Proof's Public-facing TCA: Results for 2024 Data

Abstract

Here we apply 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 here we can report a few coarse slippage stats on our Proof and Aggressive Algos, as well as short-term markout results. Several analysis designs have also been upgraded since our last public report, and we describe all metrics in detail here so that this report is self-contained. Past versions of this report can be found here. We would like to highlight new normalization techniques we have developed in Section 6 to better understand symbol volatility distributions as a function of time of day, as well as improved timing screens to look for potential sources of information leakage in our algo tactics as detailed in section 7.

1 Overview

In this report, we describe the design of the analyses we perform and provide the updated subset of results that currently pass our robustness checks. Our data set here will be all of Proof's trading activity in the one year period from January 2024 through December 2024.

Client Privacy 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.

2 Overall Activity Stats

Here is a summary of our trading data set (for clients who have not opted out) for the time period from January 2024 through December 2024 (inclusive), for flow traded through our regular algos:

- Traded by our VWAP algo: 4.3 billion notional, 56.7 million shares
- Traded by our Proof algo: 6.0 billion notional, 82.6 million shares
- Traded by our Aggressive algo: 2.5 billion notional, 29.2 million shares
- Traded by our Close algo: 3.2 million notional, 131.6 thousand shares

We note that our close algo was released somewhat near the end of this data collection period, which is why its traded amount is relatively small.

Our Vwap algo traded less than 1% of its notional value and volume in the opening auction, roughly 86% of its notional value and 85% of its volume intraday, and roughly 14% of its notional value and 15% of its volume in the closing auction.

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 time period, roughly 30% of the notional value and 35% of the volume traded by the Proof algo was accomplished through the liquidity seeker component, while roughly 70% of the notional value and 65% of the volume was accomplished through the scheduler component. The scheduler component includes the opening and closing auctions. The opening auction represented less than 1% of the notional value and volume, while the closing auction represented roughly 7% of the notional value and 6% of the volume traded by the Proof algo.

Our Aggressive algo similarly has two components that operate in parallel: a liquidity seeker that employs lower min quantities than the Proof algo version, and a scheduled component that roughly targets a 20% participation rate. In this time period, roughly 36% of the notional value and 33% of the volume traded by the Aggressive algo was accomplished through the liquidity seeker component, while roughly 64% of the notional value and 67% of the volume was accomplished through the scheduler component. The scheduler component includes the opening and closing auctions, though the Aggressive algo traded less than 1% of its notional value and volume in auctions.

2.1 Venue Breakdowns

Let's take a look at how our intra-day trading distributes over venues. 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 can add up to slightly less than 1, and 0.00 values may be displayed when only one of the metrics for a venue rises above our threshold for inclusion.] We will break things out by algo, as well as separating the scheduler and liquidity seeker components where relevant. We will look at the venue breakdowns of the scheduler flows by algo first, then we will look at the breakdowns of liquidity seeker components.

2.1.1 Intra-day Vwap

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

lastMarket	nv	volume
ARCA	0.02	0.02
ASPN	0.01	0.01
BATS	0.02	0.01
BATY	0.01	0.02
EDGX	0.01	0.01
IEXG	0.52	0.50
SGMT	0.01	0.01
UBSA	0.00	0.01
XCHI	0.01	0.00
NASDAQ	0.19	0.19
NYSE	0.19	0.20

Table 1: Venue Breakdown for Vwap Intra-day (units of 0.01 = 1%)

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 continue to find that this results in improved 1 second markouts for our passive fills.

2.1.2 Intra-day Proof Scheduler Component

Here is a breakdown of what percentage of the intra-day flow traded by our Proof algo scheduler component occurred at each venue, in terms of volume and notional value:

lastMarket	nv	volume
ARCA	0.02	0.02
ASPN	0.03	0.03
BATS	0.01	0.01
BATY	0.02	0.03
EDGX	0.01	0.01
IEXG	0.47	0.42
MEMX	0.01	0.01
SGMT	0.00	0.01
XCHI	0.01	0.01
NASDAQ	0.23	0.24
NYSE	0.18	0.21

Table 2: Venue Breakdown for Proof Scheduler Intra-day (units of 0.01 = 1%)

2.1.3 Intra-day Aggressive Scheduler Component

Here is a breakdown of what percentage of the intra-day flow traded by our Aggressive algo scheduler component occurred at each venue, in terms of volume and notional value:

lastMarket	nv	volume
ARCA	0.04	0.06
ASPN	0.01	0.01
BATS	0.04	0.07
BATY	0.02	0.01
EDGA	0.00	0.01
EDGX	0.03	0.04
EPRL	0.01	0.02
IEXG	0.35	0.24
LTSE	0.05	0.03
MEMX	0.02	0.04
XCHI	0.04	0.02
NASDAQ	0.21	0.26
NYSE	0.15	0.16
XPHL	0.01	0.01

