Technische Universiteit Delft
Faculteit Elektrotechniek, Wiskunde en Informatica
Delft Institute of Applied Mathematics

Building a model for the Limit Order Book

Verslag ten behoeve van het
Delft Institute of Applied Mathematics
als onderdeel ter verkrijging

van de graad van

BACHELOR OF SCIENCE
in
TECHNISCHE WISKUNDE

door

Sebas Hendriks

Delft, Nederland
Juli 2014

BSc verslag TECHNISCHE WISKUNDE

"Building a model for the Limit Order Book"

Sebas Hendriks

Technische Universiteit Delft

Begeleider

Dr.ir. J.H.M. Anderluh

Overige commissieleden

Dr. J.L.A. Dubbeldam

Dr. B. van den Dries

Juli, 2014 Delft

Contents

Abstract

The Limit Order Book is a widely used tool of exchanges to allow traders to buy or sell stock easily. Seeing that in the financial sector a general rule of thumb is that the guy with the most knowledge also will have the best strategy and thereby make the most money, it is critical to have a very good understanding of the workings of the Limit Order Book. We will give some deeper insights of Limit Order Books and exchanges in general. We will use an established stochastic model for the dynamics of the Limit Order Book to simulate the price path of a trading day. We also alter this model to simulate the pre-opening phase of the trading day. We also compare data generated by the model for just the regular trading session and the opening auction to real data for Vodafone stocks and evaluate the performance of our models.

Chapter 1

Introduction

1.1 What is a Limit Order Book?

Limit Order Books (LOB) are the dominant financial tool used by exchanges where all participants are able to trade assets through the use of buy and sell orders. Also, they all have access to the top few buy and bottom few sell orders from the other traders on which they can base their trading strategy. These orders can take the following two forms:

- *Market order*: an order to buy/sell a certain amount of stock for the best available price, resulting in an immediate matching,
- *Limit order*: an order to buy/sell a certain amount of stock for a specified price and will remain active until it is matched to a new market order, or is cancelled by the issuer.

It is clear that when we place a buy limit order for a higher price than an unmatched sell order, we can classify this buy order as a market order, as this will result in an immediate transaction. The same goes for sell limit orders for a price below a buy order. All these limit orders that remain unmatched form the Limit Order Book (or LOB) at that point in time.

Many exchanges also alllow traders to submit so-called *hidden orders*, where some part of their orders can not be seen by other traders. An example of this are *iceberg orders*, which are limit orders where not only a total size needs to be specified, but also a visible size, which will be visible to other traders. These types of orders give traders the advantage that they know something about the order book that others don't. It is partially for this reason that there has been an increase in popularity of so-called *dark pools*, which are LOBs where all active orders have hidden sizes. Although these dark pools can be advantageous to for instance large investments banks, as they can make massive transaction without influencing the price too much, this can also be very risky, as it becomes harder and harder to regulate the market.

The LOB forms a very interesting subject to study as it has multiple sides which can be researched. There is therefore a large contrast between some of the manner of studying LOBS. Some prefer to look more at the behaviour of the traders that influence the LOB, whereas others take a more analytical look at the dynamics. Some examples, as proposed by [6]:

- Fitting the parameters of laws of order sizes and arrival rates [1, 2, 4]
- Strategies and behaviour of high frequency traders [3]
- Finding relevant parameters for high frequency traders
- Direction of price moves under different states of the first limits of the LOB

1.2 Goal of this paper

Seeing that a lot of money can be won or lost depending on how well one understands the complex workings of an LOB, it would be a start to be able to simulate an LOB to be as similar as possible to real data in order to use this to predict an asset's price path. We start by using the stochastic model proposed in [4] where arrival rates are assumed to form an independent Poisson process, which we use to create a model for regular trading. Next, we will try to change the proposed model so that it can be used as a model to simulate the opening auction, which then can be used as a starting point in the simulation of regular trading. Lastly, we will simulate a data set for both the close price at the end of the trading day and the opening price at the end of the opening auction for a given closing price. We will see if these data sets are similar to real data by comparing their distributions.

Chapter 2

Market dynamics

When studying the field of finance, the dynamics of stock exchanges form one of the most interesting topics. Traders willing to take risks and with the right amount of financial knowledge can make a lot of money, which is why a lot of research is centered around this topic for both academic and practical insights.

In the past, most exchanges would take the form of a quote driven market. Here, a small number of market makers determine the stock price through their bid and ask quotations. As a trader, only these quotations would be visible and would be the prices you could trade for. This type of market has the advantage that the market makers have to trade at their quoted prices, however, there is no transparency in the market as only a small number of order prices are shown.

Nowadays, stock exchanges worldwide are all becoming electronic, order-driven markets. Examples of these exchanges are Euronext, NYSE, the London Stock Exchange and the Tokyo Stock Exchange, which all have been around for quite some time, unlike exchanges like BATS, which are fully electronic and therefore fairly new. Every time a event takes place, so an order is placed or cancelled, a snapshot of the order book is made, which can be accessed by the traders. On an average day for an average stock, over a hundred thousand snapshots are made. Most likely, traders will have access to what is known as the *Level 2 market data*, which consist of the snapshots with the top few price levels of the order book being visible. An example can be found in Figure 2.1.

Top of Book

Bid		Ask	
Price	Size	Price	Size
14.31	4,000	14.33	5,000
14.30	7,400	14.34	12,300
14.29	11,000	14.35	5,000
14.28	4,000	14.36	5,000
14.27	5,000	14.37	5,000

Figure 2.1: Example of a snapshot of an order book

In this example, we only have the top 5 price levels that are visible. We can clearly see what the best buy and sell prices are, as they are on the top of their respective columns, with that row forming the so-called *best bid-offer line* (BBO). The amount of other rows below this BBO is what is known as the *depth of the book*. This number varies between different assets, depending on, amongst others, how volatile the price of the asset is. Another thing to note is that the order sizes in the table are combined totals for all orders at that price level, so buying the 5000 shares at price 14.33 could mean buying 3000 from trader A and 2000 from trader B.

Different exchanges also work with different sorts of priority. As described in the example above, several active orders can have the same price at a given time. In this case, as priority principle needs to be set to dictate which orders will be executed first. Most exchanges use a priority based on time. This means that orders that were placed first, will get executed before other orders at the same price. This system encourages traders to essentially "show their hand" by placing limit orders earlier rather than later. Another system is by giving priority to orders with the largest size. This way, it rewards traders who place orders with large sizes and thereby providing the market with greater liquidity.

