Statistical Modeling and Simulation of Limit Order Markets

Felix Prenzel
St Hughs's College
University of Oxford

A thesis submitted for
Doctor of Philosophy

Trinity 2023

Acknowledgements

If somebody had told me ten years ago I would one day obtain a doctoral degree, let alone from Oxford, I would not have taken them seriously. So I want to take this opportunity to thank some of the people who have supported and pushed me towards this path.

I want to thank my mother Bettina and my sisters Theresa, Kathrin, and Julia for their continuous support and motivation throughout my studies. Special thanks to Julia for reading so much of my written work and giving me so much feedback.

A big shoutout to our cohort and all my friends I have been lucky to make and share time with over the past years in Oxford. You guys have turned this journey from an academic one also into a one of developing friendships that will last far beyond my DPhil. Special thanks to Alain and Jonathan who formed a fundamental part of this.

I want to further thank my former mentors, particularly during my undergraduate and graduate studies and my time at European Central Bank for pushing me towards the decision of doing a DPhil and also supporting my applications with recommendation letters.

My utmost gratitude goes to my supervisors, Rama Cont and Mihai Cucuringu, whom I want to thank for their mentorship and guidance throughout my DPhil, countless meetings, idea generation, brainstorms, and support. I further want to thank my industrial supervisors Jonathan and Vacslav for their invaluable input during our discussions thereby giving a whole new perspective to my projects.

Abstract

This thesis focuses on the statistical modeling of order flow in limit order markets and the development data-driven approaches for the simulation of limit order book dynamics.

In the first part, after introducing various mathematical representations of limit order books (LOB) reflecting different degrees of granularity and information, we investigate the heterogeneity of order flow submitted through brokers using proprietary execution data and unsupervised learning techniques. This results in a statistical description of client order flow as a superposition of four components representing four heterogeneous types of agents – *Quantitative*, *Day VWAP*, *Signal* and *Res* – which differ through their trade frequency, intraday activity patterns and order sizes.

The second part of the thesis develops data-driven simulation methods for limit order book dynamics. We first present a generative model for transitions of limit order book snapshots using a generative adversarial network (GANs). The model allows efficient simulation of snapshot time series reproducing desired properties and furthermore automatically reflects market impact when interacting with the order book state. Lastly, we propose a hierarchical approach to improve existing LOB simulation methods. In particular, we present a probabilistic model to generate calibrations of order flow models. This preserves theoretical properties of the underlying base model and allows to incorporate realistic features of intraday dynamics such as U-shaped intraday seasonality and trends, volatility dependency and market disruptions into LOB simulations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the past decades, the use of order-driven markets – also called limit order books (LOBs) – has skyrocketed as have trading volumes in such markets. Nowadays, the majority of orders are being sent to an exchange in an automated fashion. This growth has made it more important than ever to better understand and interact in such an environment.

The concept of a limit order book has its origins in the early days of securities trading when brokers' traders used to gather around on the trading floor of an exchange manually executing trades for their clients. However, as trading volumes increased and trading via computer algorithms intensified, a more systematically organized and efficient system was developed to match buyers and sellers.

An electronic limit order book can be understood as a record of all outstanding limit orders for a particular financial instrument, such as a stock or futures and cryptocurrencies. These limit orders indicate the willingness of a trader/agent to buy or sell a certain quantity of the corresponding instrument. With a limit order book, buyers and sellers can enter their orders directly into the system, specifying the quantity and the price of a corresponding stock they are willing to buy or sell. The LOB consists of a buy (bid) side and a sell (ask) side. The best available bid (highest price of a buy limit order) and ask (lowest price of a sell limit order) prices form important reference prices at any given time.

Limit orders then "sit" in the book until they are either canceled or executed against an incoming market order. Market orders, also called executable limit orders, are executed immediately – given there is enough liquidity – but the buyer has to pay a markup called "spread" for immediate execution. The limit order book engine matches the incoming market order with limit orders in the book based on price and time priority. This means an order at a more favorable price (e.g. higher for a buy

order) will be matched first. If there are several orders at the "best" price level, the order with the earliest submission time gets matched first.

LOBs have led to trading with more transparency and efficiency. Furthermore, the use of LOBs have caused market participants such as brokers, market makers, and proprietary trading firms to develop sophisticated trading algorithms. These algorithms automatically analyze and respond to changes in the market by trading accordingly. They can be designed to identify patterns and execute trades more quickly and efficiently than human traders. Designing such algorithms and the necessity to better understand the newly designed markets are part of the reason for a vast amount of research and literature in the field of limit order books and algorithmic trading.

One reason for this quantity of research is the challenges when acting in limit order markets. The sheer amount of market participants acting together in one system having different intentions and views on the current market and its future development makes LOBs very challenging to interact with. This is partially because each action an agent takes in the market leads to a response to it. Furthermore, high-frequency trading algorithms that rely on LOB data have caused sudden price movements and/or increased volatility, which may be harmful to market stability and even lead to flash crashes as in 2010 (see (Kirilenko, Kyle, Samadi, & Tuzun, 2017)). Also, the systems of limit order books have attracted market participants that try to exploit other agents acting in the market, trying to front run them or to display demand/supply for a stock that is not intended to be executed (so-called spoofing (Á. Cartea, Jaimungal, & Wang, 2020)).

While there are many players in the LOB market, two main players are particularly concerned with the efficient acting in limit order books. These are brokers and market makers (MMs), both with fundamental functions in electronic markets.

Execution services or brokers are financial intermediaries who facilitate the buying and selling of financial instruments on behalf of their clients. Trading a large quantity of a certain asset is not always easy if there is not enough liquidity in the market, for instance, due to low trading volumes. Executing a large trade can thus be difficult and potentially implies large transaction costs and impact on the price. Also, many participants do not have direct access to an exchange. Brokers provide this access to financial markets and help clients execute their trades based on the preferences of their clients, using proprietary algorithms.

When a client wants to buy or sell a financial instrument, they submit an order (also called parent or meta order) to their broker or execution service. The broker then

processes this order and creates an execution schedule. According to this timeline, the broker then executes the trade by sending child limit or market orders to the appropriate exchange or market trading at the prevailing market price. In addition to executing trades, brokers also provide other services such as market research, order management, and risk management to their clients.