Table 3: Venue Breakdown for Aggressive Scheduler Intra-day (units of 0.01 = 1%)

2.1.4 Proof Liquidity Seeker Component

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

lastMarket	nv	volume
BARX	0.12	0.12
BIDS	0.28	0.28
CGXS	0.02	0.02
IEXG	0.20	0.18
INCR	0.13	0.15
LEVL	0.07	0.06
SGMT	0.12	0.11
UBSA	0.05	0.05
XNAS	0.02	0.02

Table 4: Venue Breakdown for Proof Liquidity Seeker Intra-day (units of 0.01 = 1%)

We note that the XNAS flow here represented here is using MELO.

2.1.5 Aggressive Liquidity Seeker Component

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

lastMarket	nv	volume
BARX	0.09	0.11
BIDS	0.14	0.14
CGXS	0.03	0.03
IEXG	0.25	0.24
INCR	0.12	0.11
LEVL	0.15	0.11
SGMT	0.12	0.13
UBSA	0.08	0.09
XNAS	0.02	0.03

Table 5: Venue Breakdown for Aggressive Liquidity Seeker Intra-day (units of 0.01 = 1%)

Our liquidity seekers typically rest 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.

3 Parent Order Slippage Metrics

Next we look at slippage metrics for parent orders. 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 trading day. 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 Distilled Impact whitepapers.

Here are summary tables of our results on these metrics for the time period covering January 2024 through December 2024 (inclusive). Results that did not pass our client privacy or robustness checks are noted as “unstable” [1].

[1] We note that individual clients who have not opted out of analyses may 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).

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	2∗spread
slippage vs. arrival, Aggressive algo	nearest 10 bps	10 bps
slippage vs. arrival, Aggressive algo	nearest 2∗spread	unstable
slippage vs. in limit vwap, Vwap algo	nearest 1 bps	unstable
slippage vs. in limit vwap, Vwap algo	nearest 0.2∗spread	unstable
slippage vs. in limit vwap, Proof algo	nearest 1 bps	unstable
slippage vs. in limit vwap, Proof algo	nearest 0.2∗spread	unstable
slippage vs. in limit vwap, Aggressive algo	nearest 1 bps	unstable
slippage vs. in limit vwap, Aggressive algo	nearest 0.2∗spread	unstable
distilled impact, Vwap algo	nearest 10 bps	10 bps
distilled impact, Vwap algo	nearest 2∗spread	unstable
distilled impact, Proof algo	nearest 10 bps	unstable
distilled impact, Proof algo	nearest 2∗spread	2∗spread
distilled impact, Aggressive algo	nearest 10 bps	10 bps
distilled impact, Aggressive algo	nearest 2∗spread	unstable

Table 6: Parent Order Slippage Metrics

For our current sample sizes and usage patterns, most of these metrics remain unstable. However, we see that the Proof algo is exhibiting average slippage vs. arrival and average distilled slippage that are closer to twice the average spread than it is to zero or four times the average spread. We also see that the Aggressive algo is exhibiting average slippage vs. arrival and average distilled slippage that is closer to 10 bps than to 0 or 20 bps. The Vwap algo is exhibiting average distilled slippage that is closer to 10 bps than to 0 or 20 bps.

4 Volume Curve Tracking

Since the design of our VWAP algo is intended to track the intraday volume curve closely (using our own dynamic volume prediction model), we take a look at how the volume curves of our VWAP trading compare to the market-wide volume curves over the same time periods. We'll compare both to the general market-wide volume curve as well as to an in-limit version that only include market-wide volume within the order's limit price at that time. There are many nuances to this, as orders are frequently modified by clients. A single parent order might have it's algo strategy switched from VWAP to Proof, for example, its quantity updated, and its price limit changed. For our purposes, we will treat a strategy change or a quantity change as the instigation of a new "order" for analyses. We will not treat a limit change as a new order, however, as it does not affect the volume curve tracking goal (though it may constrain our ability to meet that goal if the order becomes limited away). Thus, the units of our analyses here consist of continuous periods in time when an order is assigned to the VWAP strategy with a static quantity. Such units are ended either by the order end time, or the event of a change to the algo strategy or the order quantity. We will only consider order segments that last at least 10 minutes, as shorter segments give us less of a meaningful look at our volume-tracking quality.