A difficulty presented by the most widely used time-priority is that traders with the fastest connection to the exchange will have a advantage to the rest. It is for this reason that many traders compete by buying increasingly faster internet cables to the exchange in an attempt to beat the rest. At the moment, this advantage can be a mere couple of milliseconds, however this can be worth millions to the traders. The book *Flash Boys* by Michael Lewis describes this competition between these high frequency traders and the so-called flash crashes that have happened partially due to them where the price of a stock would fall dramatically but then return nearly back to normal in a matter of minutes.

Another aspect of exchanges is the way a trading day is built up. Whereas the placing of orders is possible during the entire trading day, the execution of orders only takes place during the regular trading hours of the exchange. A trading day will consist of three phases with the following characteristics:

- *Pre-opening phase:* An opening auction held at the beginning of the day (from 7:15 to 9:00 for the Euronext), where orders are placed in the order book, but no transactions take place. At the end of this phase, all orders that can possibly be executed against an opposing order (how exactly this is done is explained later), and the resulting order book determines the opening price.

- *Main trading session:* Continuous trading takes place where all incoming orders are checked to be matched with standing limit orders.

- *Closing phase:* Similar to the pre-opening phase, at the end of the day (from 17:30 to 17:35 for the Euronext) as orders can still be submitted but not matched. After this a closing price is determined which will serve as a guideline for orders on the pre-opening phase of the next trading day.

With the Euronext, besides stock being able to be traded using though continuous matching of orders, it is also possible for stock to be traded using call auctions procedures, where orders are accumulated without execution. These auctions also consist of three phases, namely:

- *Call phase:* Here, orders are automatically recorded without giving rise to Transactions. Traders can enter new orders as well as modify or cancel existing orders. All traders do have access to the indicative price, which is the price for which the maximum number of orders can be executed, which is updated continuously.

- *Price determination phase:* After the call phase, a price is determined which is the indicative price at that time. During the price determination phase, no new orders can be entered and existing orders may not be modified or cancelled.

- *Trading-at-last:* For a brief period after an auction price has been determined, orders can be placed for execution at this auction price.

Here, we will just be looking at trading through continuous order matching for our models.

Chapter 3

Basic model for an LOB

3.1 Dynamics of an LOB

We start by looking how normal trading can be simulated with the use of Matlab for instance. First is to look at the limit and market orders. Similarly to the model in [4], we can put the limit orders on a price grid $\{1, \ldots, n\}$, where n is a price chosen high enough to ensure the implausibility of receiving a limit order for a higher price. This is possible as we will chose a rather small time frame. The grid nodes are evenly spaced out with distance π, which is the given *tick size* of the asset. This means that if, for example $\pi = \$0.01$, the smallest permissible price of an order that is larger than \$1.00 is \$1.01.

Next, we define $X(t) = (X_1(t), \ldots, X_n(t))_{t\geq 0}$ as the number of outstanding limit orders at time t at the various price levels. If $X_p(t) < 0$, there are $-X_p(t)$ bid orders at price p. Similarly, if $X_p(t) > 0$, there are $X_p(t)$ ask orders at price p. Now we can divide this matrix $X(t)$ in bid and ask orders. To do this, we define the bid and ask price as the price for which all the bid/ask orders fall below/above in our price grid, so:

$$p_A(t) \equiv \inf\{p = 1, \ldots, n, X_p(t) > 0\} \wedge (n+1).$$

as the ask price and for the bid price:

$$p_B(t) \equiv \sup\{p = 1, \ldots, n, X_p(t) < 0\} \vee 0$$

Now, using these two values, we can define the *mid price* $p_M(t)$ and the *bid-ask spread* $p_S(t)$ as:

$$p_M(t) \equiv \frac{p_B(t) + p_A(t)}{2} \text{ and } p_S(t) = p_A(t) - p_B(t)$$

Because the main idea behind the limit order book and therefore the model that we are building is the processing of incoming orders, we will now look at how these new orders are added to the order book. As mentioned earlier, exchanges make use of a base unit, the tick size, where all permissible prices are a multiple of this value. Here we will use a unit size of 1. This means that for an incoming order, for a given state $x \in \mathbb{Z}^n$ and a price p in our price grid, then:

$$x^p \pm 1 \equiv x \pm (0, \ldots, 1, \ldots, 0),$$

where x has the value 1 added or subtracted to it's p'th component. This causes the following:

- A limit buy order at price $p < p_A(t)$ decreases the quantity at price p : $x \to x^p - 1$
- A limit sell order at price $p > p_B(t)$ increases the quantity at price p : $x \to x^p + 1$
- A market buy order decreases the quantity at the ask price: $x \to x^{p_A(t)} - 1$
- A market sell order increases the quantity at the ask price: $x \to x^{p_B(t)} + 1$
- A cancellation of an outstanding limit buy order at price level $p < p_A(t)$ increases the quantity at price $p : x \to x^p + 1$
- A cancellation of an outstanding limit sell order at price level $p > p_B(t)$ decreases the quantity at price $p : x \to x^p - 1$

Our model for the order book will be built up from incoming limit orders, market orders and cancellations at the various price levels. To simulate this, we will be looking at the frequency with which these different orders arrive at each price level. For example, from [2], we learn that for limit orders closer to the opposing bid or ask price, this frequency is higher than for limit orders with a price further from that bid/ask price and therefore that their arrival rate is also higher.

To realize this feature in our model, we will be generating arrival times for limit orders, market orders and cancellations using the independent Poisson processes proposed in [4]. This means that for $i \geq 1$:

- Limit orders arrive at a distance of i ticks from the opposing bid/ask price at independent, exponential times with rate $\lambda(i)$,
- Market orders arrive at independent, exponential times with rate μ,
- Cancellations arrive at a distance of i ticks from the opposing bid/ask price at independent, exponential times with a rate dependent on the amount of outstanding orders at that price level, so for a price level with x outstanding orders, we have a cancellation rate of $\theta(i)x$.

So for a given arrival rate, we can generate an exponential arrival time in Matlab for all the price levels. For limit orders, [2] suggests a power law, so an arrival rate of:

$$\lambda(i) = \frac{k}{i^\alpha} \tag{3.1}$$

3.2 Matlab implementation

To now implement this in Matlab to generate an LOB, we start by generating a matrix R with arrival times using our vectors λ, θ, and our value for μ. So we create a $4 \times n$ matrix, which we will call R, where we have:

- The first row of R is filled so that $R(1, i)$ is the arrival time for the next limit order at distance i of opposing bid/ask price

- The second row of R consists of all values being infinite, except $R(2,1)$ which is the arrival time for the next market order
- The third and forth rows are filled so that $R(3,i)$ and $R(4,i)$ represent the arrival time of the next cancellation of respectively a sell order and a buy order at distance i of the opposing bid/ask price.

