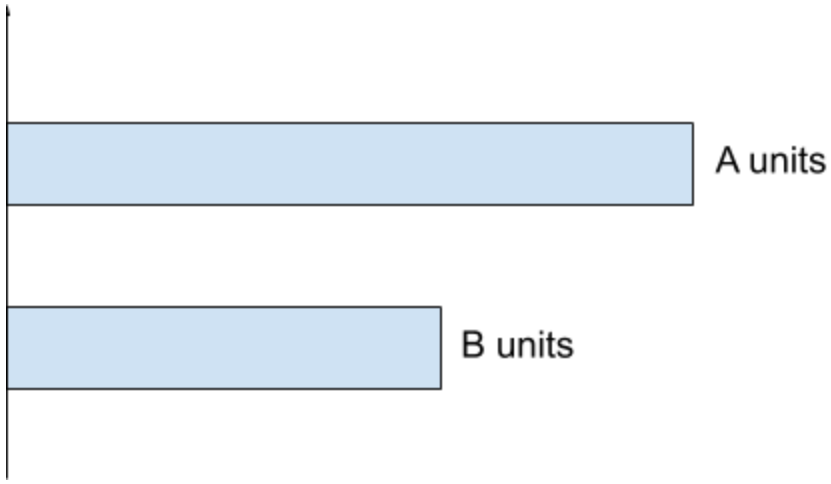
Book Pressure

Author: Pontus Ahlqvist

We derive, using a probabilistic arrival model of market orders, a notion of book pressure that differs (sometimes quite substantially) from the usual version of log(bid volume / ask volume).

Suppose that the orderbook is fixed with A units on the ask and B units on the bid:



Furthermore, suppose that market orders arrive to either buy or sell and that the volumes of these orders are fixed at exactly one unit. Let's model these orders as arriving in such a way that the inter-arrival times between two buy orders are iid exponential (and similar for the inter-arrival times between two sell orders). This will guarantee that the number of orders of either type after a period of time follows a Poisson random variable.

$$\begin{aligned} &\text{Buy Orders:} && \Delta t \sim Exp(\lambda_B) && \rightarrow && N_{Buy}(t) \sim Poisson(\lambda_B t) \\ &\text{Sell Orders:} && \Delta t \sim Exp(\lambda_S) && \rightarrow && N_{Sell}(t) \sim Poisson(\lambda_S t) \end{aligned}$$

Note that the two parameters, λ_B and λ_S are distinct. In other words, the two types of orders may arrive with different rates.

Since the orderbook is fixed and the orders that are arriving are all removing existing limit orders, it's just a matter of time before either of the two levels are taken out. What we'd like to know is which one is more likely to be taken out first (and with what probability).[1]

[1] Note that if the two arrival rates are sufficiently different, it may very well be the case that the level with the largest available volume could still be more likely to be taken out first. In other words, the naive notion of book pressure may not always present the full picture. Instead we will need to also model the arrivals of the orders (which we're doing here).

Before we answer this question, let's consider a slightly more basic question: At what time do we expect the bid to be taken out (and similarly for the ask)? Of course, the time at which this occurs will be a random variable since it depends on the arrival of market orders, but we should nonetheless be able to compute a probability distribution describing this time.

Since each inter-arrival time is iid exponential, we are really asking about the probability distribution of the sum of a set of these random variables. In particular if there are B units available on the bid, we would like to know the probability distribution of $T_{Bid\ Taken\ Out}$ where

$$T_{Bid\ Taken\ Out} = T_1 + T_2 + \ldots + T_B$$

It's well known that the sum of a set of iid exponential random variables follows an Erlang distribution (reference). In other words, the time at which the bid is taken out follows an Erlang distribution:

$$T_{Bid\ Taken\ Out} \sim {\lambda_S}^B t^{B-1} e^{-\lambda_S t}/(B-1)!$$

Note that the distribution for when the bid is taken out depends on the arrival of market sell orders λ_S and the size available on the bid, B which makes sense. Similarly, for the ask we have:

$$T_{Ask\ Taken\ Out} \sim {\lambda_B}^A t^{A-1} e^{-\lambda_B t}/(A-1)!$$

The probability that the bid is taken out before the ask is then:

$$p(Bid\ taken\ out\ first) = \int_0^\infty \int_t^\infty dt\, d\tau\ {\lambda_S}^B t^{B-1} e^{-\lambda_S t}/(B-1)!\ \cdot {\lambda_B}^A \tau^{A-1} e^{-\lambda_B \tau}/(A-1)!$$

In order to compute this integral, let's first move some stuff out of there:

$$p = \frac{{\lambda_S}^B {\lambda_B}^A}{(B-1)!(A-1)!} \int_0^\infty \int_t^\infty dt\, d\tau\, t^{B-1} e^{-\lambda_S t}\ \cdot \tau^{A-1} e^{-\lambda_B \tau}$$

To help with this, let's recall that

$$\int x^k e^{-\lambda x}\, dx = (-1)^{k+1} \frac{\partial^k}{\partial \lambda^k} \left(\frac{1}{\lambda} e^{-\lambda x}\right)$$

This means that the expression above can be integrated once to get

$$p = \frac{{\lambda_S}^B {\lambda_B}^A}{(B-1)!(A-1)!} \int_0^\infty dt\, t^{B-1} e^{-\lambda_S t}\ \cdot (-1)^{A-1} \frac{\partial^{A-1}}{\partial {\lambda_B}^{A-1}} \left(\frac{1}{\lambda_B} e^{-\lambda_B t}\right)$$

We can pull the partial derivatives out of the integral to get

$$p = \frac{{\lambda_S}^B {\lambda_B}^A}{(B-1)!(A-1)!} \frac{\partial^{A-1}}{\partial {\lambda_B}^{A-1}} \{ (-1)^{A-1} \frac{1}{\lambda_B} \int_0^\infty dt\, t^{B-1} \cdot e^{-(\lambda_S+\lambda_B)t} \}$$

This last integral is of the same form and can thus be evaluated (here we've defined $\Lambda = \lambda_S + \lambda_B$ for simplicity:

$$p = \frac{{\lambda_S}^B {\lambda_B}^A}{(B-1)!(A-1)!} \frac{\partial^{A-1}}{\partial {\lambda_B}^{A-1}} \{ (-1)^{A-1} \frac{1}{\lambda_B} (-1)^{B-1} \frac{\partial^{B-1}}{\partial \Lambda^{B-1}} [\frac{1}{\Lambda}] \}$$