Some brokers specialize in different types of financial instruments, such as stocks, bonds, options, futures, or currencies. They may also focus on specific markets, such as domestic or international markets. Some brokers cater to retail clients, while others specialize in serving institutional clients such as hedge funds or pension funds.

As a reward, brokers earn a commission/fee for their services, typically a certain percentage of the trade value. In some cases, brokers may also earn revenue from interest on client deposits or from providing margin financing to clients. The quality of execution services can have a significant impact on the overall performance of an investment portfolio, as well as the cost of trading. As a result, many investors carefully evaluate the execution quality and fees of their brokers or execution services.

Market makers, the second category of major players in LOBs also called liquidity providers, make markets. This means they constantly post prices at which they are willing to buy or sell a given asset. In LOBs they do this by posting limit orders, thereby providing liquidity. Market makers are supposed to facilitate trading in a particular market by creating this liquidity, and buying and selling an asset at any time, also when other actors do not. The prices MMs post are generally close to the current market price.

Market makers play an important role in ensuring a continuous flow of trading in a market by offering to buy or sell financial instruments at any time during the trading day. This continuous flow helps to reduce the bid-ask spread, which is the difference between the highest price a buyer is willing to pay and the lowest price a seller is willing to accept. By narrowing this spread, market efficiency is increased and trading costs for investors generally tend to decrease.

In return for providing liquidity, market makers earn a profit from the spread between the buying and selling price of the financial instruments they trade – a so-called round trip trade. This profit, known as the market maker's markup, compensates them for the risk they take by holding an inventory of securities and being willing to buy and sell them on demand. During these generally short times of inventory, market makers run the risk of price moving against them. In these situations, the round trip trade is not profitable. Depending on this risk, market makers set their prices and this is also why in periods of high uncertainty/volatility, spreads tend to

be wider as MMs seek a higher reward for providing liquidity in difficult states of the market.

Both market makers and brokers have to act efficiently, which, however, comes with many challenges that must be tackled. These problems in particular concern topics like future price prediction and the impact a particular trade execution has (so-called transaction cost analysis). Fill probability modeling of limit orders, i.e. the question of how likely an order is to be executed within some time period, is a further crucial detail that brokers and market makers have to model. Around this, the mathematical frameworks of optimal execution as well as optimal market making were developed under heavy use of stochastic optimal control.

The complex dynamics of LOBs make it very challenging to correctly model price prediction, impact of executions, etc. which in turn complicates the optimal interaction with LOBs, e.g. executing large trades or building robust trading strategies and market making algorithms. Furthermore, since every action and thus change in the LOB triggers reactions from other market participants, the pure use of historical data does not provide realistic results when, for example, creating execution algorithms. Due to all these difficulties, modeling and simulation of LOBs is regarded as a highly impactful problem for both industry and academia. That is particularly because realistic simulators enable academics and practitioners to perform realistic controlled experiments in an artificial environment, facilitating the development of trading strategies, execution algorithms, and more. This, however, presumes that the simulators are "good enough" to be deployed, in particular for an industry where potentially large amounts of money are at stake.

Building realistic models to simulate LOBs has thus received a lot of attention in the existing literature. The absence of a clear metric indicating what can be considered "realistic" aggravates the problem's difficulty since researchers or practitioners can not just fit a model minimizing this metric. A range of the statistical properties that LOBs exhibit are well studied in the literature (Bouchaud, Mézard, Potters, et al., 2002; Cont, 2001, 2011) and commonly aimed to be reproduced in studies presenting methods to model LOBs.

Progress in machine learning and advances in computational power now allow the application of data-driven, non-parametric approaches on a much larger scale than just a few years ago. Despite their lack of analytical tractability, machine learning algorithms tend to be more flexible and rely on fewer assumptions than conventional models, which potentially allows them to capture more complex properties observed in LOBs. E.g., (J. Sirignano & Cont, 2019) find that accounting for path dependencies

is important for price prediction. Machine learning techniques are able to account for these dependencies and reproduce them if designed to do so. The literature regarding the use of data-driven methods to generate financial data, in particular LOBs, is still very limited. This thesis leverages novel, data-driven methods in order to gain a deeper understanding of financial markets and their ecology, as well as improve simulation methods of such.

1.1 Outline and contributions

Outline The remainder of the thesis is structured as follows. Section 1.2 gives a comprehensive introduction to the literature on limit order books, high-frequency data, and the research being conducted in this field. Section 1.3 and Section 1.4 introduce both sub-fields, i.e. statistical modeling and simulation of limit order book markets. An introduction to LOBs, including a novel, more detailed notation accounting for the heterogeneity of agents is presented in Chapter 2 along with existing models for order flow. Some of them are used in the presented projects. Three main projects, building the core of this thesis and contributions to the literature, follow in Chapter 3, Chapter 4, and Chapter 5. A conclusion including major challenges in the field as well as future research directions is given in Chapter 6.

The contributions to the literature can be summarized as follows:

1. Chapter 2 introduces a new representation of limit order books based on (Cont, Cucuringu, Glukhov, & Prenzel, 2023) which allows for more detailed views on the limit order book. Three main views, the *anonymized view*, the *omniscient view*, and the *broker view* are introduced.

2. Chapter 3, based on (Cont, Cucuringu, Glukhov, & Prenzel, 2023), studies the heterogeneity of agents submitting parent orders to brokers (client order flow) leveraging proprietary execution data. In particular, we analyze the heterogeneity in client order flow in limit order markets and extract representative agent types that can be interpreted and exhibit strong differences in their behavior. Furthermore, we show that heterogeneity is stable both in time as well as over different instruments. Finally, we present an approach for a parent order model which efficiently captures the main sources of heterogeneity.

3. Chapter 4, based on (Cont, Cucuringu, Kochems, & Prenzel, 2023), puts the challenge of LOB simulation in a simpler setting of generating LOB snapshots. We design a probabilistic model based on generative adversarial networks

that learns the conditional distribution of LOB snapshot transitions. Properly trained, the model reproduces a range of properties observed in the data and entire paths of LOB snapshots can be generated. At the same time, the GAN approach is computationally much faster than event-based simulation. Lastly, experiments investigating the response of the model to systematic interaction show that the model automatically learns market impact. It does so for both liquidity extraction and provision and exhibits patterns that are well-known in the literature.

