctions. But this doesn't seem to explain the phenomenon for our particular data set. It is important to remember here that our sample of liquidity seeker trades is very small, and it's probably not worthwhile to go too far down a rabbit hole at these data sizes.

We can do the same calculation for the minute *after* our liquidity seeker's trades, though this time the rounded value is not stable over the removals of single client's data. We suspect this is merely due to random chance given the sample size of our liquidity seeker's trades over this time period.

The only conclusion we can really draw here is as follows: **there is currently no strong evidence of prices moving abnormally in the minutes surrounding our liquidity seeker's trades.**

B.2.2 Our price movement model in practice

For our impact scheduler, we can try to evaluate the quality of our model and our instantiation of it in a few different ways. Our model centers on two features: the two pressure gauges we've defined that track spread-crossing and NBBO-joining activities. The model is built on a meaningful (though noisy) relationship between the values of these pressure gauges and price movements over roughly 10-minute timescales as exhibited in historical market data. [For details on our price impact model and it's development, see our whitepaper on the topic: https://www.prooftrading.com/docs/main-algo.pdf]

This prompts the question: does this same relationship seem to hold for times that we are trading? If our real-time trading experience is inconsistent with the historical relationship between the pressure gauges and price movements, this could suggest that our model features are failing to capture some important aspect of our trading that may be leading to impact. Or it could be an indication that our model is not sufficiently robust or stable over time to use in the way that we are using it. In any case, it's definitely something we want to look for!

So we'll examine the day/symbol pairs where we were actively trading, and we'll divide the trading day into 10-minute buckets. In each bucket, we'll compute:

- the value of the spread-crossing volume pressure gauge for the overall market in this symbol and time bucket
- the value of the NBBO-joining pressure gauge for the overall market in this symbol and time bucket
- our measure of the price movement over this symbol and time bucket
- the notional value that we traded in this symbol and time bucket

[Technical Note: we will not separate out flow by algo here, even though we are analyzing the impact model. This is because the low level algo tactics are shared between the VWAP and Proof algos, so the model of our algo's actions is just as valid to analyze over our VWAP trading data, and we can benefit from having the additional sample size.]

The goal here is to study the relationship between the pressure gauges and price movements, to see if what we find here for time periods of our active trading is consistent with what we found when we built our model from historical market data. Our full model has a few more variables: it is separated into three time periods (the first 30 minutes of the trading day, the middle portion, and the last 30 minutes of the trading day), and it also considers the lingering influence of the pressure gauges in the previous 10-minute time period. Supporting these additional variables requires a larger sample size, and so we will put off comparing our trading data to our full model until we have more trading data. Nonetheless, our full model can be used to derive a simpler model over a subset of the variables by averaging over the variables we are not tracking here. Specifically, we will consider only two variables here: the values of our two pressure gauges over the current ten-minute interval.

For each combination of values for our two pressure gauges, our (reduced) model predicts the expected value of the price movement (which may be positive or negative, depending on if we are predicting the price will go up or down.) [Here, we use the term "expected value" in the sense of probability theory, where it represents a probabilistic average over possible values.] A natural question is: are these model expectations generally closer to the observed price movements than a trivial default, like always predicting that the price movement is 0 (i.e. the price doesn't move at all)? To answer this, we'll compute the "mean squared error," treating our model's expected values as predictions for the true values. In other

words, for each time bucket we'll take the model's prediction and the true price movement, subtract them, and then square the result. This squaring means that we will get a positive number when our prediction differs further from the true value, regardless of the direction of the difference. Once we have these squared errors for each time bucket, we'll take a weighted average of them across our data set, where the weights correspond to the notional value we traded in each time bucket. [This basically asserts that we care more about the quality of our model's predictions when we are trading more dollars.] We can then do this same calculation where we always use "0" as the expected price movement. If our model is any good in the situations we're actually trading in, we should see a higher mean squared error when we use the default predictions as opposed to our model's outputs.

It is worth noting that we periodically retrain our model on more recent historical data. In particular, our latest model is trained on data from a time period that overlaps with the time period of trading we are analyzing here. If we wanted to perform the most faithful test we could of the model's behaviour in practice for our trading, we would need to go back and use predictions from whatever iteration of the model was used in production for each time period in our trading data. Instead, we will just universally use the predictions of our latest model, which was trained on data contained in Q1 2022. We do this for two reasons: 1. it is more convenient than tracking and reverting to older models for TCA, and 2. it may give us a clearer a view of the quality of our raw modeling framework, with less interference from varying market conditions. If our model framework is meaningful but the resulting models must be retrained more quickly with changing market conditions, then the more faithful test would show poorer model efficacy, and we might not be able to the see that the problem is the frequency of retraining, rather than something deeper like the feature selection. Frankly, given the large amounts of data that are needed to train robust models at this level, and the stability over time that we saw in model testing and development, we are less worried about time-varying effects at the moment than the question: is there something *different* about the times/ways we are trading as compared to the overall market averages that drive our modeling? This question is perhaps best addressed by testing model-generated predictions for our trading patterns where the model is trained on the *same* time periods. The usual concern with this overlap between trading and testing data is that we could have over-fitting. That's not really a plausible concern in this case though, as the notional value represented by our trading is minuscule compared to the overall market.