The inner derivative can easily be computed to yield

$$p = \frac{{\lambda_S}^B {\lambda_B}^A}{(B-1)!(A-1)!} (-1)^{A-1}(-1)^{B-1} \frac{\partial^{A-1}}{\partial {\lambda_B}^{A-1}} \{ \frac{1}{\lambda_B} (-1)^{B-1}(B-1)! \frac{1}{\Lambda^B} \}$$

Putting the definition for Λ back in, we then finally arrive at

$$p = \frac{{\lambda_S}^B {\lambda_B}^A}{(A-1)!} (-1)^{A-1} \frac{\partial^{A-1}}{\partial {\lambda_B}^{A-1}} \{ \frac{1}{\lambda_B} \frac{1}{(\lambda_B+\lambda_S)^B} \}$$

Let's take a look at the derivative term since it's a little tricky to evaluate (it turns out to be closely related to Pascal's triangle.) In order to attack this derivative, let's write things in a less cluttered way by focusing on the following:

$$\frac{\partial^k}{\partial x^k} \{ \frac{1}{x^n} \frac{1}{(x+y)^m} \}$$

Let's call the function inside the derivative here, $f_{n,m}(x,y)$. It's simple to show that

$$\frac{\partial}{\partial x} f_{n,m}(x, y) = -n f_{n+1,m} - m f_{n,m+1}$$

Perhaps you can see Pascals' triangle in this expression? Basically, you're summing the prior two levels. The end result is the following sum[2]:

$$\frac{\partial^k}{\partial x^k} f_{n,m}(x, y) = \frac{(-1)^k}{(n-1)!(m-1)!} \sum_{i=0}^{k} (n+k-1-i)!(m-1+i)!\, f_{n+k-i,\, m+i}(x, y) \cdot \frac{k!}{i!(k-i)!}$$

We can now go back to our original expression (using $n = 1,\ m = B,\ k = A-1$):

$$p = \frac{{\lambda_S}^B {\lambda_B}^A}{(A-1)!} (-1)^{A-1} \frac{(-1)^{A-1}}{(B-1)!} \sum_{i=0}^{A-1} (A-1-i)!(B-1+i)! \frac{1}{{\lambda_B}^{A-i}} \frac{1}{(\lambda_B+\lambda_S)^{B+i}} \cdot \frac{(A-1)!}{i!(A-1-i)!}$$

Simplifying this a bit, we find

[2] As a sanity check, you can work this out for the two cases above, k=0 and k=1.

$$p = \frac{{\lambda_S}^B {\lambda_B}^A}{(B-1)!} \sum_{i=0}^{A-1} \frac{(B-1+i)!}{i!} \frac{1}{{\lambda_B}^{A-i}} \frac{1}{(\lambda_B+\lambda_S)^{B+i}}$$

We can simplify this a bit further by pulling out the lambdas from the sum:

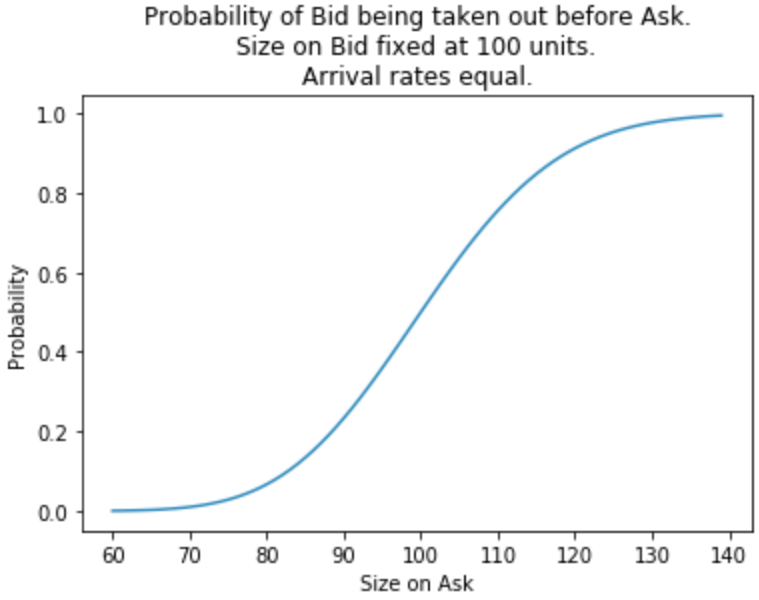
$$p = \left(\frac{\lambda_S}{\lambda_B+\lambda_S}\right)^B \sum_{i=0}^{A-1} \frac{(B-1+i)!}{i!(B-1)!} \left(\frac{\lambda_B}{\lambda_B+\lambda_S}\right)^i$$

If we define the ratio $\gamma := \lambda_B/(\lambda_B + \lambda_S)$, we can write this succinctly as

$$p = (1-\gamma)^B \sum_{i=0}^{A-1} \frac{(B-1+i)!}{(B-1)!} \frac{\gamma^i}{i!}$$

This expression can be written in terms of hypergeometric functions, but that's not particularly helpful in our case.

Here's a plot of what this looks like for equal arrival rates as a function of the size on the ask (bid size fixed at 100 units):



Sanity Check

As a sanity check, let's compute this for the case $\lambda_B = \lambda_S$ and $A = B$. In this case, since everything is symmetric, we would expect the probability of the bid being taken out before the ask to be exactly 50%. To avoid clutter, let's just call the two rates $\lambda_A = \lambda_B \to \lambda$ and the two volumes $A = B \to V$.

In this case, $\gamma = 1/2$, so

$$p = \frac{1}{2^V} \sum_{i=0}^{V-1} \frac{(V-1+i)!}{i!\,(V-1)!} \frac{1}{2^i}$$

This sum can be evaluated explicitly to give

$$p = \frac{1}{2^V} 2^{V-1} = 1/2$$

In other words, if the rates are the same and the volumes are the same, neither level is more likely to be taken out first. This makes sense and gives us faith in the expression above.

Generalizing to Arbitrary Order Sizes

The question is now how we remove the assumption of equal order sizes. In reality, each order does not have the same size but rather the size of the orders follows some type of distribution (possibly something like a log-normal distribution). The question of how long one would need to wait before the level is taken out is then not as simple as just asking what the distribution is of a sum of a fixed number of exponential variables. Rather, the number of such variables is itself a random variable since it's not a priori clear how many orders will be necessary in order to take out a level.