For each order time segment lasting at least 10 minutes, we will compare our volume curve to the volume curve of the general market, both unconstrained and limit constrained. We will break the time into minutes and compute the total volume traded by our algo on the order, the total volume traded in the market, and the total in-limit volume traded in the market. From this, we can compute three volume curves: our trading curve, the general market curve, and the in-limit market curve. More precisely, each curve is represented as a series of minute-by-minute fractions of the total volume over the time period that has traded so far. Such fractions are equal to 0 at the beginning of the interval and equal to 1 at the end.

We can then compute the distance between each pair of curves by taking the absolute value of the differences of each minute-by-minute fraction. (For the analysis nerds, this is the $L1$ distance. For the geometry nerds, this is proportional to the area between the curves.) We normalize this distance by dividing by the number of minutes in the time segment. The result can be viewed as an error metric that measures our error in tracking the volume curve. We compute this both for our curve compared to the general market, and for our curve compared to the in-limit market. Once we have an error metric computed for each order segment, we'll average over them with weighting by our trading volume. A perfect tracking of the volume curve would result in a score of 0, while the worst possible score is 1. In our general experience, common scores on this metric may range from 0 to 0.1, depending on the order and market characteristics. Very small orders may trade relatively chunkily (say a round lot here or there), and hence will score somewhat high typically. A big block in a low ADV symbol may also cause a sudden jump in the market volume curve that is hard to anticipate, so such conditions may also cause high scores.

To check stability, we round the resulting volume-weighted average L1-distance to ranges of (0,0.02), (0.02,0.04), (0.04, 0.06), etc. In this data set, the results were unstable, both for our comparison to the general market curves and for our comparison to the in-limit market curves.

5 Short Term Markouts and Passivity Rates

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.

order type	Tif	rounding and units	result
Limit	DAY	nearest 0.1∗spread	−0.2∗spread
Limit	IOC	nearest 0.1∗spread	−0.2∗spread
Pegged	DAY	nearest 0.1∗spread	−0.1∗spread
Pegged	IOC	nearest 0.1∗spread	unstable

Table 7: 1 second markouts by order type and tif

If we dig in a bit deeper into our Day Limit orders, we can see a stable difference between our outcomes on IEX vs. other exchanges. Rounded to the nearest 0.2∗spread (note the slightly wider window needed for stability), we observed a +0.2 spread improvement for Day Limit markouts on IEX vs. other exchanges. We believe this difference validates our IEX-heavy trading distribution, as it is larger than the differences in access fees/rebates across exchanges.

Exchange	rounding and units	result
IEX vs. non-IEX	nearest 0.2∗spread	0.2∗spread

Table 8: 1 second markouts for Day Limit orders, IEX vs. non-IEX

We also look at the passivity rates for intra-day fills achieved by each algo. We calculate passivity rates by taking the total shares filled with "time in force" values (tif) of "DAY" and dividing by that total shares filled with tifs equal to "DAY" or "IOC." It should be noted that this counts midpoint fills obtained by pinging the mid with tif="IOC" as active fills rather than passive. Such fills are a rather small portion of our flow, and thus their classification here is not particularly important. Unfortunately we cannot say anything meaningful about passivity rates for each algo, as none of these are stable over our robustness/privacy checks:

algo	rounding and units	result
Vwap	nearest 10%	unstable
Proof	nearest 10%	unstable
Aggressive	nearest 10%	unstable

Table 9: Percentage of intra-day shares passively filled, by algo

6 A New Metric for Analyzing Liquidity Seeking Fills

We evaluate the quality of liquidity seeker fills by examining the price dynamics just before and just after our liquidity seeker's trades. There are two phenomena in particular we 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 atypical post-trade price movement can be taken as possible evidence of information leakage. But the word "atypical" is crucial here - some price movement is to be expected due to market noise, and how much is a function of parameters such as a time of day and symbol characteristics.

If we can improve our understanding of "typical" price movements, this can help us in isolating any undesirable effects of our own trading activity. Previously, we had taken a rudimentary approach to this, looking at price movements per minute for each symbol/day we were trading, and measuring if the minutes of our liquidity seeking trades represented higher than average levels of price movements. This is rudimentary because it implicitly treats all times of day as independent samples of the same underlying distribution. But we know that price volatility is greatest in the morning around the open, so meaningful nuances in terms of time of day are getting lost here. Another complication is that price trends over a day (a stock generally trending up or generally trending down) will contribute to our sampled price movements, muddying our view of the more localized affects around individual trades that we are trying to see.