4. In Chapter 5, based on (Prenzel, Cont, Cucuringu, & Kochems, 2022), we discuss why traditional order flow models potentially suffer from stationary/global parameters. We then build a generative model to produce unseen, realistic calibrations of such models that can be based on any conditions that are uncovered and automatically learned by the model directly from the data. Using the probabilistic model's generated calibrations, we demonstrate that using dynamic calibrations instead of global calibrations leads to well-known non-stationary effects in the simulated data and more diverse simulations in general.

1.2 Related literature

The advent of electronic limit order books and algorithmic trading has generated burgeoning research literature related to the study and modeling of data emanating limit order markets.

Overview The limit order book has roughly been around since the beginning of the 2000s. Before exchanges adopted LOBs, quote-driven markets were dominating. An overview is given in (Cont, 2011; Gould et al., 2013). Both introduce limit order books themselves, what different order types exist, and how limit orders are matched with incoming market orders (also called executable limit orders). They also provide an adequate overview of the literature until about 2014. (Gould et al., 2013) review empirical observations regarding order flow or market impact profiles and stylized facts such as heavy-tailed return distributions or long-term memory in the order flow. They further give an overview of modeling LOBs from perfect-rationality approaches over zero-intelligence approaches to agent-based models. The article concludes with unresolved problems in the LOB literature, such as understanding regularities, more recent data, and non-stationary behavior.

Empirical and statistical properties The exploration of empirical and statistical properties of limit order books and financial time series has been the main focus of many articles, most commonly (Cont, 2001, 2011; Bouchaud, Farmer, & Lillo, 2009; Bouchaud et al., 2002). (Cont, 2001) describe "distributional properties, tail properties and extreme fluctuations" as well as "pathwise regularity, linear and nonlinear dependence of returns" for time series in general. The paper further discusses how many of the properties stand against the assumptions of "common statistical approaches". Empirical results and models of stock order books are discussed in (Bouchaud et al., 2002). The study focuses on the statistics of the prices of limit orders and the shape of the order book claiming that the latter can be "reproduced using a 'zero intelligence' numerical model". Further, they describe how the number of incoming limit orders decreases with distance from the best prices in a particular pattern and investigate order and queue size distributions. (Potters & Bouchaud, 2003) follow up on (Bouchaud et al., 2002), also focusing on the lifetime of limit orders depending on the depth of the book. (Cont, 2011) review stochastic models for dynamics of the LOB in continuous time. Models describing the LOB as a queuing system are the focus. (Bouchaud et al., 2009) investigate how "fluctuations in supply and demand are slowly incorporated into prices", arguing that many large orders executed over longer time horizons lead to a long memory effect in the LOB, with price formation being the center of the study.

Market ecology Studies on market ecology are very sparse as proprietary data is generally needed to track the origins of different orders. Nonetheless, several studies exist which investigate the effect of certain market participants, mostly high-frequency traders and market makers on e.g. spread, liquidity, etc. (Brogaard et al., 2010; Brogaard, Hendershott, & Riordan, 2014; Hagströmer & Nordén, 2013; Hasbrouck & Saar, 2013; Van Kervel & Menkveld, 2019). (Brogaard et al., 2010, 2014) investigate a data set in which orders submitted HFTs are flagged to assess their impact on market quality and price discovery. They find generally positive effects on price discovery and claim HFTs potentially "dampen intraday volatility". (Hagströmer & Nordén, 2013) outline the differences between HFTs and MMs in the market ecology of limit order books. (Hasbrouck & Saar, 2013) attempt to identify HFTs based on "strategy runs" to develop their own measure of liquidity and investigate the impact of such strategy runs. (Toth et al., 2012) look into the question of how markets react to order flow finding brokers act very heterogeneously and that they depend on the behavior of other brokers. (Barbon, Di Maggio, Franzoni, & Landier, 2019; Di Maggio, Franzoni,

Kermani, & Sommavilla, 2019; Hendershott, Li, Livdan, & Schürhoff, 2020) investigate the information content of broker order flow, arguing whether the traded volume brokers' preferred clients increases during large liquidations. The most detailed study on market ecology leveraging non-public data is performed in (Kirilenko et al., 2017) carefully analyzing the flash crash using a classification method for trading accounts based on an algorithm in (Mankad, Michailidis, & Kirilenko, 2013).

Prediction Prediction is yet another central part for practitioners. Brokers and market makers can both improve their decision making when having more information about where the price (or best prices) will move within some future time horizon to adjust their strategy. Classical methods such as the ARIMA model (see (Hamilton, 2020)) have mainly been used to predict daily time series, e.g. (Ariyo, Adewumi, & Ayo, 2014) under the discussion to which extent markets are at all predictable (Ang & Bekaert, 2007; Malkiel, 1989, 2003). Due to the high dimensionality of LOB data and the rise of computational power, machine learning techniques have recently dominated the forecasting field. (Kercheval & Zhang, 2015) forecast limit order book dynamics with support vector machines (SVMs). In particular, they perform multi-class prediction for mid-price movements and price spread crossing based on hand-crafted features.

Deep Learning (DL) techniques, i.e. neural networks (NNs), have the advantage of being able to extract features from raw, noisy input data and thus have been applied in many studies such as (J. A. Sirignano, 2019; J. Sirignano & Cont, 2019; Zhang, Zohren, & Roberts, 2019; Zhang, Lim, & Zohren, 2021; Zhang & Zohren, 2021; Lucchese, Pakkanen, & Veraart, 2022). (J. A. Sirignano, 2019) develop a new architecture called "spatial neural network" tailored for the spatial structure of limit order book data. (J. Sirignano & Cont, 2019) perform a large scale experiment under the use of Long Short Term Memory (LSTM) neural networks finding that price features in LOB appear to be universal and thus global models outperform ticker-based models. Furthermore, they claim that accounting for path dependency substantially increases the results of the predictions. LSTMs are combined with convolutional neural networks (CNNs) and inception modules in order to classify future price moves into three classes by (Zhang et al., 2019). The problem of forecasting multiple horizons is addressed in (Zhang & Zohren, 2021). Market by order (MBO of L3) data is leveraged in (Zhang et al., 2021) to improve forecasting power. (Lucchese et al., 2022) review different state-of-the-art architectures under the main questions of how far predictability exists and the importance of a robust representation of the limit

order book which is also addressed in (Y. Wu, Mahfouz, Magazzeni, & Veloso, 2021). Overall, research applying deep learning techniques to price forecasting future price movements in LOBs is still ongoing and yet, no clear consensus has been found on which method is best.