Under the assumption that each order's size is iid, we ought to first ask how many orders will be necessary to take out a level. This is then a random variable whose distribution we can determine. Then, once we have that count, we can then ask about the conditional distribution of the total time necessary to take out that level. We thus get a chain of distributions:

$$p(t) = \sum_{k=0}^{\infty} p(t \mid k \text{ orders}) \cdot p(k \text{ orders})$$

The first of these factors is what we have already worked with above: the Erlang distribution. It's the second factor that's new. In the analysis above, we knew already that a size of A on the ask would require precisely A orders to arrive, but here the story is more complicated.

$$p(t) \sim \sum_{k=1}^{\infty} \lambda^k t^{k-1} e^{-\lambda t}/(k-1)! \cdot p(k \mid V)$$

Here the exponents are no longer related to the volume available but rather to the number of orders that must arrive. This latter variable is then governed by the distribution in the second factor, and this distribution *is* conditioned on the volume available. At the end of the day, the probability that we're ultimately interested in (bid taken out before ask or vice versa) will be an

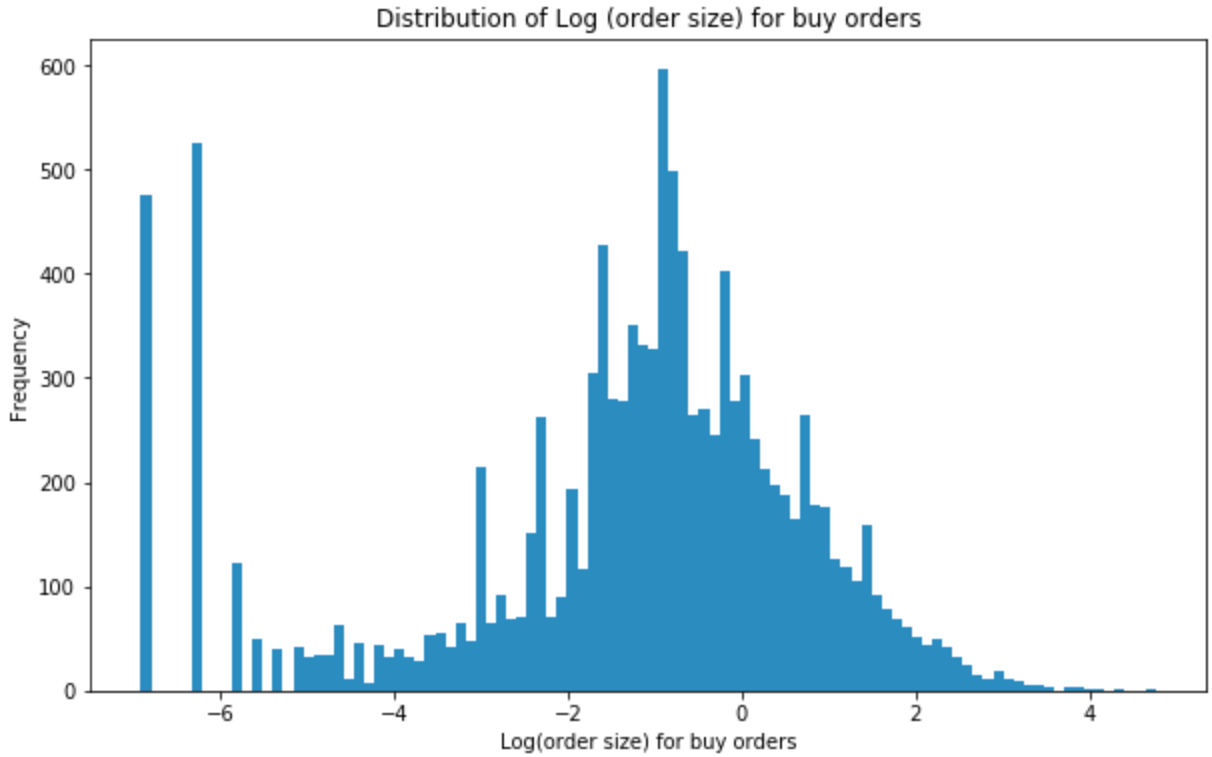
infinite sum of expressions identical to the ones above (let k represent the number of orders required to take out the bid and p the number of orders required to take out the ask):

$$p = \sum_{k=1}^{\infty} \sum_{p=1}^{\infty} p(k \mid B)\, p(p \mid A)\, (1-\gamma)^k \sum_{i=0}^{p-1} \frac{(k-1+i)!}{(k-1)!} \frac{\gamma^i}{i!}$$

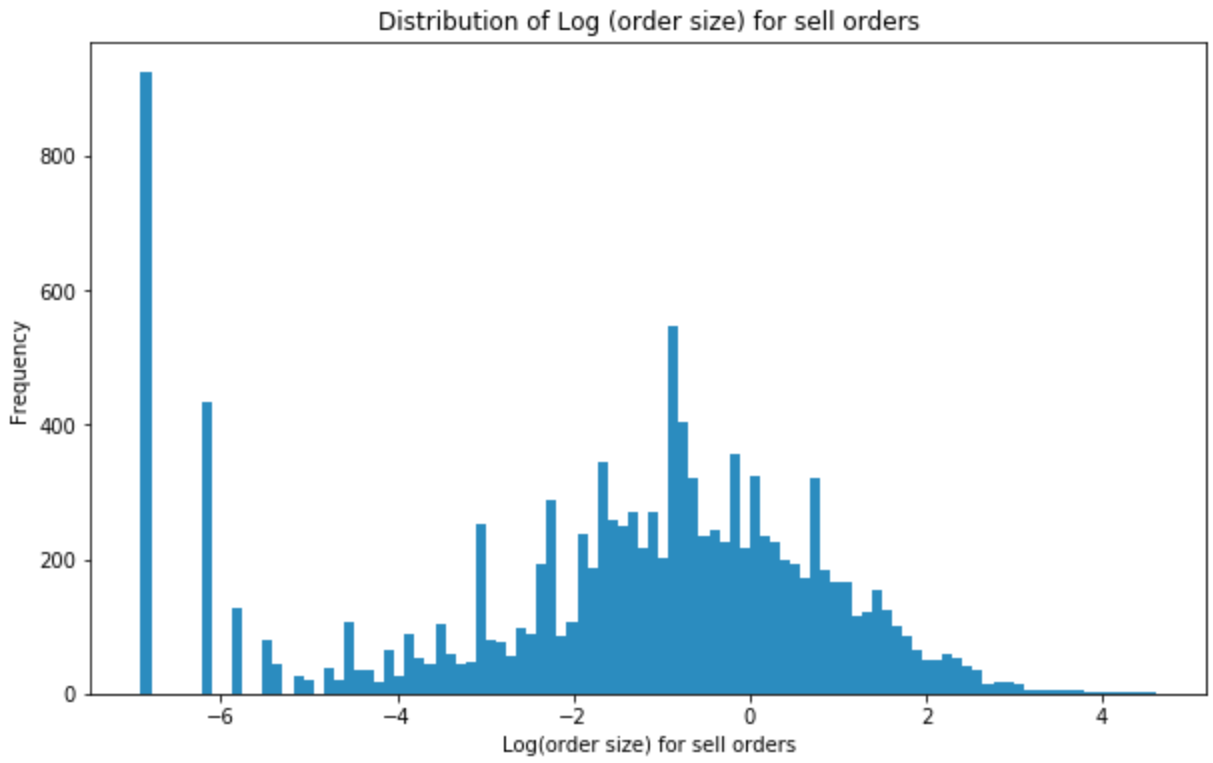
Some of the heavy lifting has thus already been done in our analysis above. We now move on to model the distribution of the number of orders required to take out either the bid or the ask.