First off, it must be noted that in this construction of R we have only one row for the limit and market orders, whereas we have two for both the cancellations of buy and sell orders. The reason for this is that the arrival rate of limit and market orders work identically whether they are buy or sell orders, as they are only dependent on the parameters λ and μ. This is not the case for cancellations however, as they are also dependent on the amount of orders for the price level of which we are calculating the arrival rate. These most often differ for the amount of buy orders at a given distance i from the ask price, and the amount of sell orders at the same distance from the bid price.

So we know the arrival times are exponentially distributed, so for instance $R(1,1) \sim Exp(\lambda(1))$. We now want to use a uniform random variable $U(0,1)$ to generate the arrival time. We know that for $F(t)$, the distribution function of our exponential distribution, we have:

$$\{U \leq F(t)\} = \{F^{inv}(U) \leq t\} \tag{3.2}$$

Because for $U(0,1)$ we know that $P(U \leq x) = x$ for $x \in [0,1]$, we see that $P(U \leq F(t)) = F(t)$. From (3.2), we see that:

$$P(F^{inv}(U) \leq t) = F(t)$$

which means that the random variable $F^{inv}(U)$ has the same distribution as $F(t)$. So we need to find F^{inv} as follows:

$$F(t) = U \Leftrightarrow t = F^{inv}(U)$$

so for our exponential distribution $Exp(\lambda)$, with $F(t) = 1 - e^{-\lambda t}$, we find:

$$\begin{aligned} F(t) = U \quad &\Leftrightarrow \quad 1 - e^{-\lambda t} = U \\ &\Leftrightarrow \quad e^{-\lambda t} = 1 - U \\ &\Leftrightarrow \quad \lambda t = -\ln(1-U) \\ &\Leftrightarrow \quad t = -\ln(1-U)/\lambda \end{aligned}$$

However, as $1 - U$ also follows the $U(0,1)$-distribution, we can write this as: $-\ln(U)/\lambda$. Now we knew that $R(1,1) \sim Exp(\lambda(1))$, so we can generate the arrival time with $-\ln(U(0,1))/\lambda(1)$. In Matlab, this will mean for the matrix R we get the following values:

$$\begin{array}{lcllcl} R(1,i) & = & -\log(U)/\lambda(i) & R(2,1) & = & -\log(U)/\mu, \\ R(3,i) & = & -\log(U)/(\theta(i)|x_{bp+i}|) & R(4,i) & = & -\log(U)/(\theta(i)|x_{ap-i}|) \end{array}$$

where U is a random number generated from the $U(0,1)$-distribution. Next is to find the minimum of all these values and the row and column of this minimum in R. So we will call

$$m = min(R) = R(x,p)$$

In our program, we can now determine which order needs to be added to the current order book through the value of x. If $x = 1$, we add a limit order at distance p of opposing ask/bid price. If $x = 2$ we add a market order. If $x = 3$ or $x = 4$ we add a cancellation of a limit order at distance p of opposing ask/bid price. For our value m, we can see this as the time that has passed between our previous order and the current one. So if we start our model with $t = 0$, we can set t after this order as $t = t + m$.

Now, because we are assuming these orders arrive following a Poisson process, we know that the arrival times of all orders are independent. Therefore, we can now just subtract m from our matrix R and generate a new arrival time for the next order of the same kind as the last. We can then just repeat this process of finding the minimum of R, adding the corresponding order, increasing our value t, subtracting m from R and generating a new arrival time until our value t reaches a certain end time.

We can now simulate a very simple model for an order book. We start by finding a value for our parameters. As it is to see whether the model works correctly and we are not comparing it to real data yet, we can use the values found in [4]. These are for λ: $k = 1.92$ and $\alpha = 0.52$, so:

$$\lambda(i) = \frac{1.92}{i^{0.52}}$$

We also have:

$$\mu = 0.94 \text{ and } \theta(i) = \begin{bmatrix} 0.71 & 0.81 & 0.68 & 0.56 & 0.47 & 0.47 & \dots \end{bmatrix}$$

Let's say that we start by simulating an hour of normal trading. In this case, we thus choose an end time of 3600, the amount of seconds in an hour, and a price grid of $p \in \{1, 100\}I$. Lastly, because our bid-ask spread will be very large if we start with just an empty order book, we start the hour by adding a buy order at price $p = 44$ and a sell order at price $p = 57$. If we now use the Matlab code in the Appendix, we will find the following bar chart representing our order book at the end of the hour:

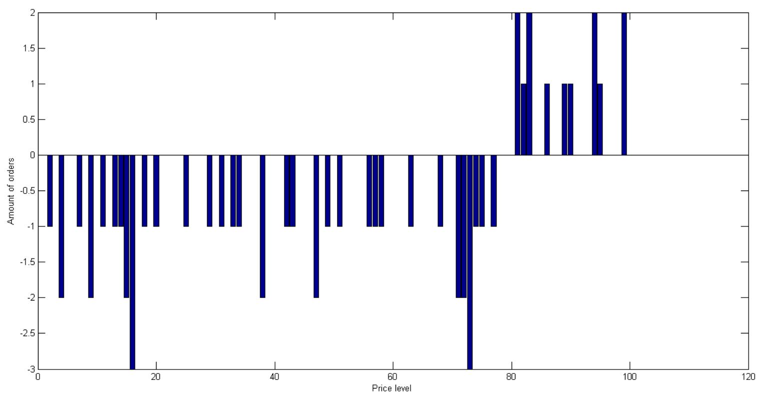

Figure 3.1: Bar chart of the order book at time $t = 3600$

Next, we can also plot the mid price P_{mid} as it has shifted over the hour, this gives us:

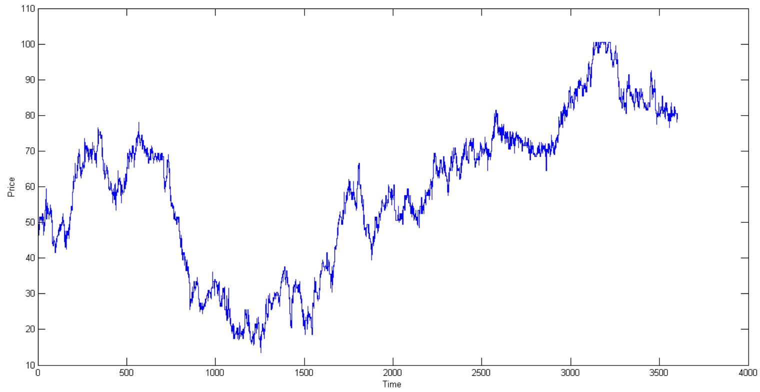

Figure 3.2: Plot of P_{mid} against the time

Even though there is not a lot we can learn from the bar chart, the plot for the mid price does look quite a lot like a real price path. Given the corresponding parameters and then simulating for an entire day, we should be able to get a model with many similarities to the data. However, it might prove even more accurate to improve this model on certain features.