Models for dynamics Modeling specific dynamics of the order book using e.g. stochastic processes is another central part of the literature. Price changes and their (time-varying frequency), the arrivals of orders, the impact of order book events, or modeling the occurrence of trades have been at the center of this research direction.

(E. Smith, Farmer, Gillemot, Krishnamurthy, et al., 2003; Cont, Stoikov, & Talreja, 2010; Cont & De Larrard, 2013) study dynamics of the LOB in which order arrivals are modeled in a Markovian queueing system with endogenous price dynamics. (Cont & De Larrard, 2013) sheds light into the "relation between order flow and price dynamics". Models for the impact of order events are built in (Hautsch & Huang, 2012; Cont, Kukanov, & Stoikov, 2014; Eisler, Bouchaud, & Kockelkoren, 2012). More recently, (Cont, Degond, & Xuan, 2023) present a mathematical framework for the modeling of order book dynamics.

Hawkes processes – introduced in (Hawkes, 1971) – are another class of point processes used to model certain dynamics of the Limit Order Book. (Bacry, Mastromatteo, & Muzy, 2015) provide an overview of Hawkes processes applications in finance. Regarding high-frequency data and LOBs, (Bacry, Delattre, Hoffmann, & Muzy, 2013) model the microstructure noise in high-frequency data leveraging Hawkes processes. (Bacry, Iuga, Lasnier, & Lehalle, 2015; Bacry & Muzy, 2014) build market impact models based on Hawkes processes, (Bacry, Iuga, et al., 2015) do so by using proprietary broker data. (Hewlett, 2006) were the first to apply Hawkes processes to the optimal execution of a large amount of stocks modeling the arrival of buy and sell market orders with a Hawkes process. More recently, the problem of optimal execution using Hawkes processes has been studied in (Alfonsi & Blanc, 2016).

Optimal execution and market making Many control problems were studied for both execution and market making applications. (Á. Cartea, Jaimungal, & Penalva, 2015) and (Guéant, 2016) cover substantial areas of algorithmic trading. (Donnelly, 2022) recently published an overview of optimal execution.

The "Almgren Chriss Framework" is amongst the earliest and most famous works on optimal execution of large orders (R. Almgren & Chriss, 2001). One of the authors extends the work to non-linear impact functions in (R. F. Almgren, 2003). (Alfonsi,

Fruth, & Schied, 2010) study optimal execution strategies in LOBS with general shape functions. (A. Cartea & Jaimungal, 2015) deal with optimal execution with both limit and market orders. (Guéant, 2017) deal with the problem of optimal market making.

The complexity of the more involved control problems makes optimal execution a suitable candidate for reinforcement learning which does not rely as heavily on the constraints of stochastic optimal control. (Nevmyvaka, Feng, & Kearns, 2006) apply reinforcement learning to trade execution early on. More recently, (Hendricks & Wilcox, 2014) extend the Almgren-Chriss framework (R. Almgren & Chriss, 2001) via reinforcement learning. (Ning, Lin, & Jaimungal, 2021) use double deep q-learning for optimal execution. In general, for reinforcement learning, an agent may only be as good as the environment that is used to train the RL agent. I.e., if the environment lacks some properties or contains some biases, the results of the agent will also lack in these dimensions or learn to exploit biases that do not exist in real data.

Market and Price impact A central part – especially for brokers – is the market impact when executing large trades via several child orders. This is difficult as market impact is not directly observable and cannot easily be measured. Among others, (Gatheral, 2010; Hautsch & Huang, 2012; Gatheral & Schied, 2013) are famous works in this direction. (Gatheral, 2010) summarize the state of the literature and discuss combinations of different impact and decay functions which – under certain assumptions – are free of arbitrage. The study concludes that exponential decay is only free of arbitrage if there is a price impact proportional to trade size. The model is extended in (Gatheral & Schied, 2013) and the optimal execution strategies given the corresponding market impact model. They conclude that there "is price manipulation and no well-behaved optimal order execution strategy" for many models.

Simulation of markets In order to model and simulate the book itself, many approaches are used. Point process models based on Poisson and Hawkes order flow have been studied in (E. Smith et al., 2003; Cont et al., 2010; Cont & De Larrard, 2013). The Poisson approach is extended in (Huang, Lehalle, & Rosenbaum, 2015) to account for queue-dependent dynamics – including different arrival rates for different spreads. More recently, (Morariu-Patrichi & Pakkanen, 2022) model ask and bid events using state-dependent Hawkes processes which depend either on the spread or the queue imbalance. (P. Wu, Rambaldi, Muzy, & Bacry, 2019) build a Hawkes process that depends on the queue, multiplying the Hawkes intensities with queue dependent factor. (Cont & Müller, 2021) model the entire order book state as a

density that evolves over time via a stochastic partial differential equation. They consider cumulative changes rather than event-driven models. An agent-based approach with different agent types such as noise or fundamental trader including a simulation framework is done in (Byrd, Hybinette, & Balch, 2019). More data-driven approaches for order stream simulation with generative models are in (J. Li, Wang, Lin, Sinha, & Wellman, 2020; Coletta, Moulin, Vyetrenko, & Balch, 2022). Both studies attempt to simulate sequences of orders via GANs.

Outlook Limit order books are far from being fully understood and research is still to be conducted in order to improve models of LOBs. In particular, the area of market ecology and dynamics in the order book is not considered as solved. While our study (Cont, Cucuringu, Glukhov, & Prenzel, 2023) sheds some light on the structure of different agents, the overall interaction of agents, also from other brokers as well as HFTs and Market Makers, remains to be investigated. Another field is the market impact of execution strategies which is of ongoing research and difficulty since market impact can not directly be measured and in fact, is of high difficulty. To this end, improving methods to generate synthetic limit order books can help to improve the estimation of market impact and actually simulate realistic scenarios that allow for controlled experiments.