First we must determine how the individual order sizes are distributed. One would guess that they may be log-normal since a lot of “size” data is distributed in this way. Taking a look at the distribution seems to confirm that this is a reasonable model:

Buy Orders:



Sell Orders:



Of course, they are not perfectly normally distributed, but it does appear to be a reasonable model that captures a lot of the true underlying distribution.

Assuming then that each order follows a log-normal distribution with some mean and variance (which generally are different for the buy and sell orders), we must now answer the question of how many such orders are necessary to take out a level. The total incoming order volume after k orders will follow the distribution of the sum of k iid log-normal random variables. This distribution is not simple, but fortunately is reasonably well approximated by another log-normal distribution. Using the Fenton-Wilkinson approximation (see this paper) we can then approximate:

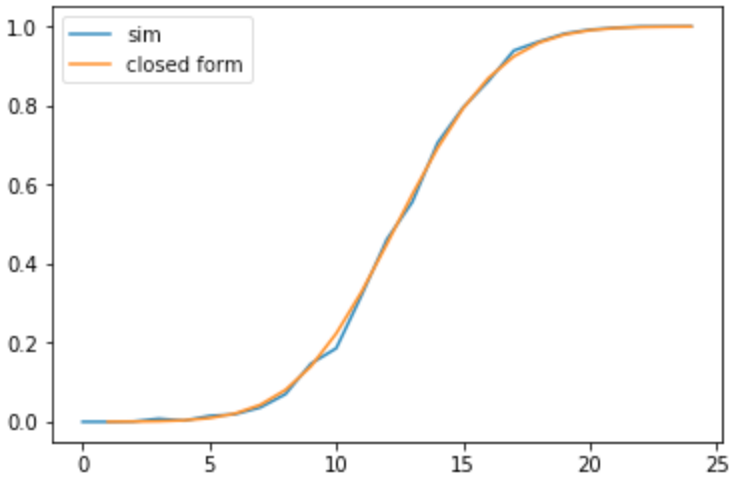
$$V^{(k)} = \sum_{i=1}^{k} V_i \sim LN(\tilde{\mu}_k, \tilde{\sigma}_k), \quad \text{with} \quad \tilde{\sigma}_k^{\,2} = \log\left[(e^{\sigma^2} - 1)/k + 1\right] \quad \text{and} \quad \tilde{\mu}_k = \mu + \log(k) + \tfrac{1}{2}|\tilde{\sigma}_k^{\,2} - \sigma^2|$$

Here μ, σ are the mean and variance of the normal distribution that underlies the log-normal distribution from which the order sizes are generated. Note that the mean of the total volume traded increases with the number of orders considered (k). This, of course, makes sense.

So, given the total volume necessary to be taken out (let's call this Λ) as well as a fixed number of orders, k, we can compute how likely it is that the level was taken out:

$$p(\Lambda \mid k) \;=\; \int_{\Lambda}^{\infty} LN(v\,;\,\tilde{\mu}_k,\,\tilde{\sigma}_k)\,dv$$

I performed a simple Monte Carlo simulation to compare the results against the theoretical distribution (basically 1-CDF for the cumulative log-normal distribution) and found very strong agreement. For example, with a volume of 10 units, and $\mu = -0.5,\ \sigma = 0.8$, the probability of the level being taken out as a function of the number of orders are almost identical:



What we're really interested in, however, is the reverse probability, $p(k \mid \Lambda)$. We need to be careful though since the above probabilities, $p(\Lambda \mid k)$, are not independent. In reality we only proceed to the next trial (i.e. look at the next order) if the prior trial fails. As such, the probability of succeeding on the k^{th} trial is equal to the probability of failing on trial $k-1$ followed by a success on the k^{th} trial:

$$p(k \mid \Lambda) \;=\; p(\Lambda \mid k) \cdot (1 - p(\Lambda \mid k-1))$$

We thus have:

$$p = \sum_{k=1}^{\infty} \sum_{p=1}^{\infty} \{\, p_S(B \mid k) \cdot (1 - p_S(B \mid k-1)) \,\} \;\cdot\; \{\, p_B(A \mid k) \cdot (1 - p_B(A \mid k-1)) \,\}\, (1-\gamma)^k \sum_{n=0}^{p-1} \frac{(k-1+n)!}{(k-1)!} \frac{\gamma^n}{n!}$$