Chapter 4

Adding the pre-opening phase

A problem we end up facing when building a model with the tools we have up to this point is that there is no realistic starting point from which we continue our simulation. To get this starting point, we will add a starting phase resembling the pre-opening phase as described in the introduction, so where orders can be placed but not executed.

So for the model we will start the pre-opening phase with no standing limit orders in the order book and we decide a closing price (CP) for the previous day. Once again we will work with arrival rates for the various types of orders. However, because no transactions take place, we will split our order book up into two parts: the bid orders and the ask orders.

During the pre-opening phase, it is still allowed to place market orders which will, upon the opening of the auction, be processed against the opposing orders with the best price. To do this, we will also have a third part for our order book consisting of the two values for the amount of buy and sell market orders. Cancellation orders can still be placed and executed immediately, as we can just cancel the outstanding bid or ask orders at their respective sides.

Similarly to what we did with our model for the main trading session, we can now just determine an end time for our pre-opening phase and create a similar matrix R with the arrival times of the next order at the various price levels. There is a slight twist in this again for determining the arrival rates for the different types of orders. This time, because we can have buy orders above the ask price and vice versa, we no longer can make use of our formula for the arrival rate λ like we did before.

This time, we are given a closing price around which we can assume most limit orders will be placed. So, here we will assume both sides to have symmetric arrival rates with respect to the closing price. So instead of the formula for λ in (3.1), we will use the following:

$$\lambda(i) = \begin{cases} \frac{k}{-i^{\alpha}} & \text{for } i \in (1, CP-1) \\ \frac{k}{i^{\alpha}} & \text{for } i \in (CP, n) \end{cases} \tag{4.1}$$

where instead of i being the distance to the ask or bid price, it is now simply the price level. This will look as follows:

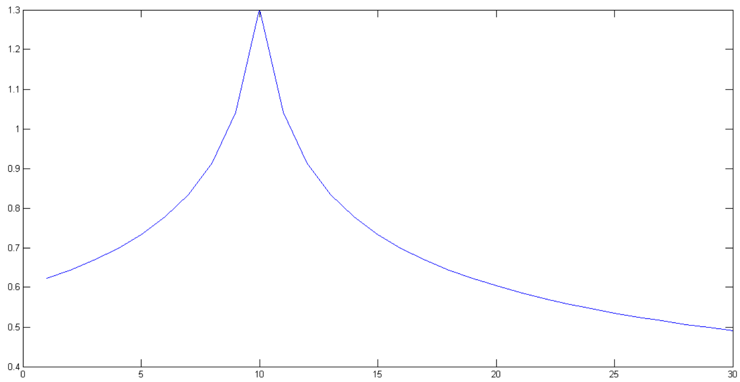

Figure 4.1: Plot of λ with n=30 and CP=10

However, an issue during testing was that if we use this λ for both buy and sell orders, we have a very small variance for the opening price, with the difference between the opening price and CP being no more than 3 price levels, as the amount of limit buy and sell orders will be nearly identical in their symmetry with respect to CP. For this reason, and because we can assume traders will try to save as much money as possible when trading, we will shift this function to be symmetric with respect to $(CP-1)$ and $(CP+1)$ for the buy and sell orders respectively. By doing this, the ask orders have a higher average price than the buy orders. For market orders we can make use of the same arrival rate of μ. For cancellations we can still use $\theta(i)|x|$, but again i is not the distance to the ask or bid price, but the price level.

We can now use these arrival rates to generate a new R matrix which is now a $5 \times n$ matrix, as we now have two rows for arrival times of the buy and sell limit orders. Now, similar to before, we keep looking for the m =minimal value of R, add the corresponding order, subtract R by m, add m to t, our time value, and generate a new arrival time for this element of R. This process is repeated until t reaches a certain value. We now have 2 order books consisting of just sell or buy orders, which we will call X_1 and X_2 respectively, and a third matrix X_3 with the two values of the amount of buy $(X_3(1))$ and sell $(X_3(2))$ market orders.

Next is two merge these two order books into one which will be our starting point we were looking for to start the main trading session with. To do this, we first execute the market orders. To do this, we look at the minimum between the amount of buy market orders and the amount of sell limit orders at the ask price, so $\min(X_3(1), X_1(\text{ap}))$. We subtract this value from both $X_3(1)$ and $X_2(\text{ap})$, to either execute all the buy market orders, or to have the ask price move up one price level as $X_1(\text{ap})$ then becomes zero. In this case we repeat until the amount of market orders is zeros, so:

$$\begin{aligned} \text{while } X_3(1) \neq 0 : \qquad & m = \min(X_3(1), X_1(\text{ap})) \\ & X_3(1) = X_3(1) - m \text{ and } X_1(\text{ap}) = X_1(\text{ap}) - m \\ & \text{ap} = \text{ap} + 1 \end{aligned}$$

We do the same thing for the sell market orders and the buy limit orders at the bid price.

Now we need to combine the buy and sell orders in such a way that we end up with that the bid price is lower than the ask price. We do this, if it is not the case that $BP < AP$, by looking at the minimum between the amount of buy orders at the bid price and the amount of sell orders at the ask price. We subtract both values by this minimum and move the ask price up by a price level if this amount was the minimum, and the bid price down if that amount was the minimum. We can then combine the orders to find our new order book and starting point for our main trading session.

We can also give a more formulaic description of the opening price after the pre-opening phase. As described by [5] We can find a value C_p for each price p which would equal the total volume of trades possible by matching buy orders with a price greater or equal to p to sell orders with a price less or equal to p. We can then find the *uncrossing price* as:

$$\hat{p} = \arg\max_p C_p$$

which would be the value of our opening price. To be more in line with our model, if again we have X_1 as the array for the amount of sell orders at the various price levels after the pre-opening phase, X_2 the array of the amount of buy orders and $X_3(1)$ and $X_3(2)$ the amount of market sell and buy orders, we can write $\hat{p}$ as:

$$\hat{p} = \max\{p \in (1,n) : \sum_{i=p}^{n} X_2(i) + X_3(2) \geq \sum_{j=1}^{p} X_1(j) + X_3(1)\} \tag{4.2}$$

which is essentially the same as the definition given above, as we find the value for which all sell orders for price smaller or equal to p can be matched to a buy order of price strictly greater than p.