Here we will develop a more layered and nuanced approach to modeling typical price movements around daily trends as a function of symbol and time of day. We will construct this approach one layer at a time - accumulating corrections and normalizations intended to clarify our view of the kind of effects we are most interested in. To avoid biasing our metric design with our training data, we will develop our approach on historical market data and finalize our metric design before applying it to analyze our own liquidity seeking fills.

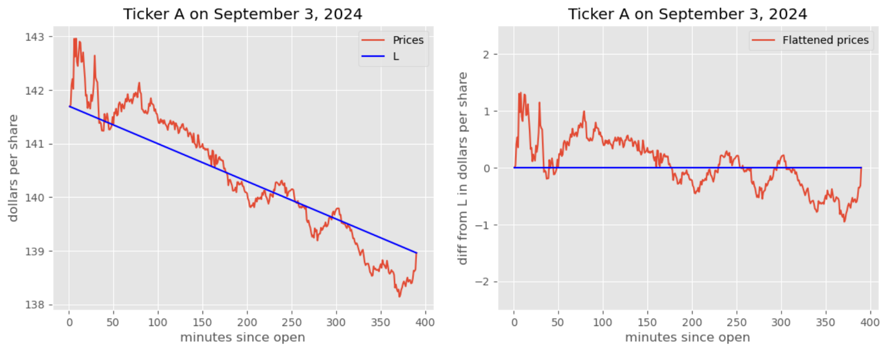
6.1 Pre-processing historical data to account for daily price trends, time-variant volatility patterns, and symbol-specific parameters

Our starting point will be two months of SIP data (September and October 2024), and we'll look at 1000 of the most traded symbols by notional value. For each symbol and each day in our data set, we will break the trading day into minutes: 9:30 am to 9:31 am, 9:31 to 9:32 am, ..., 3:59 pm to 4:00 pm. For each minute, we will take the last trade price occurring as of the end of that minute. Formally:

$$P_{S,d,m} := \text{ last trade price for symbol S on day d, as of the end of minute m.}$$

For minutes that do contain trades, $P_{S,d,m}$ will simply be the last price of a trade occurring within minute m. For minutes that do not contain trades, $P_{S,d,m}$ will be the propogation of the most recent trade price.

From the minute containing the first trade onward, $P_{S,d,\cdot}$ as a function of m will be a time series of prices for a particular symbol over a particular day. We let m_0 be the first minute for which this is defined and m_T be the last. We can define a basic daily trend from this time series by defining $L_{S,d}$ to be straight line that connects P_{S,d,m_0} to P_{S,d,m_T}. We can then subtract the daily trend line from each data point to obtain a new time series. This new time series of differences (relative to the linear trend) will always begin and end at 0 by construction, and we call the this "flattened" price series. Intuitively, this corresponds to taking our raw price series and tilting it so that the trend line flattens to become the new horizontal axis:



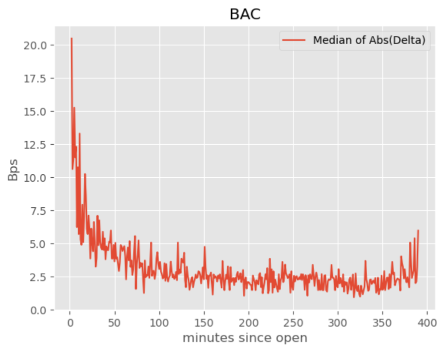
We will let $F_{S,d,m}$ represent the flattened price for symbol S on day d as of the end of minute m. Since we are interested in aggregating data across symbols, we will need to normalize our flattened prices into more relative, comparable terms. For this, we will compute differences between flattened prices from minute to minute, and then divide by the first price so that we are measuring relative movements vs. a fixed benchmark:

$$\Delta_{S,d,m} := \frac{F_{S,d,m} - F_{S,d,m-1}}{P_{S,d,m_0}}.$$

We will look at Δ values in units of basis points (i.e. multiplying the above by 10000) to make the scale convenient for plotting and interpreting.

We can imagine that $\Delta_{S,d,m}$ is sampled from a family of probability distributions that do not change *too* drastically as the indexing parameters S, d, and m vary. In particular, we might imagine that things are similar day to day and in similar symbols, and things change as a function of m in a somewhat smooth way (e.g. starting with a high variance early in the day and settling into a lower variance as the day progresses). To get a sense of this, we'll start by looking at one symbol S and grouping our $\Delta_{S,d,m}$ values by m as d ranges over our two month data set. For each m group, we'll compute the median of the absolute value of $\Delta_{S,d,m}$ to get a sense of the variance of Δ as a function of m.