1.3 Heterogeneity of markets

Public data sources of LOBs are generally anonymous. In particular, public limit order book data may have different formats of different granularity called *L1*, *L2*, *L3* data. The majority of the studies hence use public LOB data which - at most - shows orders on a tick level (i.e. order by order). This, however, is just the end of an entire process of which often a lot of limit orders are part of.

Public data does not contain any information on what kind of agent with what intentions submits an order and furthermore whether it is part of a larger "parent order". Figure 1.1 shows the ecology of the market for a given stock. In the first instance, market participants may be separated into two groups:

1. Traders with proprietary/direct access to an exchange
2. Traders without direct access, generally relying on brokers/execution services

The first group primarily contains market participants such as high-frequency traders (HFTs) or market makers (MMs), who trade intensively and thus tend to

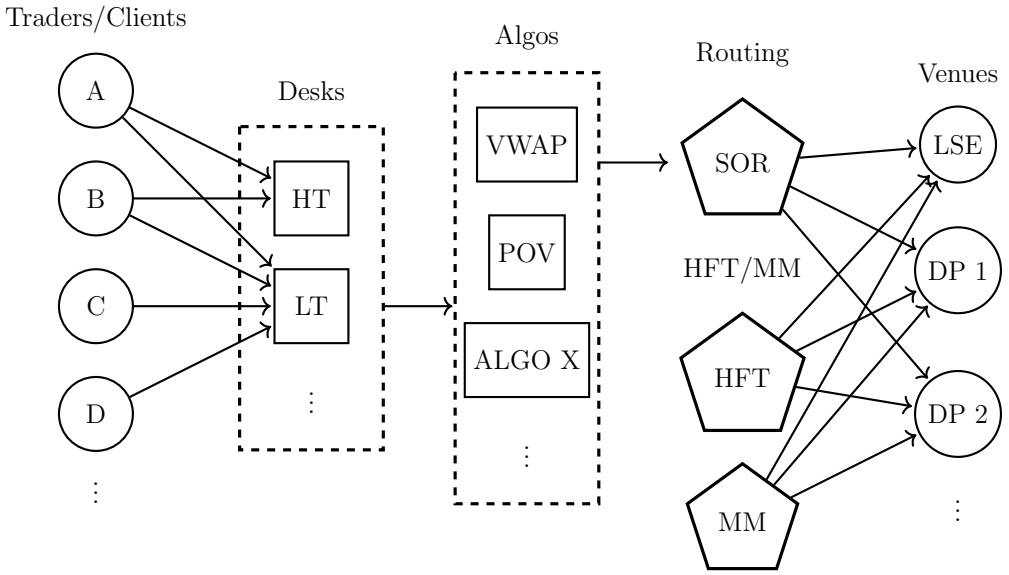


Figure 1.1: Clients submit "parent orders" to execution services and different trading desks. For example, these could be high touch (HT) and low touch (LT) desks. Parent orders are sliced into child orders via "Desks" and "Algos" and routed to different venues for cost-efficient execution. Usually, a routing system (SOR) determines to which venue a child order is sent.

have direct access to the exchange. Such participants – generally equipped with sophisticated trading platforms – skip the execution service and interact directly with different exchange venues.

The second group of market participants are those trading through a broker and are depicted on the left-hand side of Figure 1.1. A market participant (e.g. a quantitative firm or large institutional trader) submits a parent order to a broker or company offering execution services. Depending on the order it may arrive at different desks such as the high touch or the low touch desk shown in the second "column" of Figure 1.1. This broker then processes the client's parent order and executes it via individual child orders which are sent to the LOB of the exchange. This generally happens via leveraging different execution algorithms which are designed to trade differently depending on the preferences of the client. These mostly depend on which horizon a trade shall be executed or how "urgent" the client wants the trade to be done. The algorithms then process the parent order and slice it into different child orders. These child orders are then placed at different "lit" and "dark" venues via a smart order routing (SOR) system, as shown in Figure 1.1. (Kirilenko et al., 2017) show that order flow differs substantially depending on the type of trader it is coming from during extreme events. Publicly available historical data, however, does not

contain any information about the origin of the order but only about the last section when an order is submitted to a venue.

1.3.1 Analysis and modeling of client order flow in limit order markets

The paper "Analysis and Modeling of Client Order Flow in Limit Order Markets" has been published in Quantitative Finance, 2023 Volume. 23, No.2 (Cont, Cucuringu, Glukhov, & Prenzel, 2023).

Studies using proprietary data are rare. Most famously, (Kirilenko et al., 2017) analyze the 4 days of audit trail data from the E-mini S&P 500 stock index futures market – a data set with great granularity. Within the frame of our work, execution data from a major broker is used in order to shed light on the order process from Figure 1.1 explained above. The aim of the project is to understand the structure of the traders (shown on the left-hand side in Figure 1.1) and to investigate whether consistent patterns may be found. The following questions are of particular interest:

1. Can we segment traders acting through execution services in a data-driven way? What groups may be found and can they be interpreted?

2. How stable is a segmentation of traders over time and across different instruments? Are the observations from the segmentation random or persistent across different dimensions?

3. How can insights potentially be used to build LOB models that account for the heterogeneity of the traders?

Unsupervised learning in order to segment the set of traders sending orders for execution to a broker into an optimal partition. More specifically, the agents are clustered based on the properties of their parent orders. These properties, i.e. an agent's behavior of sending parent orders is summarized in an array of features. This feature vector is then used to cluster the agents per ticker and per instrument basis.

Following various metrics as well as the input from experts, the agents are classified into Quant, Day VWAP, Signal and Res (residual) agents. The agents' behavior between different groups and in particular the representative types differ substantially.

1. The Quant agent type typically sends many orders of rather small sizes. This happens throughout the day. Furthermore, the distribution of its net inventory is centered around zero while the other agent types tend to be more inventory accumulating.

2. The Day VWAP agent type sends only very few orders, mostly in the morning around market open. These orders are then executed throughout the day and cancellations happen rarely.

3. The Signal agent type sends rather large trades and does this particularly often in the afternoon. It is noticeable that this order flow type tends to execute more in dark venues and often has a larger percentage of volume (POV) in its trades. This leads to assuming that this flow is based on some sort of signals which leads to more aggressive trading.