4.1 Example in Matlab

Just like we did for regular trading, we can now simulate a pre-opening session with the resulting order book which we could use to start the main trading session with. Once again, we will use similar parameters as we did for the previous simulation, so now for our new formula for λ from (4.1), we use $k = 1.92$ and $\alpha = 0.52$. For our μ and θ we need to make some modifications however. When testing in Matlab, it became apparent that the values we used earlier caused some problems in the simulation. Initially, we had as before:

$$\begin{aligned} \mu &= 0.94 \\ \theta(i) &= [\ \ldots\ 0.47\ \ 0.56\ \ 0.68\ \ 0.81\ \ 0.71\ \ 0.71\ \ 0.81\ \ 0.68\ \ 0.56\ \ 0.47\ \ \ldots\ ] \end{aligned}$$

with θ being symmetric with respect to $n/2$. For it to be visually clearer, we will want to use fewer price levels, so we chose $n = 24$, set our closing price in the middle of our price range, so $CP = 12$, and we simulate a pre-opening phase of just 10 minutes or 600 seconds. However, with these parameters, we found that we ended up with too many cancellations and market orders, as we would end up with a nearly empty order book. For this reason we choose slightly lower values for these parameters μ and θ. We now set $\mu = 0.1$ and we just divide our previous θ by 2 to get our new θ.

Using the method explained earlier in Matlab, we would generate our sell and buy order books X_1 and X_2 and our market orders X_3. Adding the difference between the buy and sell market orders to the amount of buy orders on the highest price level, we get the following bar graph:

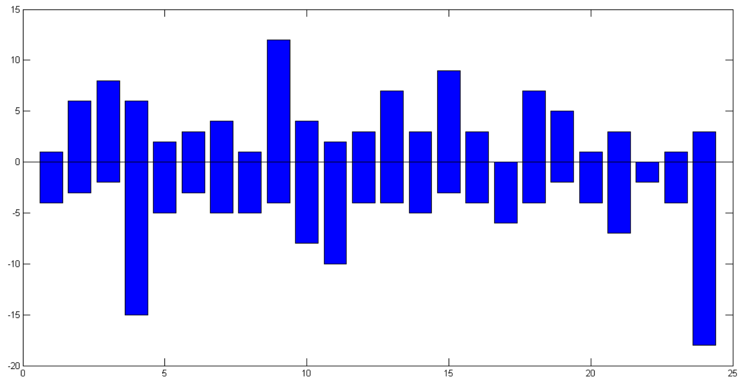

Figure 4.2: Bar graph of the order book before execution of orders

We then use equation (4.2) to find our value $\hat{p}$ to determine which orders get executed. In this case, we find that $\hat{p} = 13$. We now get the same graph as before, except now all executable orders, which are the buy orders above and the sell orders below $p = 13$ that are coloured red:

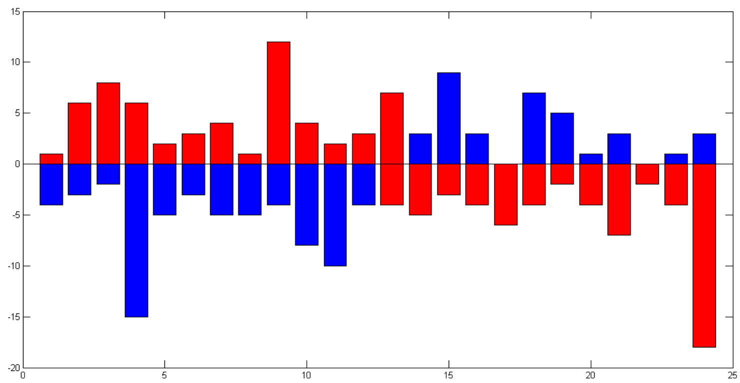

Figure 4.3: Bar graph of the order book before execution of orders

When these orders are then executed upon the beginning of the main trading session, we are left with our new order book generated by the pre-opening phase:

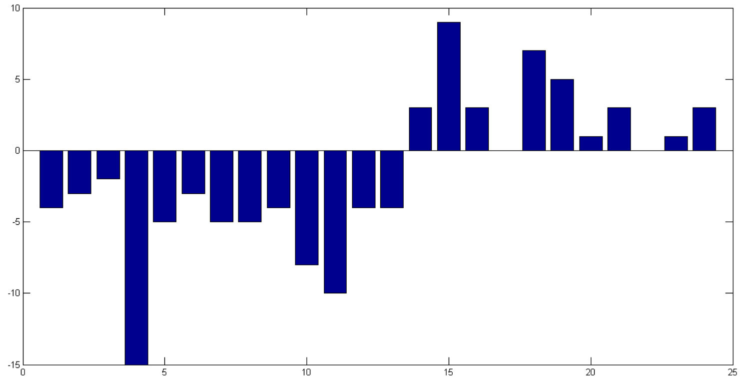

Figure 4.4: Bar graph of the order book after pre-opening phase

We can see that we have generated a valid order book as all the bid prices have a lower price than all the sell orders. We can also see what our opening price, or the mid price of our new order book, is so in this case: $P_{mid} = 13.5$. So now we can see that, as our closing price was $CP = 12$, our price has gone up by 1.5 price units. To see if our model for the pre-opening phase is accurate, we will look at the difference between the distribution of the difference between closing and opening price for real data and for our model. First however, we will need to make sure the parameters in our model will give the best approximation of the data.

Chapter 5

Model Calibration

Because our goal was to make our model as realistic as possible, we are going to have to also find the most accurate parameters for our λ, μ and θ to calibrate our model. To do this, we will need data for order arrival at distances to the opposing best price. Again we will follow the method proposed by [4] to do this estimation. We start by finding the values S_m, the average size of market orders, S_l, of limit orders and S_c of cancelled orders. Up to now, we have always worked with a unit represented by 1 when adding an order to the order book. Here, we assume S_l is the new value for this unit. Our λ can now be estimated to be:

$$\hat{\lambda} = \frac{N_l(i)}{T_\star}, \qquad \text{for } i \in \{1, \dots, n\} \tag{5.1}$$

with $N_l(i)$ being the total amount of limit orders at distance i from the opposing best price, and $T_\star$ being the total trading time. This approximation is quite reasonable, seeing as that we define the arrival rate λ as the amount of orders arriving per time unit. However, we need to also conform to our idea of this arrival rate being a power law function in the form of (3.1). To do this, we will use the least-square fit method to find our values for k and α:

$$\min_{k,\alpha} \sum_{i=1}^{n} \left(\hat{\lambda}(i) - \frac{k}{i^\alpha} \right)^2 \tag{5.2}$$

Finding our parameter μ can be done in a similar fashion. Again, as we defined our μ to be an arrival rate, it will again be the amount of market orders during a unit of time. However, we also need to still stick to our order size unit of S_l, resulting in the following value:

$$\hat{\mu} = \frac{N_m}{T_\star} \frac{S_m}{S_l} \tag{5.3}$$

with N_m being the total amount of market orders during the time period $T_\star$.