Here's a graph of these medians for the symbol BAC, as m ranges from over the trading day:

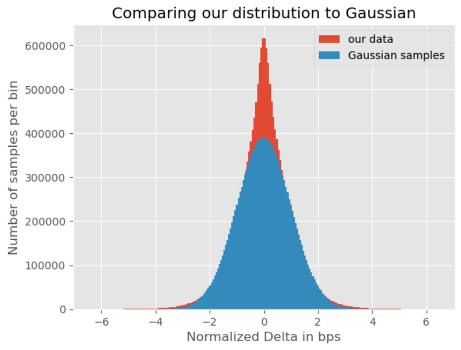


We see here roughly what we would expect - higher variance in the morning, settling in for the middle of the day, and rising again a bit into the close. We also see the remaining challenge of noise: the "wiggles" that we see here are generally an indication that our sample size per stock per minute over a two month period is too low to directly get a very smooth curve as a function of m. We'll return to this issue in a bit.

But for now, we are also interested in the shape of the distribution of Δ values *within* each minute grouping. Studying this will be better if we can normalize data and throw more of it together to increase our sample size. Doing this, of course, will implicitly introduce further assumptions. In particular, let's imagine that the general shape of the distribution is the same over symbols and over minutes, *once we have accounted for how the variance changes as a function of the symbol S and the minute m.* Operating under this assumption, we can compute the empirical mean and variance for the $\Delta_{S,\cdot,m}$ values for each S and m, as the value of d varies over the days of our data set. We can then normalize our Δ values within each group by subtracting the mean and dividing by the standard deviation, and we'll denote the resulting adjusted values by $\widetilde{\Delta}_{S,d,m}$.

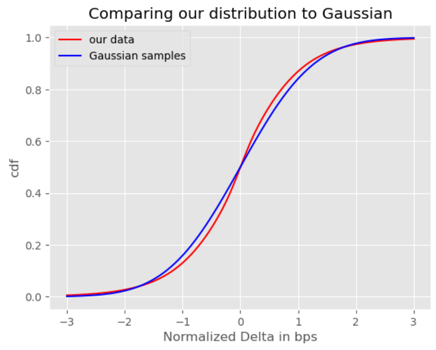
6.2 Understanding the distributional shape after normalizations

After these normalizations, we'll throw all of $\widetilde{\Delta}$ values together and treat them as independent samples from a single distribution (which should be mean=0, variance = 1 by construction). There are a few different ways we can visualize this distribution to get a sense of its shape. We can divide values into bins and draw a histogram:



We have included here the histogram that a normal Gaussian would generate, as a reference. Our distribution looks more concentrated around the middle, but also slightly heavier in the tails.

A downside of histograms is that they depend on the choice of bins. A smoother representation can be obtained by plotting the cumulative density function (cdf). Here is what our cdf looks like versus a normal Gaussian:



One way to measure similarity of two distributions $\mathbb{D}_1$, $\mathbb{D}_2$ is to take the maximum gap between two cdfs (e.g. the Kolmogorov-Smirnov test). Formally, this corresponds to:

$$\max_x \left| \mathbb{P}\left[y_1 \leq x \text{ where } y_1 \leftarrow \mathbb{D}_1\right] - \mathbb{P}\left[y_2 \leq x \text{ where } y_2 \leftarrow \mathbb{D}_2\right] \right|$$

In our case, we have a maximum gap of 0.043, which is much greater than we would expect if our distribution was truly a normal Gaussian at this sample size. To gauge this, we can resample the Gaussian distribution in a Monte Carlo fashion at this sample size, and compute max gaps between Gaussian samples. Typical values obtained in Monte Carlo experiments are about 100 times smaller (e.g. in the range 0.0002-0.0004). The deviation from Gaussian exhibited by our data is apparently much more than is explained by sampling coincidence, and thus we have a meaningfully different shape to our distribution.

6.3 Sample size and robustness checks over historical market data

Our shape also appears to be reasonably robust over time periods and over different samplings of symbol and minute combinations. To spot check robustness over time, we collected the same data for the two month period covering July and August 2024 and compared the resulting cdf of the $\widetilde{\Delta}$ values to our cdf from the September/October data. The maximum gap was 0.0034, an order of magnitude smaller than the gap between our cdf and a Gaussian distribution.