4. The Res agent type shows properties from several other types and seems to be located in the center between the first three types. One reason for this is potentially that several strategies are traded with the same account.

In order to check for the stability of the partition and verify the pattern is not a random observation, we both cluster several slices (months) of execution data together and perform meta clustering. Results show that the log-weighted maximum affiliation probability is $75\% - 80\%$. In other words, almost 80% of an agent's observations end up in the same cluster. Meta clustering the first stage clusters indicates that the representative agent types do not vary substantially. The only confusion occurs with Res and another cluster in a few cases. This supports the interpretation that the Res cluster appears to be a mixture of the other three clusters.

Investigating the heterogeneity of the aggregated flows from each agent type further reveals differences in the properties of child orders, correlation with price moves, and the agent types' profitability. In particular, Quant tends to send more child orders of smaller sizes closer to the mid while Day VWAP flow causes orders deeper into the book which is in line with the interpretation of the agents on parent order level. Regarding profitability, the Signal flow appears to outperform the other clusters, especially on a longer horizon. The Quant flow exhibits an excess return on a 1-day horizon significantly different from zero. Correlation of the order flow imbalances with large price moves of the corresponding ticker as well as the STOXX 600 show, that Quant exhibits a rather low correlation to strong moves which is in line with its inventory distribution and the interpretation that the Quant flow tends to pursue more intra-day trading. At the same time, Day VWAP and Res show a positive correlation. The only negative correlation is found for Signal which tends to have a negative order flow imbalance when prices move strongly upwards and vice versa.

Lastly, we present an approach toward a heterogeneous model for the parent order flow of a broker which summarizes the heterogeneity of the different flows. This allows for the construction of heterogeneous order book models when adding the flow of HFTs and MMs by modeling how the parent orders are executed in the LOB with conventional algorithms. Furthermore, it stands to question whether assuming homogeneous order flow is a valid assumption for other conventional models. From an empirical perspective, the insights provide a better understanding of limit order markets and their heterogeneity. To which extent heterogeneous LOB models outperform other conventional methods, and for which tasks they may be more suitable is one of the many future research directions.

1.4 Simulation of limit order books

Over the past years and the increasing use of computational and machine learning methods, the use and applications for synthetic data have gained a lot of interest. Synthetic data is artificially generated data that exhibits the statistical/distributional properties of real-world data. Assuming some real distribution $\mathbb{P}_r$ and some generated distribution $\mathbb{P}_g$, it would be desirable if $d(\mathbb{P}_r, \mathbb{P}_g) \to 0$ for some probability distance measure d. Such a distance d may be the Jensen-Shannon divergence or the Wasserstein distance.

Synthetic data nowadays is created leveraging statistical algorithms but also more sophisticated machine learning models. The areas of potential applications of synthetic limit order book and financial time series data more generally are plenty:

1. Studying scenarios and different dynamics of LOBs: Instead of relying on just one path, i.e. realization of the LOB process, one can simulate a wide range of paths that have the same distributional properties. E.g. where the LOB is likely to evolve after a certain event or from a particular state. This helps to understand its dynamics and permits to derive potentially useful quantities of interest in applications.

2. Robustness of trading strategies: Trading strategies are often designed using a limited amount of data and history. Hence, there is a substantial risk that a strategy is being fit to a path rather than to the entire data-generating process the path is from. Realistic synthetic data allows to test a given strategy not only on one particular path observed in history but on thousands of generated

paths. This helps, for instance, to obtain more robust parameters of a trading strategy.

3. Environment for reinforcement learning applications: As mentioned, in particular brokers and market makers want to act optimally in LOBs. To do that, reinforcement learning (RL) is a potential approach. However, in order to apply RL for LOBs an environment must be simulated. This is indispensable in the setting of market making and execution as every action an agent takes has an impact on the system and changes the probability of future events/transitions. Hence, historical data may not be used for the application of RL in LOBs. Furthermore, it is important to keep in mind that potential flaws and biases in the simulated environment may lead to undesirable behavior of the RL agent.

4. Backtest environment for execution algorithms: Synthetic LOBs can not only be useful in RL settings but also to generally test execution algorithms. Execution strategies are dealing to a substantial degree with market impact. A backtest environment of synthetic LOBs that allows efficient approximation of a given strategy's market impact. This can be of great use both in academia and industry.

5. Data privacy and security: Synthetic data is used to protect sensitive data from being exposed or breached. Instead of using real data, synthetic data can be used which prevents sensitive information from being leaked. Synthetic data does not contain any personally identifiable information or other sensitive information which helps for data sharing and collaboration purposes without compromising the privacy or security of the original data.

6. Training machine learning models: Synthetic data is often used to train machine learning models when real-world data is scarce, sensitive, or unavailable. By generating synthetic data that has similar statistical properties to real-world data, machine learning models can be trained more effectively and accurately.

Overall, synthetic data is a powerful tool and can be used in a variety of applications by both industry as well as academia. Hence, it is becoming increasingly important in a data-driven world. In particular, the issue of data privacy and sharing issues is an important roadblock for collaborations, for which synthetic data offers a simulation and enables data sharing.

Recently, simulators – also called market generators – have been developed using more data-driven techniques in order to have synthetic limit order book data. In particular, generative models such as Generative Adversarial Networks (GANs) or Variational Autoencoder (VAEs) have been leveraged to model and sample from complex (conditional) probability distributions.

This thesis considers two approaches to simulate LOBs directly and to improve existing models via a hierarchical model on top of it.

1.4.1 Limit order book simulation with generative adversarial networks

The paper "Limit Order Book Simulation with Generative Adversarial Networks" (Cont, Cucuringu, Kochems, & Prenzel, 2023) is available as a preprint on SSRN.

The paper considers the case of modeling a reduced representation of the LOB. In particular, we are dealing with the public view as shown in Chapter 2. In particular, we design a GAN-based approach to learn the probability distribution of future order book snapshots, creating entire time series of such by recurrent simulation. This problem is natural as many market participants rather act on LOB snapshots at specific time intervals rather than on the entire event stream. Most studies using GANs have been focusing on modeling event streams which is a very hard problem and furthermore is a problem with many discrete variables for which GANs are not very suitable. While with this representation some granularity is lost, it shrinks down the problem significantly as instead of modeling the entire/incremental change, we model the cumulative effect on the queue sizes for some Δt.