Writing the individual probabilities $p(\Lambda \mid k)$ in terms of error functions (which have fast implementations in numpy), we have

$$p = \sum_{k=1}^{\infty} \sum_{p=1}^{\infty} \{(\tfrac{1}{2} - \tfrac{1}{2} erf(\tfrac{\log(B)-\tilde{\mu}_k}{\sqrt{2}\tilde{\sigma}_k})) \cdot (\tfrac{1}{2} + \tfrac{1}{2} erf(\tfrac{\log(B)-\tilde{\mu}_{k-1}}{\sqrt{2}\tilde{\sigma}_{k-1}}) \,\} \;\cdot\; \{(\tfrac{1}{2} - \tfrac{1}{2} erf(\tfrac{\log(A)-\tilde{\mu}_p}{\sqrt{2}\tilde{\sigma}_p})) \cdot (\tfrac{1}{2} + \tfrac{1}{2} erf(\tfrac{\log(A)-\tilde{\mu}_{p-1}}{\sqrt{2}\tilde{\sigma}_{p-1}}) \,\}\, (1-\gamma)^k \sum_{n=0}^{p-1} \frac{(k-1+n)!}{(k-1)!} \frac{\gamma^n}{n!}$$

This is a rather tricky expression and I suspect that there is not a particularly useful alternate form for it (I could be wrong!). However, in practice, we will not need the full expression but rather only an approximation to it. As such, I suggest that we truncate the infinite sums

whenever the probabilities $p(k \mid \Lambda)$ drop sufficiently low (or better yet when the cumulative probability up to k exceeds some large threshold like 0.99).

Here is some working python code. Generally it appears to take ~0.01 seconds to perform one call to prob(...).

```
import numpy as np
import math

def p_on_or_before_trial(Lambda, k, mu, sigma):
    # Probability that the level is taken out by trial k (could
    # happen before k too).
    sigma_sum = np.log((np.exp(sigma**2) - 1)/k + 1)**0.5
    mu_sum = mu + np.log(k) + 0.5 * abs(sigma_sum**2 - sigma**2)
    return (1 - (0.5 + 0.5 * math.erf((np.log(Lambda) - mu_sum) /
            (np.sqrt(2)*sigma_sum))))

def p_on_trial(Lambda, k, mu, sigma):
    # Probability that the level is taken out on trial k
    # (neither before nor after).
    prob = (p_on_or_before_trial(Lambda, k, mu, sigma) *
            (1-p_on_or_before_trial(Lambda, k-1, mu, sigma)))
    return prob

def p_bid_first(k_bid, k_ask, rate_buy_orders, rate_sell_orders):
    # Probability that the bid is taken out before the ask given
    # the number of orders necessary to take out the levels.
    gamma = rate_buy_orders / (rate_buy_orders + rate_sell_orders)
    p = (1-gamma)**k_bid * sum(
        [math.factorial(k_bid + i - 1) / math.factorial(k_bid - 1) *
         gamma**i / math.factorial(i) for i in range(k_ask)])
    return p

def prob(vol_bid, vol_ask, mu_buy, sigma_buy, mu_sell,
         sigma_sell, rate_buy, rate_sell):
    # Puts it all together and computes the probability that
    # the bid is taken out first.
    center_k = int(vol_bid / np.exp(mu_sell + sigma_sell**2/2))
    center_p = int(vol_ask / np.exp(mu_buy + sigma_buy**2/2))
    low_k, high_k = max(center_k - 5, 1), center_k + 5
    low_p, high_p = max(center_p - 5, 1), center_p + 5
    prob_bid_first = 0
```

```
    for k in range(low_k, high_k):
        for p in range(low_p, high_p):
            prob_bid_first += (
                p_on_trial(vol_bid, k, mu_sell, sigma_sell) *
                p_on_trial(vol_ask, p, mu_buy, sigma_buy) *
                p_bid_first(k, p, rate_buy, rate_sell)
            )
    return prob_bid_first
```

Evaluation

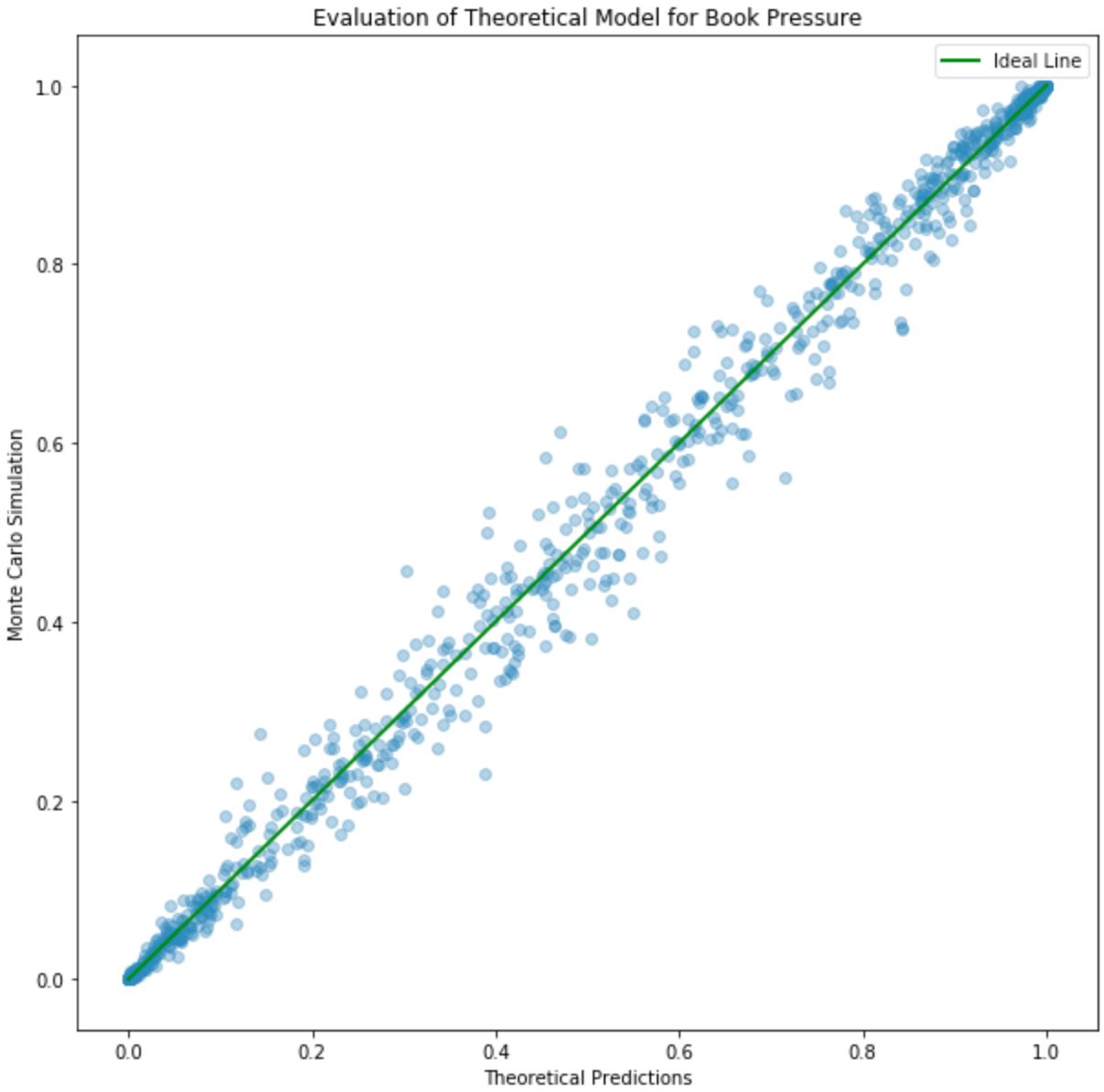
I performed a Monte Carlo simulation to see how well this approximation works. The simulated values were generated using the following logic:

```
def time_to_take_out(target_vol, order_rate, mu, sigma):
    t = 0
    vol = 0
    while vol < target_vol:
       # print(t)
        t += np.random.exponential(1.0/order_rate)
        vol += np.random.lognormal(mu, sigma)
    return t

def bid_first(bid_size, ask_size, buy_mu, buy_sigma,
              sell_mu, sell_sigma, buy_rate, sell_rate):
    times_bid_taken_out = []
    times_ask_taken_out = []
    for i in range(1000):
        times_bid_taken_out.append(
            time_to_take_out(bid_size, sell_rate, sell_mu, sell_sigma))
        times_ask_taken_out.append(
            time_to_take_out(ask_size, buy_rate, buy_mu, buy_sigma))

    bid_first = np.mean(
        [b < a for b, a in zip(times_bid_taken_out, times_ask_taken_out)]
    )
    return bid_first
```