When attempting to estimate our parameter θ, because this cancellation rate is linked to the amount of orders at the price level in question, we start by estimating the steady-state shape of the order book Q_i, which is the average number of orders at distance i of the opposing best price. Through the assumption that orders have to be of unit size, we have:

$$Q_i = \frac{S(i)}{S_l(i)}$$

where $S(i)$ is the average volume at distance i from the opposing best price. Using these values we can make an estimation for $\theta(i)$:

$$\hat{\theta}(i) = \frac{N_c(i)}{T_\star Q_i} \frac{S_c}{S_l} \tag{5.4}$$

All the data required can be found by looking through all the snapshots of an Level 2 order book for time $T_\star$, where every time an order is placed, a new snapshot of the top of the order book is made. As this data requires a license to access, we used data provided in [1]. Here, data is provided for Vodafone stocks over a period of 16 trading days. Thus, we have for the average order sizes:

	S_l	S_m	S_c
Average	17401	11757	15863

Table 5.1: Average number of limit orders, market orders and cancellations

Next, we are given the values for the total number of limit orders and cancellations at distance i from the opposing best price, as well as the amount of market orders during the 16 day period:

i	1	2	3	4	5	6	7	8	9	10
$N_l(i)$	168434	270302	155195	97001	57428	44719	41023	21415	12672	8792
$N_c(i)$	252976	233402	169534	93442	54942	46192	43516	21875	13371	8062
N_m	18042									

Table 5.2: Total number of limit orders, market orders and cancellations

We also find for Q_i, the steady-state shape of our order book, the following values:

i	1	2	3	4	5	6	7	8	9	10
Q_i	4.2512	7.4415	9.5656	10.6088	11.6250	12.8369	11.7427	11.6372	10.8244	10.7222

Table 5.3: Average volume at distance i from opposing best price

Now, using (5.1), (5.3) and (5.4), we can use the given values to find our parameters we will use in our simulation. We get:

i	1	2	3	4	5	6	7	8	9	10
$\hat{\lambda}(i)$	0.3440	0.5521	0.3170	0.1981	0.1173	0.0913	0.0838	0.0437	0.0259	0.0180
$\hat{\theta}(i)$	0.1108	0.0584	0.0330	0.0164	0.0088	0.0067	0.0069	0.0035	0.0023	0.0014
$\hat{\mu}$	0.0273									

Table 5.4: Values for $\hat{\lambda}, \hat{\theta}$ and $\hat{\mu}$

The value for $\hat{\lambda}$ is not fully correct as we still need to fit a power law function. So we use (5.2) to find the values of k and α for the power law function that lies closest to our found values. These are:

$$k = 0.4721 \text{ and } \alpha = 0.6923$$

We can now incorporate these parameters into our model and test to see if there are likenesses to the real data

Chapter 6

Comparing data and model

6.1 Open-Close price comparison

Now, we want to see how accurate our models are by comparing them to real data for Vodafone stocks. We do this by looking at the distribution of the shift in stock price over a trading day, which, seeing as we have a model for both regular trading and the pre-opening phase, we can split up in both these parts. We will start by comparing the part of the model that should be the most realistic, namely the model for the main trading session. This is because the model used here is based on research papers for the arrival rates and the calibration of the parameters.

So, we have a model and the parameters from the Vodafone data, so we can easily simulate a trading day just like we did in Section 3.2. However, now we won't be looking for the entire price path of P_{mid}, but just the final value, which can be considered to be the closing price of the day. We can now also find real data for opening and close prices through Yahoo Finance for Vodafone stock. From this data, we subtract the closing price from the opening price for 200 trading days to find a data set containing the shift of the stock price over one day.

We can then find a similar data set through our model. We start by simulating a trading day, so first a pre-opening phase to create a valid starting point, followed by 30600 seconds worth of trading. The duration of the pre-opening phase is not really important, nor is the precise value of the opening price it generates, as we are only interested in the difference between this value and our closing price, which relies on the price path simulated by our model for regular trading. So, we chose a price grid of size 100, and give it $CP = 50$ so that it starts in the center of the grid. We then store the P_{mid} from the order book created after the execution of all the possible orders after the opening auction, and the P_{mid} at the end of the trading day. This is repeated for 200 times in Matlab, followed by creating an array containing the second P_{mid} subtracted by the first for each of the 200 trading days simulated.

We now have two data sets that we can compare by making a histogram. The only problem is that at this stage we still have a price unit of 1, which is obviously way too large, as we can see in Figure 6.1:

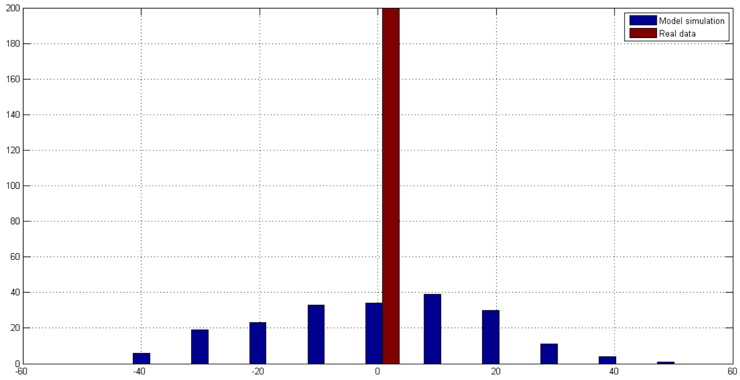

Figure 6.1: Histogram of both the simulated and the real data

So all we need to do now is look for the price unit for which the histograms fall in the same bins. A quick look at the data tells us that a unit size of 0.015 provides this, and we get the following figure:

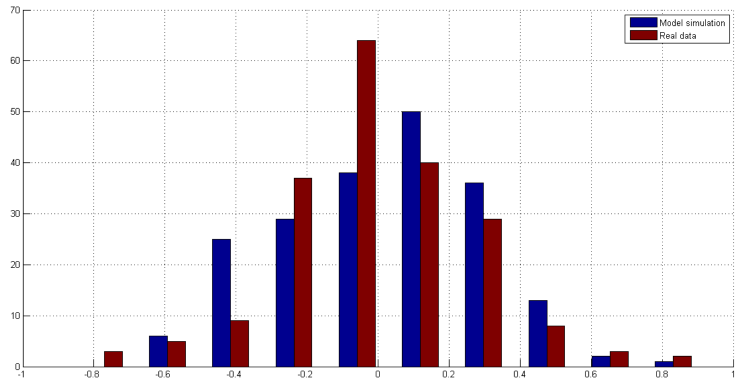

Figure 6.2: Histogram of the adjusted simulated data and the real data

We can see that, although it is not a perfect fit, the shape of both data sets are very similar to the shape of the normal distribution. Another thing we need to realize is that we are working with a model that works simply by generating exponential times for arrival times, whereas the real Vodafone stock is prone to fluctuation that can't really be predicted, yet our model still provides a nearly identical distribution. It is therefore fair to say that, although it is not perfect and should not be used by real traders in an attempt to predict price paths, with some more precision for the parameters it is definitely useful

6.2 Close-Open price comparison

Next, we can test how our proposed model for the opening auction compares to real data. So we are looking at the price change that happens during the opening auction depending on the closing price of the day before. In the previous comparison we started by simulating the opening auction for the opening price and then the rest of the trading day for the closing price. There, we said the duration of the opening auction was not really important, as it had no effect on the distribution we were looking at there.

Here however, if we take the time to be 6300 seconds, or the hour and 45 minutes the opening auction takes for the Euronext, we can use the first P_{mid} found for our comparison here, as it is the opening price, and we started with a fixed closing price of $p = 50$ of the day before. So now we can simulate 200 opening auctions with $CP = 50$ and again $n = 100$ for our price grid, then we can look at the distribution of $(P_{mid} - CP)$. For the real data, we do the same of subtracting the closing price of the previous day by the opening price for 200 days. Like before, we can't compare these distribution yet, as we still need to find a correct price unit size for the histogram to be nearly the same width. Here, we find a value of 0.3 for this price unit and we get the following histogram:

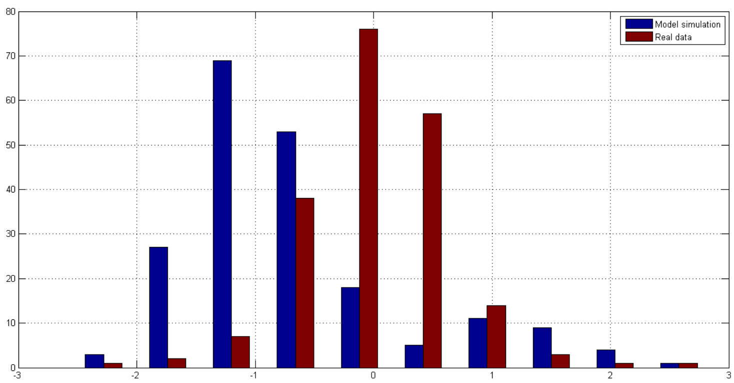

Figure 6.3: Histogram of the adjusted simulated data and the real data

It is clear that this model is a lot less accurate than the last. Both distribution again have the shape of the normal distribution (with the simulated data edging towards a Poisson distribution), however whereas the real data is centered around zero, the simulated data has shifted slightly to the negative side, implying that on average the price goes down from one day to the next. So, as expected, this model is not as good as the previous one, and still requires more calibration for it to be useful when trying to predict price paths. We can blame this inaccuracy on the fact that we had no good function λ for the arrival rate of the limit orders. Seeing as that we made the assumption that by mirroring the λ function used with regular trading over the mid price, it was likely that this would not be a good representation of the arrival rate of real orders.

Chapter 7

Summary

We started by using the model described in [4] to simulate the price path of a trading day. We then altered this model to simulate the pre-opening phase of the trading day. Next, we used [4] again to find the parameters corresponding to Vodafone data to calibrate our model. We then generated data with our models using these parameters to look at the distribution of price changes during the trading day. When comparing this to real Vodafone price changes over 200 days, we found that our model for regular trading was pretty accurate as it was similar in both size and distribution to the real data.

Our second model however was not as accurate. We found that here the distribution for the price change shifted to the negative side instead of being centered around zero. This gives us enough reason not to trust this model and tells us we will need a new function for the arrival rate of the limit orders, as with the one used here, there is too much inaccuracy.

It should be noted though that for the parameter estimation, we made use of values from [1] which in turn found these values from thousands of order book snapshots. As we have no access to these snapshots, we have no way of checking whether these values are 100% correct, as this could change the parameters we would have had to use in the comparison of our models, which could be the cause for the inaccuracy in our second model. However, we will just have to assume these values are correct and our model has to be modified.

7.1 Ideas for future research

In a similar style to what is done in [2], it would be an idea to look at patterns in data which could implicate if a power law fit for the arrival rate for limit orders during the opening auction is possible, which in theory should make our model a lot more similar to real data.

Appendix

addorder.m

```
function [X,ap,bp] = addorder(Y,p,t1,t2)
    %Y = matrixrepresentation of X until previous timestep
    %p = price of the order
    %t1 = type order (1 = sell , 2 = buy)
    %t2 = type order (1 = limit , 2 = cancellation)

    B=length(Y);
%      if 1<=p<=B
        bp=find([-1,Y]<0,1,'last')-1;
        ap=find([Y,1]>0,1);
        X=Y;
            if t2==1
                if t1==1
                    if p>bp
                     X(p)=X(p)+1;
                    else
                      X(max(bp,1))=X(max(bp,1))+1;
                    end
                elseif t1==2
                    if p<ap
                      X(p)=X(p)-1;
                    else
                      X(min(ap,B))=X(min(ap,B))-1;
                    end
                end
            elseif t2==2
                if t1==1
                    if p>bp
                       X(p) = max(0,X(p)-1);
                    end
                elseif t1==2
                    if p<ap
                        X(p) = min(0,X(p)+1);
                    end
                end

            end
      bp=find([-1,X]<0,1,'last')-1;
      ap=find([X,1]>0,1);
end
```