The particular interests of the projects were the following:

1. What dynamics of the limit order book can we learn in this reduced order book representation? How realistic are the dynamics learned?

2. What happens when starting to interact with the order book?

The chosen state representation, in comparison to event stream modeling which is a discrete process in continuous time, makes it suitable to use generative adversarial networks to draw samples from a conditional probability distribution

$$\mathbb{P}(X_{t+\Delta t}|S_t)$$

where $X_{t+\Delta t}$ is the future order book snapshot with queue sizes for some price array and S_t is some condition. In the most simple Markovian case, this may be $S_t = X_t$

i.e. the current order book state. The simulator then maps some noise seed together with the state condition into the space of future limit order book states trying to reproduce the real distribution as closely as possible.

Training the GAN sufficiently well and regularizing the training process allows to accurately fit the process of LOB snapshot transitions. Marginal distributions and correlations of both levels and changes of the order book state are fit closely. Recurrent simulation allows for the generation of entire paths. Findings show that the quantity process remains stable over several hundreds of steps and the extracted price paths properly match the data. Also, conditional probabilities of queue sizes and price changes show realistic patterns. The conditional probability price changes based on the best queues are reproduced with high precision. This shows, that properly learning this reduced form can lead to proper macro-level results of the simulations.

Following, we perform different experiments in order to investigate the behavior of the market generator when interacting with the limit order book state via liquidity extraction and liquidity provision.

1. Systematic liquidity extraction shows exhibits a price impact of the right correction, i.e. buy executions drive prices up while sell executions drive them down respectively. The impact increases with increasing order size and furthermore shows a decaying marginal impact. The result is a slightly concave market impact curve, in line with models such as (Gatheral, 2010). Approximating the expected market impact with a rule of thumb — exact comparisons are difficult since the market impact is not directly observable, especially with public data — shows the model learns 70-80% of the market impact.

2. Systematic liquidity provision is done via increasing the queue size at certain time steps. As expected, the liquidity provision supports the corresponding price level, driving prices up for buy limit orders and down for sell limit orders. Two major observations are noteworthy. First, the impact of liquidity provision, i.e. passive orders, is less than liquidity extraction, i.e., aggressive orders. This is in line with observations in the literature. Second, the model does not only show an impact of liquidity provision at the best queues but also at deeper levels. The impact of liquidity provision at deeper levels shows even less magnitude than the impact of liquidity provision at best queues which is again in line with expectations.

3. Execution via liquidity extractions in different frequencies leading to the same percentage of volume (POV) show the same impact. This shows that the driving factor for market impact is trade participation.

In summary, a simulator is built for modeling and simulating a reduced state of the order book. Doing this properly allows to extract many realistic both unconditional and conditional patterns already in a Markovian setting. Future research allows to build up on that in order to improve on this model's weaknesses.

1.4.2 Dynamic calibration of order flow models with generative adversarial networks

The paper "Dynamic Calibration of Order Flow Models with Generative Adversarial Networks" (Prenzel et al., 2022) has been published as a conference paper at the Third ACM International Conference on AI in Finance in 2022 and furthermore won one of the three best paper awards.

Primarily, the paper deals with non-stationarity in limit order books. Plenty of order flow models exist which recover some of the phenomena observed in financial data such as for instance the order book shape or self and cross-exciting effects of limit orders. Furthermore, some of these models have proven theoretical properties regarding their stability or certain quantities such as the probability of future price change and more. However, most of them tend to be stationary. I.e. they are calibrated using several days (or even weeks) of limit order book data. This assumes that parameters are averaged over many different market scenarios. However, it is widely known that limit order books exhibit strong non-stationarities in the order flow, amongst the following:

1. Trading volume as well as the order intensities tend to have a classical U-shaped intraday pattern. In the morning and in the evening, order intensities tend to be higher while trading generally takes a slight dip during noon.

2. Temporal trends and imbalances in order arrival rates. For certain reasons, the may be significantly more buy market than sell market orders, etc.

3. Dependency on volatility, generally more (less) trading activity during days of high (low) volatility.

To this end, we suggest building a hierarchical model on top of an existing order flow model to induce new, non-stationary dynamics. We loosen the assumption of

stationarity and assume order arrival rates are most likely not stationary over several days or months but potentially over a few minutes (depending on the stock).

Using a short estimation time frame of just a few minutes we create a dataset whose distribution is learned by a GAN. The trained model can be used in order to generate unseen but realistic calibrations on conditions. The results show that GANs can easily learn the conditional distribution of the calibrations which is shown as an example for the intraday patterns as well as the volatility pattern. Furthermore, the generated calibrations contain imbalances which is generally not the case for global calibrations in which parameters are sometimes even averaged in order to obtain symmetric dynamics.

Using the synthetic calibrations in order to simulate the event streams with changing parameters. In particular, the order flow becomes non-stationary and exhibits more macro patterns such as temporal trends, U-shaped volatility, as well as trading activity. Two advantages of this approach are that any conditions may be added without much handcrafting. The model automatically learns the dependency (in comparison to a handcrafted baseline model).

Chapter 2

Limit Order Books

This chapter formally introduces a new way to view limit order books accounting for different agents submitting orders. This allows to derive different views on the LOB. Afterwards, common approaches to model the order flow in limit order books are introduced which are particularly important in the remainder of the thesis.

2.1 The limit order book as a heterogeneous queueing system

Most limit order book representations do not contain any information about the origin of the order. For our purposes, however, it is necessary to include the origin and derive a new notation. In essence, we add the origin of an order and represent the LOB in measure spaces which allows to derive more detailed views of the limit order book. To start, we define two sets of agents.

Definition 2.1.1 (Agent Set)**.** The set of different agents is denoted by A. Every element of this set A denotes a trading account.

We also consider that brokers exist and some agents act through a broker via the broker's execution services.

We consider a subset of agents $B \subset A$ who submit orders through a given broker.

Definition 2.1.2 (Limit Order)**.** A limit order is an order to buy or sell a certain quantity of an asset at a pre-specified price. A limit order (LO) (t, p, q, α) is characterized by

- an arrival time $t \in \mathbb{R}^+$,
- a price $p \in \delta\mathbb{N}$ which is a multiple of the price tick $\delta > 0$ (the minimal increment of prices),

- a quantity $q \in \mathbb{Z} \setminus \{0\}$ with $q < 0$ denoting buy orders and $q > 0$ denoting sell orders,
- the identity $\alpha \in A$ of the agent submitting the order.