The results are very encouraging! In fact, the correlation looks almost perfect (there's noise because of the simulation):

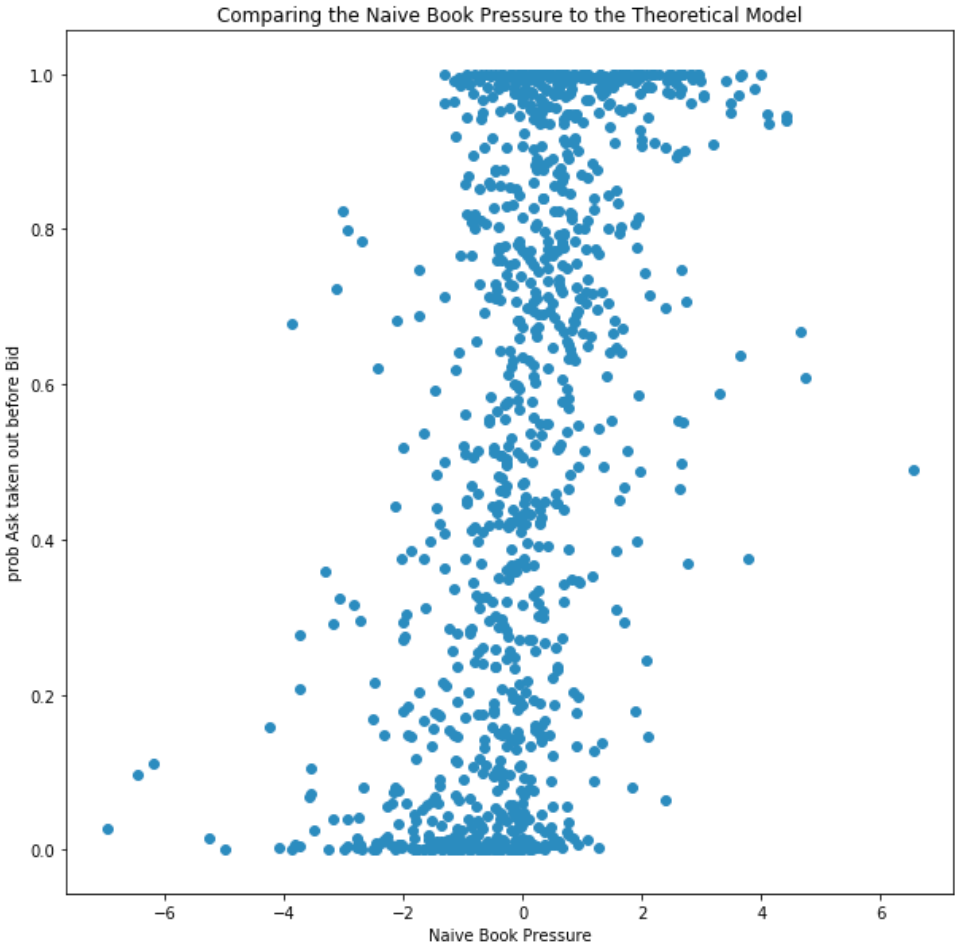


This makes me confident that the expression that we derived above (and also the following approximation in code) are representative of the underlying distribution that governs book pressure.

Application

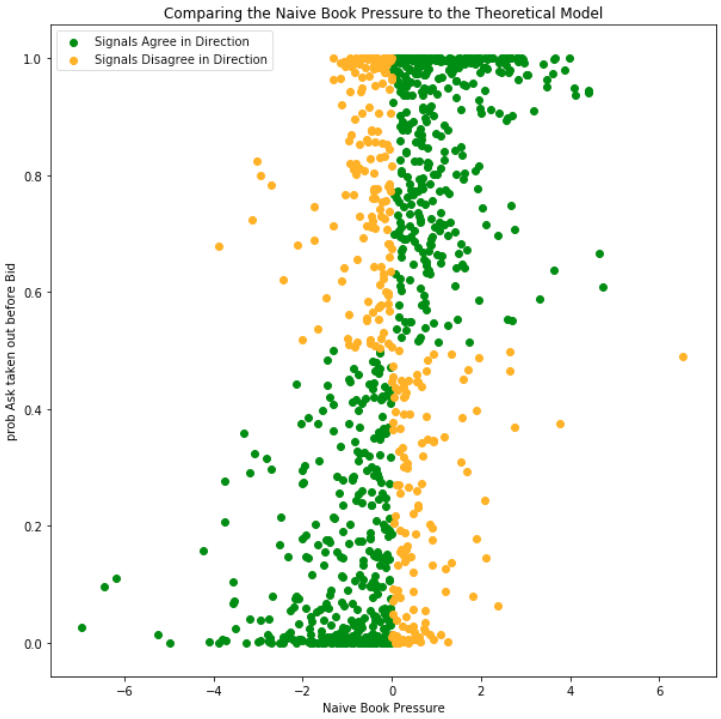
One nice way to apply this would be to compare this expression against the “naive” book pressure formula of log(bid/ask). It’s possible that for various order arrival rates and sizes, this more complete book pressure formula could point in the opposite direction from the naive one. This would allow us to bet against the market but in the correct direction.