Next, we experimented with randomly sub-sampling from our data to get a sense of what sample size is needed to achieve a clear view of our distributional shape. Our total number of data points is about 17 million. For various sample sizes, we sub-sampled our data points uniformly at random and computed the max gap of the resulting cdf from the reference cdf computed over our entire data set. Here are the results:

sample size	max cdf gap
1,000,000	0.0007
100,000	0.0015
10,000	0.0078
1,000	0.0343
100	0.1124

Table 10: Max cdf gap as a function of sample size

This suggests that sample sizes of around 100,000 or even around 10,000 can give us a meaningful view of our distribution here. But naturally, the samples we get from real trading data will not be distributed uniformly over symbols and minutes of the trading day. The goal of all the pre-processing and normalizations we have done to this point is to account for that, so we should hope at this point that our distribution of $\widetilde{\Delta}$ looks quite similar regardless of which symbols and minutes we sample. We can spot check this by limiting our data to certain symbols and/or certain minutes of the day, sub-sampling again, and comparing the resulting cdfs to our reference cdf computed on the full data set. For example, we tried restricting our data to an arbitrary set of 30 symbols and a period of about 50 minutes in the middle of the day and sub-sampling 10,000 data points, and the max cdf gap compared to our reference was 0.0083. We also tried restricting to a disjoint set of 50 symbols and the period covering the first 30 minutes of the day, still sub-sampling 10,000 data points, and the max cdf gap compared to our reference was 0.0135. Sub-sampling another disjoint set 30 symbols for just the last 30 minutes of the day resulted in a max cdf gap of 0.0152. Overall, these results give us some indications that our pre-processing and normalization approaches are working to give us reasonably robust and distinctly non-Gaussian distributions.

We might also wonder: do we really need two months of data in order to compute workable standard deviations for each $\Delta_{S,\cdot,m}$ group in order to compute $\widetilde{\Delta}$ values? To explore this, we took data from the single month of November and computed $\widetilde{\Delta}$ values using empirical standard deviations computed just within this month. Comparing the resulting cdf to the one computed from September/October data, we found a max gap of 0.0071. This suggests that normalizing data on a month-to-month basis is reasonable.

6.4 Applying this to study our own liquidity seeking fills

For TCA purposes, month-to-month stats might be useful once we have more trading data, but for now we'll normalize and split our analyses by quarter, as this seems like a suitable way of aggregating enough samples of our trading without conflating over wildly disparate market conditions.

With all of this in place, we are ready to evaluate whether our liquidity seeker fills occur at times when price movements confirm to typical distributions. To look for atypical magnitude of movement, we'll compute $\widetilde{\Delta}$ values for the minutes before and after our liquidity seeker fills. We'll then take the median of the absolute value for each of the before and after $\widetilde{\Delta}$ distributions, and compare to one hundred Monte Carlo samples of the same size from the $\widetilde{\Delta}$ computed from broader historical market data over the same quarter. We'll compute empirical standard deviations by quarter for each symbol/minute pair to obtain means and standard deviations for the translation from Δ to $\widetilde{\Delta}$ for each data point. For our broader historical market data, we will use the top 1000 traded symbols by notional value over the first month of the quarter as our symbol set. We're interested in seeing if the medians we find for our before and after trading distributions are *larger* than the typical values we obtain for medians in the MonteCarlo sampling. We will define a before or after distribution median as "suspiciously large" if it is larger than 5% of the medians we obtain in our MonteCarlo experiments.

For our Q1 data, the distribution of $\widetilde{\Delta}$ values for the minute preceding our liquidity seeking trades *is not* suspiciously large, and this holds robustly across our client privacy checks. The status of the distribution of $\widetilde{\Delta}$ values for the minute following our liquidity seeking trades, however, is unstable.

For our Q2 data, the distribution of $\widetilde{\Delta}$ values for the minute preceding our liquidity seeking trades again *is not* suspiciously large, but the distribution for the minute *after* our liquidity seeking trades *is* suspiciously large.

For our Q3 data, the distribution of $\widetilde{\Delta}$ values for the minute preceding our liquidity seeking trades *is not* suspiciously large, and the status of the distribution for the minute *after* our liquidity seeking trading is unstable.

For our Q4 data, the distribution of $\widetilde{\Delta}$ values for the minute preceding our liquidity seeking trades *is not* suspciously large, but the distribution for the minute *after* our liquidity seeking trades *is* suspiciously large.

Since we have two quarters (Q2 and Q4) with a stable finding of the suspiciously large distributions *after* our liquidity seeking trades, we do suspect that the result is meaningful. It is not immediately clear what this means in terms of information leakage and impact, but it is perhaps an intriguing clue as to where we can look to better understand our liquidity seeking behavior and its consequences, in the hopes of finding substantial improvements that we could make.

6.4.1 Looking for correlation between order side and price movement direction

We can also look for correlation between the sign of the $\widetilde{\Delta}$ values before and after our trades and the side of the underlying orders. Since side and sign are both binary variables, we can start by simply counting combinations. More precisely, we classify each minute before a liquidity seeking fill as either p (for positive) or n (for negative), based on the sign of the corresponding $\widetilde{\Delta}$ value. We then count how many buy orders are p, how many buy orders are n, how many sell orders are p, and how many sell orders are n. We'll denote these counts by B_p, B_n, S_p, and S_n respectively.