There is one more technicality worth noting here. When we train our model, we aggregate over symbols and there are a few normalizations that we do to make this workable. Most, like defining our features in terms of percentages relative to the ADV, are straightforward. But one adjustment is more involved: for each symbol over each week of training data, we normalize the distribution of observed price movements to have a mean of 0 and a standard deviation of 1. When we later use our model for trading, we consult a symbol-specific volatility parameter that allows us to translate the normalized prediction back into the proper scale for this particular symbol. Here we will perform the opposite translation: we will take the raw price movements we observe for the time buckets we traded in, and we will perform this normalization using market data for that symbol over the week so that we can make an apples-to-apples comparison with our model's raw predictions.

Because of this normalization, we actually know what we should expect to see as the mean squared error for the trivial price movement predictions of "0" if our trading data behaves as if it comes from the same distribution as the overall historical training data. We would expect to see an average mean squared error roughly equal to 1. This is because when we have a random process that has been normalized to have mean 0 and standard deviation 1, the mean squared error becomes the variance, which is also 1.

Now that all of those technicalities are out of the way and we have some context for what we should expect to see, let's look at the result! We computed the average mean squared error for the trivial predictions of "0" for all price movement, and then subtracted the average mean squared error for the model predictions. Our hope is to get a value that is meaningfully positive. We rounded this value to the nearest multiple of 0.1 for our client privacy checks, and we did get a stable rounded value of +0.1 across all removals of individual clients. This provides some evidence that our model's predictions do apply in the real situations in which we are trading, at least to the weak extent that they improve upon the trivial default.

One interesting thing we noticed along the way was that the default predictions outperformed our expectation of mean-squared error equal to 1, even though they under-performed our model. These values individually were not stable across client removals, so we do not give the quantitative results here. However, we can say that there is evidence indicating that the distributions we are dealing with are non-Gaussian, and gaining a fuller understanding of these distributions may be a fruitful subject of future research.

B.2.3 Our distribution of market conditions

In translating our core model of how the pressure gauges influence prices into scheduling decisions, we treat the contribution of other market participants to the pressure gauges as being independent from our actions. The interesting question here isn't whether this is false (it is!) but rather, is this false in a detectable and correctable way? We can begin to investigate this by looking at the pressure gauge values around our trading activity *if we subtract out our contribution to these gauges*. Does the distribution of values remaining after we subtract our activity look "normal" in the historical context? Or is it skewed in a way that we can quantify and use to adjust our model to better reflect our live trading experience?

To study this, we compute how the notional value we traded distributes over the possible values of the pressure gauges, and compare this to how the overall market notional value distributes over pressure gauge values in our model's historical training data. As detailed in our whitepaper (https://prooftrading.com/docs/main-algo.pdf), the model ultimately collapses the pressure gauge values into just 5 categories, which are labeled $-2, -1, 0, 1, 2$. For spread-crossing volume pressure (denoted VP), a 0 indicates a relative balance between spread-crossing to trade at the bid vs. spread-crossing to trade at the ask. This is computed by taking a normalized measure of volume at the bid minus volume at the ask, so the positive values 1 and 2 here indicate more trading at the bid than the ask, while the negative values -1 and -2 indicate more trading at the ask than the bid. The 1 and -1 values indicate a relatively mild imbalance, while the 2 and -2 values are catch-all categories that cover all greater imbalances. For NBBO joining pressure (denoted JP), a 0 indicates a relative balance between sizes joining the bid vs. joining the offer. This is computed by taking the sum of joining sizes on the bid minus the sum of joining size on the ask, so the positive values here indicate more joining on the bid side while the negative values indicate more joining on the ask size.

We when initially codified these pressure gauge definitions, we thought it would be simplest to have a consistent form of "take what happens on the bid minus what happens on the ask, and normalize," but there is one very annoying consequence. Positive values of volume pressure should exert a downward pressure on prices, while positive values of joining pressure should exert an upward pressure on prices. So if we want to look at pressure gauge value combinations where the likely price effects are directionally aligned, we need to flip one of their signs.

We can compare the distribution of the pressure gauge values that we experienced in our trading side by side with our historically-developed model. We won't show you this comparison now, as the distribution we experienced while trading was not very stable over client removals in our client privacy checks. This is one indication that we don't really have enough data to deeply study this yet, but we intend to further develop this line of analyses in the future.