I performed such a comparison and found the following very weak correlation:



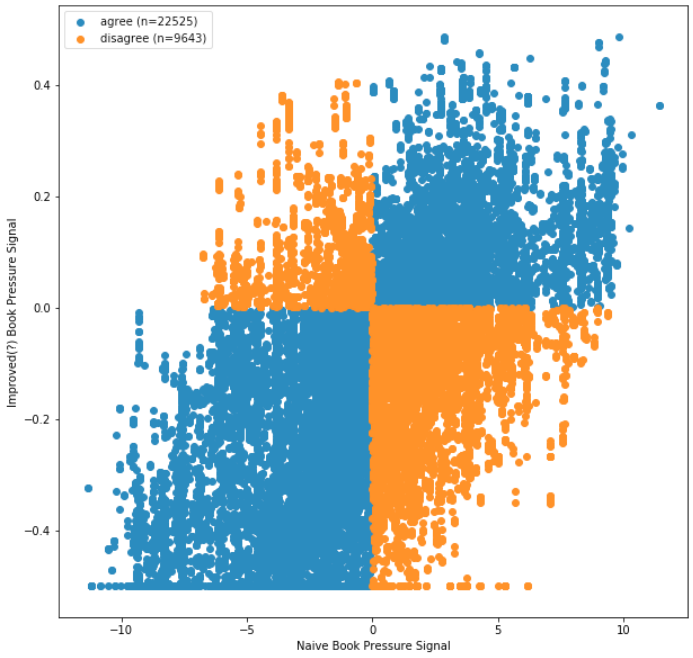
Note that here we look at the probability that the ask is taken out before the bid (opposite of the computation above) which is really the thing we ought have looked at all along since this would generally indicate a positive price movement.

Note also that the two variables are somewhat correlated: when the naive book pressure increases, the probability of the ask being taken out before the bid also generally increases. However, this relationship is very weak and there are many instances where the two point in opposite directions:



The actual applicability of this signal will depend on what the actual order arrival rates etc are.

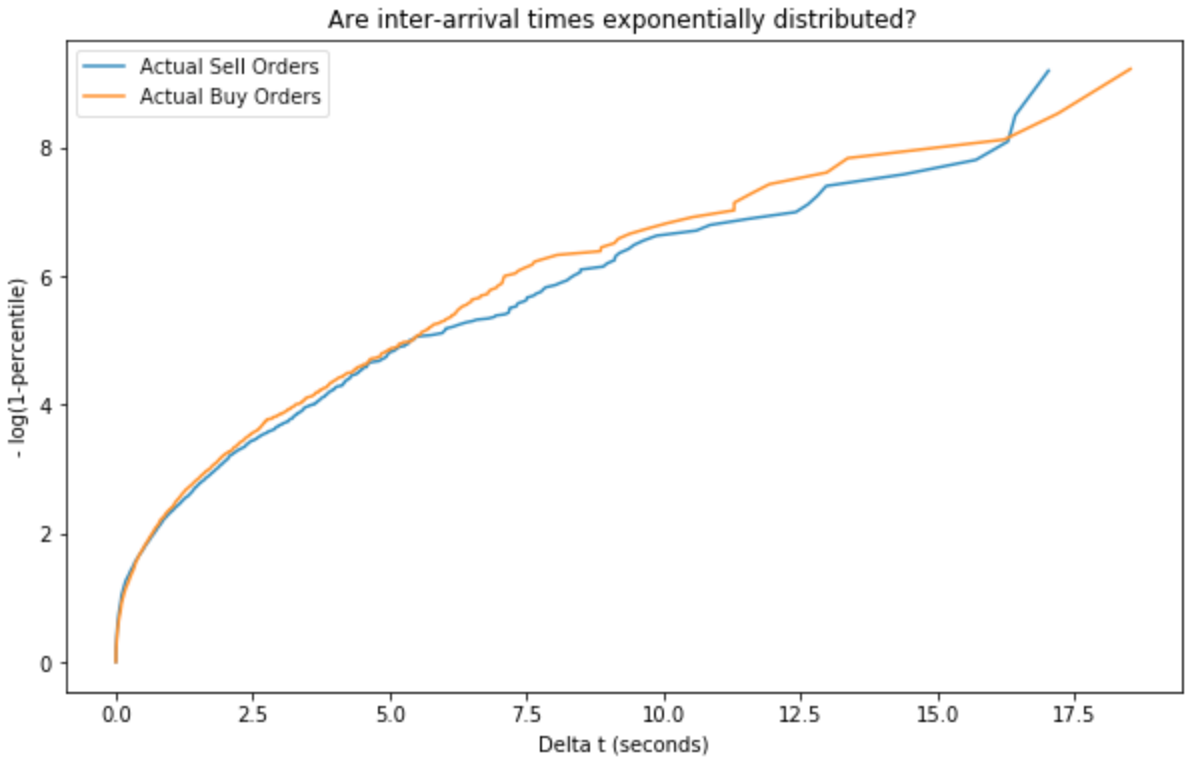
I generated this same plot using real order data and found that there are quite a bit of situations where the two signals disagree, suggesting that we may be able to act on low-competition opportunities:



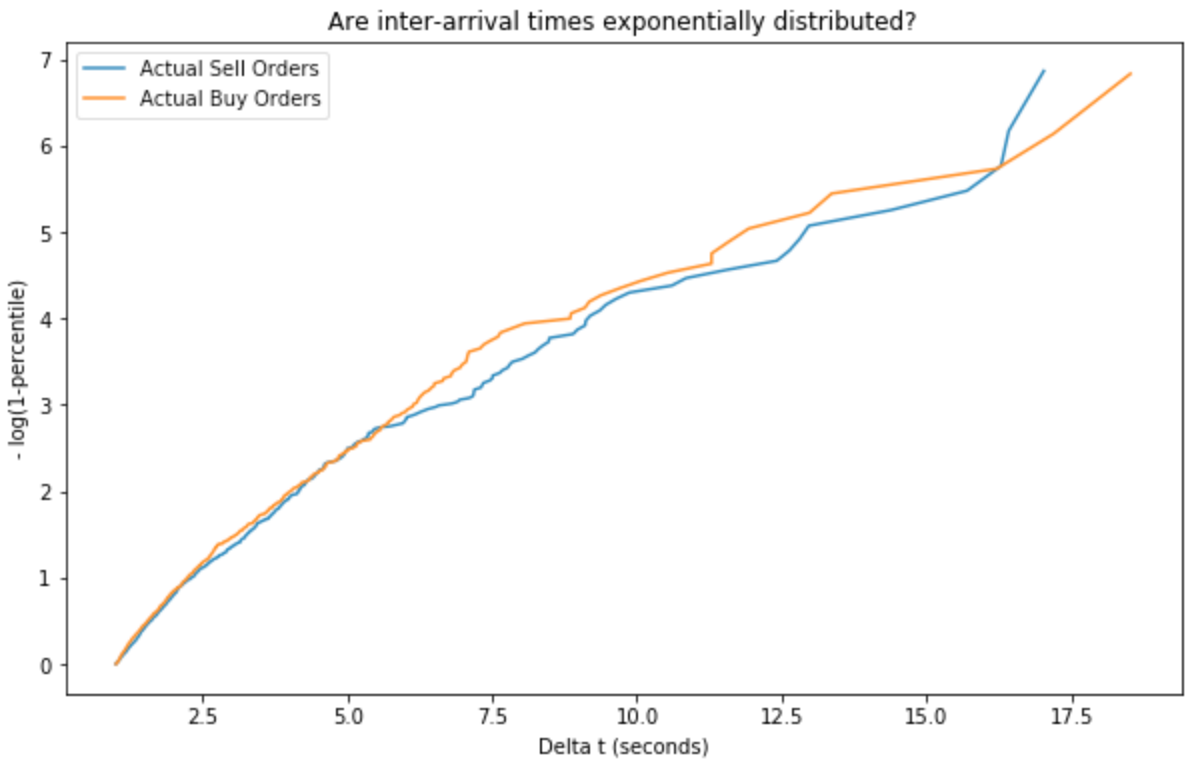
Inter-Arrival Times

Are the inter arrival times actually exponentially distributed? It turns out that they're not truly exponentially distributed, but almost. In particular, "slow" trades are exponentially distributed, but "fast" trades are not. Or, more precisely, they at least follow two distinct types of distributions.

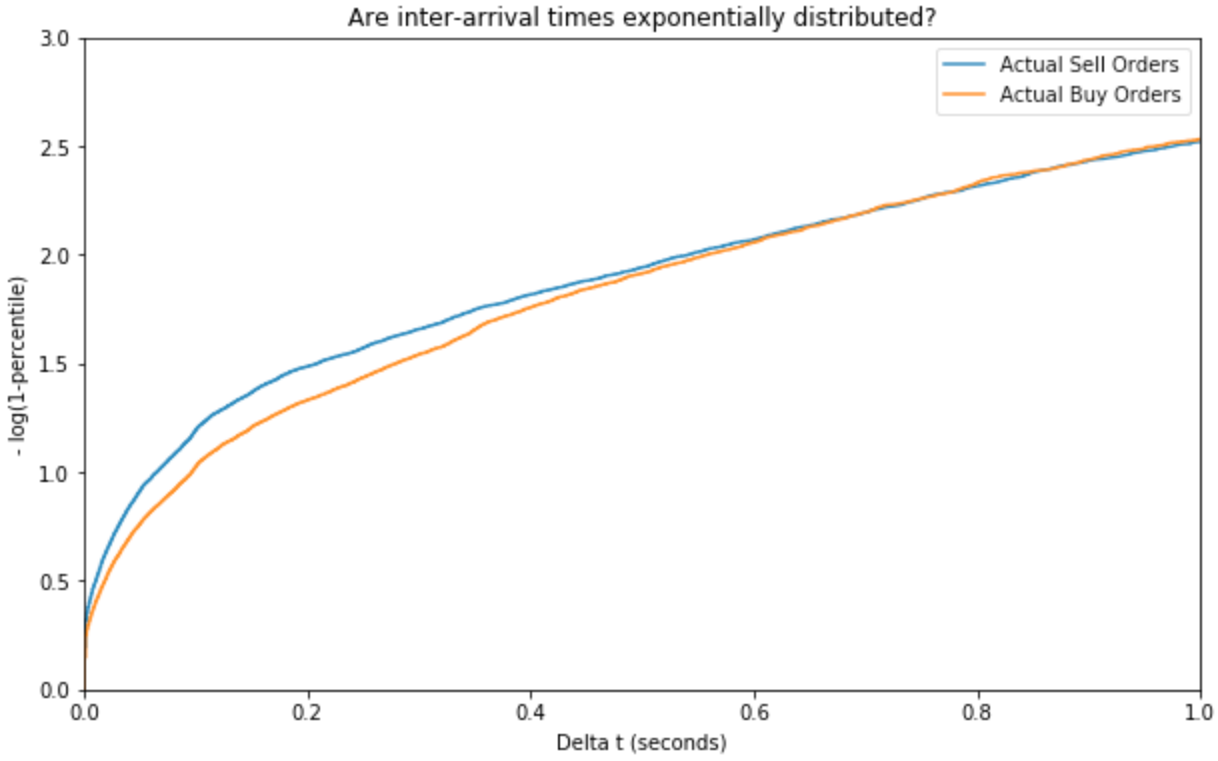
To check this, one can look at the cdf. However, since the CDF for hte exponential distribution is also exponential, it's tough to inspect this visually. Instead we plot $-\log(1-cdf)$ and expect to see a straight line with slope λ. In reality we see something that approaches a line for larger times, but has a distinct curve near the bottom:



If we exclude these "fast" trades (let's say sub-second), then the plot looks very linear:



Similarly, if we drop the points after one second, the plot is also quite linear:



Point being that I think the exponential distribution reasonably well models the inter-arrival times (although certainly not perfectly).