An odds ratio can then be computed as:

$$O_{before} := \frac{\left(\frac{S_n}{S_p}\right)}{\left(\frac{B_n}{B_p}\right)}$$

A value greater than 1 here, for example, would indicate that sell orders exhibit more of a tendency to be preceded by negative price movements than buy orders do. In our case, we find the opposite. The fact that O_{before} is less than 1 is stable across our client privacy checks. We similarly find that O_{after} is less than 1 for our liquidity seeking fills, and this is also stable across our client privacy checks.

This suggests that our buy fills tend to happen in the middle of negative price movements, and our sell fills tend to happen in the middle of positive price movements. It is important to remember though, that these are positive and negative values of $\widetilde{\Delta}$, which are adjusted movements rather than raw movements.

6.5 Concluding thoughts on liquidity seeking fill analyses

We are not yet in a position to draw a firm conclusion over what this all of this means for our data. The odds ratios, for example, could mean that our buy orders are being hit most when sellers are reacting to negative price movements. It could be that somehow our price adjustments (removing the overall daily trend as a linear function, normalizing by symbol/minute combination over a quarter of data) are contributing to the effect.

It may not feel satisfying yet, but overall we are encouraged that the normalizations and analyses we are doing here to investigate the nature of our liquidity seeking fills appear to be capable of:

1. capturing robust general trends in historical market data
2. exhibiting meaningful and stable results on our current trading sample sizes
3. providing clues as to where to look for reducing our information leakage and/or improving our impact models.

All of these items are threads we are continuing to unravel. For 1. in particular, we have additionally uncovered some interesting market-wide phenomena by looking at $\widetilde{\Delta}$ distributions under symbol clustering. We will shortly be releasing a further writeup of those results.

7 Timing Pattern Screens

Timing Pattern Screens 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 (e.g. we randomize interval sizes).

We will perform screens for timing patterns at both the level of seconds and the level of minutes. For calculations looking at seconds, we count trades occurring within the same second for the same order as being part of one trade event. For calculations looking at minutes, we count trades occurring within the same minute for the same order as part of one trade event.

We then look at how the second value of our second-level trade events distributes over the range from 0 to 59, and how the minute value of our minute-level trade events distributes modulo 10 (we are less worried about patterns at longer ranges of minutes because these are greater than the time scales that our tactical algo layer typically operates on). If we look holistically across orders and our trading decisions are sufficiently randomized, we might expect a roughly uniform distribution over the possible values. However, there are few phenomena that can understandably skew the results. One is that heavy trading into the close may tilt things slightly toward the later second and minute values. For this reason, we remove the last 30 minutes of the trading day from our timing analyses here.

Another is that when many orders arrive at once, their early trades may be correlated in timing. Thus, a skewed distribution for second values over all trades does not necessarily mean we have insufficient randomization in our trading behavior. Conversely, it's possible to have insufficient randomization even if the overall trade distribution is relatively uniform! We might imagine a hypothetical where trade events for a single order happen with predictable timing, but holistically orders arrive and start at random times, thereby making the overall distribution close to uniform.

For these reasons, we do some pre-processing that is order-specific. For each parent order, we look at the second values of the trade events, and we count how many collisions of values we have. For example, if a parent order has five trade events with second values of 2, 15, 2, 57, and 33, then we have one collision (one pair of values that are the same). We then take the number of collisions for each parent order and sum these up over our set of parent orders. We can then compare this to what we would expect to see if the values were uniformly distributed. We note that orders arriving at the same time will no longer cause a skew on this metric, as we only look for collisions of values *within* a single parent order. But if our trading logic had a tendency to concentrate trades in certain values, we would expect to get more collisions than a uniform distribution on this metric.

We might also be concerned about patterns in the gaps of time between consecutive trading events. Imagine, for example, that we get into a state where our algo is taking liquidity every 10 seconds like clockwork. Maybe an order crosses the spread at 10:02:03 am for example, and then at 10:02:13, and then at 10:02:23, and so on. If this continues for several minutes, we might detect higher collisions through the prevalance of second values 3, 13, 23, etc. However, this kind of pattern would not yield higher collisions modulo 60 if the frequency is something like 7 seconds (coprime to 60), instead of something that shares common factors with the modulus. We can try to detect such patterns more directly by looking at the gaps between consecutive trade events and seeing if these exhibit a higher rate of collisions than would be expected. But the "expected" distribution here is a bit tricky to define. A uniform distribution over gap values would not be expected, for example, as larger orders that take many trades to complete would tend to have smaller rather than larger gaps, and this alone does not mean there is a problem with randomization. We should also expect that time gaps will naturally get smaller during times of heavier trading across the market, and correcting for this may be subtle. We are developing much more sophisticated pattern screens as part of our ongoing information leakage research, so we intend to address these scenarios in a satisfying way in the future. But for now, we will stick with looking at collision rates for trade events within a parent order, computed in terms of seconds mod 60 and minutes mod 10.