C Stats on Individual Fills

In this section, we detail our statistics that have to do with individual fills. This includes breakdowns of our fills across characteristics like passive vs. aggressive, auction vs. intraday, short term markouts, etc. We'll start with some basic breakdowns to provide a high-level picture of our results.

Active vs. passive fills We categorize each fill as "passive" if it has a tif of "Day," and active if it has a tif of "IOC." [Note: this excludes auction fills.] The percentage of fills that traded passively was not stable over the client privacy checks when rounded to the nearest 10%, so we do not report the results here.

Intraday vs. auctions We computed the percentage of the shares that we traded in this period that traded intraday vs. in auctions. Rounded to the nearest 10%, the percentage traded in auctions was 10%, and this was stable through the client privacy checks.

Markouts by tif and order type For each fill, we measure the 1-second markout to mid (i.e. the difference between our execution price and the midpoint price 1 second later). This is computed as the later midpoint minus our trade price when we were buying, and as our trade price minus the later midpoint when we are selling. Thus, positive or neutral markouts are generally preferred to negative ones, particularly for passive fills. [There is some subtlety for aggressive fills, as positive markouts can result from pushing the stock.] We normalize markouts by spread, meaning that we divide each markout by the average spread in that symbol. We aggregate these spread-normalized markouts over fills with weighting by volume.

We calculate aggregate results separately by tif and order type, and round them to the nearest multiple of $0.1 * spread$ for our client privacy checks. For pegged day orders, the results were unstable. For auction orders, the results were also unstable. For the remaining cases, the results were stable and given below:

Tif	Order Type	Spread-Normalized Markout
DAY	Limit	-0.1
IOC	Limit	-0.3
IOC	Pegged	0.00

Table 9: Spread-Normalized 1-Second Markouts

Markouts for day limit fills split by IEX vs. other We also looked at fills with "DAY" as their tif and "Limit" as their order type, and split these into two categories based on whether they occurred at IEX or elsewhere. [Note: these are the fills that result from posting.] We then computed the volume-weighted, spread-normalized markout for each category. For client privacy checks, we rounded these to the nearest $0.1 * spread$, and the results were stable at 0.0 and -0.2 respectively. **This means the *difference* between the average IEX markout and the average non-IEX markout is stable as $0.2 * spread$ (rounded to the nearest $0.1 * spread$).** This is strong evidence that our relatively heavy use of IEX results in favorable outcomes at this level of market microstructure.

D Screening for Unintended Patterns

Finally, we try to find any patterns in our trading behavior that are unintentional. This allows us to investigate how/why they are appearing, and then to change our behavior to mitigate/avoid any patterns we do not think are desirable.

Time patterns Currently we search for patterns in the timing of our trades and orders. Since we increased randomization of our algo tactics somewhat recently, we will limit ourselves here to looking at the data for Q1 2022. In particular, we look at the timing of our IOC Limit trades. These represent trades where we fully control the timing, so we want to see what kind of patterns (if any) might be present in their timestamps. We looked at two different features of these timestamps: which second they occurred within the minute (0 to 59), and which minute they occurred within the hour, modded by 10 (0 to 9). For these calculations, we count trades occurring within the same second for the same order as being part of one trade event. For each timestamp feature, we look at how many times each value occurred for some trade event.

Heavier trading into the close can skew these numbers a bit, as the steep tail of the volume curve will induce a tendency to trade closer to the end of the last 10 minute interval, as well as closer to the end of the last minute. To remove this effect, we remove trades that occurred in the last 10 minutes of the trading day from this analysis.

For the remaining data, our default expectation is that all values are equally likely to occur, due to the randomization we add to our algorithms' tactics. To decide if our data is consistent or inconsistent with this hypothesis, we run what are called "Monte Carlo" simulations. If our data includes N trade events, than each simulation will pick N random values from the default distribution where all values are equally likely. We then compare the frequency of the mode value in our real data with the modes in the simulations. We ask: what's the probability that the mode of such a simulation is at least as frequent as what we see in our real data? This probability is well estimated by running a lot of independent simulations. (In particular, we run 10,000 simulations.) If this probability is very low (say, less than 1%), this suggests that our algo tactics display a tendency towards a particular value for this feature on our flow, rather than being uniformly distributed across all possible value.

Through these methods, we observed no meaningful concentration in the mode for the second values within a minute, and this lack of pattern was stable across removing individual clients. In no case was the mode an event with probability less than 70% of occurring by chance.

For the minute values mod 10, there was no stable pattern, though a lack of pattern was also unstable. This may be an artifact of our very small sample sizes. Altogether, we don't have very strong evidence here that we are falling into noticeable timing patterns, but we will continue to screen for this going forward.