orderbook.m

```
clear all
clc
```

```
n=100;
endtime=3600;
X=zeros(1,n);X=addorder(X,57,1,1);[X,ap,bp]=addorder(X,44,2,1);
i=1:n;k=1.92;a=0.52;
lambda=k*i.^(-a);mu=1.2;theta=[0.71,0.81,0.68,0.56,0.47*ones(1,n-4)];
R=Inf*ones(4,n);
R(1,:)=exprnd(1./lambda);
R(2,1)=exprnd(1/mu);
R(3,(1:n-bp))=exprnd(1./(theta(1:n-bp).*abs(X((bp+1:n)))));
R(4,(1:ap-1))=exprnd(1./(theta(1:ap-1).*abs(X(ap-(1:ap-1)))));
t(1)=0;
Pmid(1)=(ap+bp)/2;
N=2;

while t<endtime
    t1=randi(2,1);
    [x,p]=find(R==min(min(R)));
    m=R(x,p);
    if x==1
        R=R-m;
        if t1==1
            if p+bp<n+1
                [X,ap,bp]=addorder(X,bp+p,t1,1);
                if R(3,p)==Inf
                    R(3,p)=-log(rand(1))/(theta(p)*abs(X(bp+p)));
                end
            end
        elseif t1==2
            if ap-p>0
                [X,ap,bp]=addorder(X,ap-p,t1,1);
                if R(4,p)==Inf
                    R(4,p)=-log(rand(1))/(theta(p)*abs(X(ap-p)));
                end
            end
        end
        R(1,p)=-log(rand(1))/lambda(p);
    elseif x==2
        if t1==1
            [X,ap,bp]=addorder(X,max(bp,1),t1,1);
        elseif t1==2
            [X,ap,bp]=addorder(X,min(ap,n),t1,1);
        end
        R=R-m;R(2,1)=-log(rand(1))/mu;
    elseif x==3
        R=R-m;
        if p+bp<n+1
            [X,ap,bp]=addorder(X,bp+p,1,2);
            R(x,p)=-log(rand(1))/(theta(p)*abs(X(min(bp+p,n))));
        else
            R(x,p)=-log(rand(1))/(theta(p)*abs(X(n)));
```

```
        end
    elseif x==4
        R=R-m;
        if ap-p>0
            [X,ap,bp]=addorder(X,ap-p,2,2);
            R(x,p)=-log(rand(1))/(theta(p)*abs(X(max(ap-p,1))));
        else
            R(x,p)=-log(rand(1))/(theta(p)*abs(X(1)));
        end
    end
    Pmid(N)=(ap+bp)/2;
    t(N)=t(N-1)+m;
    N=N+1;
end
%plot(Pmid,t)
%bar(X)
```

openingsession.m

```
clear all
clc
n=100;cp=50;endtime=6300;
k=1.92;a=0.52;
x1=[k*(cp+2-(1:cp+1)).^(-a),k*((cp+2:n)-cp).^(-a)];
x2=[k*(cp-(1:cp-1)).^(-a),k*((cp:n)-cp+2).^(-a)];
x3=0.1;
x4=0.5*[0.47*ones(1,n/2-4),0.56,0.68,0.81,0.71,0.71,0.81,0.68,0.56,0.47*ones(1,

R=Inf*ones(5,n);X1=zeros(1,n);X2=zeros(1,n);X3=zeros(1,2);
R(1,:)=exprnd(1./x1);
R(2,:)=exprnd(1./x2);
R(3,1)=exprnd(1/x3);
R(4,:)=exprnd(1./(x4.*X1));
R(5,:)=exprnd(1./(x4.*X2));
time1=0;

while time1<endtime
    [x,p]=find(R==min(min(R)));
    m=R(x,p);
    R=R-m;
    if x==1
        X1=addorder(X1,p,1,1);
        R(1,p)=-log(rand(1))/x1(p);
        if R(4,p)==Inf
            R(4,p)=-log(rand(1))/(x4(p)*abs(X1(p)));
        end
    elseif x==2
        X2=addorder(X2,p,2,1);
        R(2,p)=-log(rand(1))/x2(p);
```

```
            if R(5,p)==Inf
                R(5,p)=-log(rand(1))/(x4(p)*abs(X2(p)));
            end
        elseif x==3
            t1=randi(2,1);
            X3=addorder(X3,(t1==1)+1,t1,1);
            R(3,1)=-log(rand(1))/x3;
        elseif x==4
            X1=addorder(X1,p,1,2);
            R(4,p)=-log(rand(1))/(x4(p)*abs(X1(p)));
        elseif x==5
            X2=addorder(X2,p,2,2);
            R(5,p)=-log(rand(1))/(x4(p)*abs(X2(p)));
        end
        time1=time1+m;
end
for i=1:n
        if (-sum(X2(i:n))-X3(2))>=(sum(X1(1:i))+X3(1))
            p=i;
        end
end
bp2=find([-1,X2]<0,1,'last')-1;
ap1=find([X1,1]>0,1);
while X3(1)~=0
        m=min(abs(X3(1)),X1(ap1));
        X3(1)=X3(1)+m;
        X1(ap1)=X1(ap1)-m;
        ap1=ap1+1;
end
while X3(2)~=0
        m=min(X3(2),abs(X2(bp2)));
        X3(2)=X3(2)-m;
        X2(bp2)=X2(bp2)+m;
        bp2=bp2-1;
end
bp2=find([-1,X2]<0,1,'last')-1;
ap1=find([X1,1]>0,1);
while bp2>=ap1
        Y1=X1(ap1);Y2=X2(bp2);f=abs(Y2)>=Y1;g=abs(Y2)==Y1;
        X2(bp2)=Y2-(1-f)*Y2+f*Y1;
        X1(ap1)=Y1+(1-f)*Y2-f*Y1;
        bp2=bp2-(1-f)-g;
        ap1=ap1+f;
end
X=X1+X2;ap=ap1;bp=bp2;
Pmid(l)=(ap+bp)/2;
%bar(X)
```

Bibliography

[1] C.J.L de Abreu: *Stochastic Modeling of Order Book Dynamics*, Delft (2011)

[2] J.-P. Bouchaud, M.Mézard & M.Potters: *Statistical properties of stock order books: Empirical results and models* (2002)

[3] S.N. Cohen, L. Szpruch: *A limit order book model for latency arbitrage* (2000)

[4] R. Cont, S. Stoikov & R. Talreja: *A Stochastic model for order book dynamics*, IEOR Dept, Columbia University, New York (2008)

[5] M.D. Gould et al.: *The Limit Order Book: A Survey* (2013)

[6] W. Huang, C.-A. Lehalle, M. Rosenbaum: *Simulating and analyzing order book data: The queue-reactive model* (2013)