For minute values modulo 10, we *do not* see more collisions than the uniform distribution would produce on this data set. In fact, we see fewer collisions, and this finding is robust across our client privacy checks. This suggests that our randomization exhibits a small bias against repeating second values that have recently appeared, which does not seem concerning.

For second values modulo 60, the relationship of the collision rate to what the uniform distribution would produce on this data set was unstable across our client privacy checks. We investigated the source of this instability by isolating high collision rate examples in our data, and were able to add randomization to address edge case scenarios so that we expect a stable result of lower collision rates going forward. Though we are working on the design of much more sophisticated pattern screens as part of our ongoing research efforts into market-wide mechanics of information-leakage, we are encouraged that our current collision screens are capable of catching edge case behaviors and enabling us to pro-actively fix them.

8 Future Work and Concluding Thoughts

We note that there are two analyses done in previous reports that we decided not to run this time. One is an analysis of how our volume curve tracking compares to a heuristic simulation of what may have happened if we had used historical curves instead of our dynamic volume prediction. This simulation is a little questionable, as lower level algo tactics and real-time trading conditions induce significant variation independent of volume prediction. Since our volume curve tracking metric was unstable anyway, we didn't expect to learn anything from this comparison on this round of data analysis, so we skipped running it. We also skipped running an analysis on our impact model predictions, restricted to the real days/symbols we were trading. The reason for this is that as our flow gets bigger, the set of days/symbols we are trading is broader, and is expected to converge to the sort of general distribution of days/symbols that form the training set for the impact model's development. At this point, the check we do in model training to see that the model performs better than the baseline is a better check than what we used to do in TCA, so we have retired that metric for now. In the coming months, we plan to do some iterative research on improving our impact model, and we expect our whitepaper on those results to be more meaningful than what we can currently learn from TCA about our impact model at our current sample sizes.

We have already highlighted two places in this report where we expect gains from further research: one is continuing to explore the kind of distributional normalizations we developed in Section 6, and the other is improving our capability to screen for timing patterns and related information leakage, as discussed in Section 7. But there are two other ideas we have had that we would also like to mention. One is related to volume curves: the current design of our volume prediction model and its incorporation into our VWAP algo works most seamlessly when an order is not being constrained by a price limit. To some extent, this is unavoidable: we can't know ahead of time what limits are clients will use, so we can't exactly incorporate them into our model training process. Except... maybe we can? Perhaps there are patterns to what times of day, or what kind of symbols, or what kinds of market conditions tend to lead to VWAP orders being limit constrained in the way our clients trade. If some patterns are evident across clients and somewhat stable over time, we could try to predict the shape of the in-limit curve rather than only predicting the general curve. We did some preliminary analyses on this with this data set, but the set of meaningfully constrained vwap orders was too small to exhibit any stable patterns.

Another intriguing possibility is something that occurred to us while thinking about the challenge of evaluating our parent order slippage. Currently, we use tools like our distilled impact metric to try to reduce noise by adjusting for contemporaneous and extraneous market conditions, but we do not use any kind of pretrade model to try to adjust for "expected" impact as a function of order characteristics. This is because pretrade models tend to ... well... suck. We tried to build one awhile ago using historical TAQ data, and while it was an interesting process that gave us a better sense of the nature of market noise, it didn't yield a model of the quality one would want in order to plug it confidently into TCA processes. Pretrade models offered as a service from other entities are not ideal for use either, as they often don't come with full disclosure on how they work or were developed.

Another limitation of using a pretrade model and a contemporaneous market correction to separately subtract terms from raw slippage is that this structure assumes an independent, linear relationship between the effects of order characteristics and the effects of extraneous market activity during the lifetime of the order. This assumption is probably false! We would probably be better off trying to build a single model of the kind of distribution of slippage we see as a joint function of order characteristics and contemporaneous market conditions. This could allow non-linear relationships and perhaps fit reality far better. We plan to spend some of our research time going forward on investigating such a joint model in addition to investigating pre-trade models and post-trade metrics separately.