A limit order is considered "outstanding" as long as it has neither been canceled nor fully executed.

A *market order* is an order for immediate execution. We represent a market order as an *executable limit order* with price $p = 0$ (for a market sell order) or $p = \infty$ (for a market buy order). It is thus immediately executed if there is enough liquidity in the book, i.e. enough orders of the opposite sign against which the order can be executed. We assume that there is always enough liquidity. Hence, a market order is never outstanding and never forms part of the LOB.

If an order is canceled, it is removed from the LOB. Alternatively, when a partial cancellation occurs, the order size q is decreased by the amount of the cancellation.

Denote by ϵ_x the unit point mass at x by and by

$$\mathcal{M}_+(\mathbb{R}_+ \times \delta\mathbb{N} \times A),$$

the space of (Radon) measures on $\mathbb{R}_+ \times \delta\mathbb{N} \times A$. One can represent the collection of buy (resp. sell) orders of agent $\alpha \in A$ as a pair of measures

$$\mu_+, \mu_- \in \mathcal{M}_+(\mathbb{R}_+ \times \delta\mathbb{N} \times A)$$

where $\mu_+([0,t] \times \{p\} \times \{\alpha\})$ represents the volume of buy orders submitted by agent α at price p, between time 0 and t. The measure $\mu_-([0,t] \times \{p\} \times \{\alpha\})$ represents the corresponding volume of sell orders respectively.

The order book may then be represented as a signed measure

$$\mu = \mu^+ - \mu^-. \tag{2.1}$$

This representation allows to derive different views on the limit order book with different degrees of granularity. In particular, we will introduce the anonymized, broker, and omniscient view.

2.1.1 Anonymized view

The *anonymized* or *aggregate view* of the limit order book corresponds to the information available to market participants who observe the volume of orders at each price. It does not contain any information regarding the origin or the submission times of the orders. Hence, most market participants only observe the *queue size* Q_p at each price level p.

Definition 2.1.3 (Queue Size)**.** The queue size for any price $p \in \delta\mathbb{N}$ is denoted by

$$Q_p = \sum_{\alpha \in A} \mu([0,t], \{p\}, \alpha). \tag{2.2}$$

We call $Q : \delta\mathbb{N} \to \mathbb{Z}$ the anonymized limit order book, $Q \in \mathcal{M}(\delta\mathbb{N})$ and belongs to the state space $\mathbb{Z}^{\delta\mathbb{N}}$. Note, for the anonymized view, t must correspond to the current time. In other words, market participants with access to the anonymized view at time t are not able to access the net volume of those outstanding orders that have been submitted in $[0, t']$, with $t' < t$. This would reveal information about the queue position, which generic market participants generally do not have.

The *anonymized view* is visualized in Figure 2.1a and corresponds to the sum of all orders' quantity at a particular level. The agent neither observes the number of orders nor the color (i.e. the agent id).

2.1.2 Omniscient view

The *omniscient view* of the limit order book is the collection of all outstanding limit orders, including the information about their time of submission and the identity of the submitter. This information is represented by the measure μ. E.g. in Figure 2.1c, the *omniscient view* not only contains the single orders with prices and quantities but also the color (i.e. the agent id). In other words, the omniscient view allows to measure the volume of orders that have been submitted between $[t, t']$ at price p from agent α. Also the queue position can be tracked by subtracting all orders that have been submitted before an order's submission time t. Outstanding orders from agent $\alpha \in A$ are then represented by

$$\mu(\cdot, \cdot, \{\alpha\}) \in \mathcal{M}_+(\mathbb{R}_+ \times \delta\mathbb{N}). \tag{2.3}$$

While the *omniscient view* exists usually no one has access to it. This is due to the circumstance that even exchanges do not necessarily have full information on the

origin of an order. Nonetheless, the closest to the *omniscient view* is the view that the corresponding exchange has on the order book because it has some origin about all the flow in the LOB. However, in many cases, the exchange is not able to distinguish between the flow of different clients from a broker. Thus, the exchange sees all orders of a particular broker as one aggregated flow.

2.1.3 Broker view

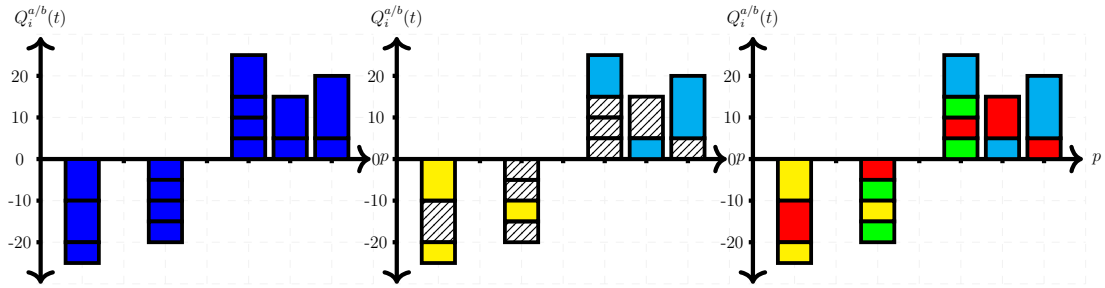
The *omniscient view* and the *anonymized view* of the limit order book represent two extreme cases of information. However, there are intermediate situations corresponding to partial information on the limit order book. An important case is the case of a broker who can observe the order submission times and identities for a set B, where $B \subset A$, of agents. This broker, however, will have a less detailed view of the limit order book for all other agents $\alpha \notin B$. This corresponds to aggregating over all agents not in B. Denote by $B^* = B \cup \{\Delta\}$ the set obtained by adding one element to B; this element will represent all other agents not included in the broker's set of clients. The *broker view* of the limit order book may then be described by a measure $\mu_B \in \mathcal{M}(\mathbb{R}_+ \times \delta\mathbb{N} \times B^*)$ defined by

$$\mu_B(\cdot,\cdot,\{\alpha\}) = \mu(\cdot,\cdot,\{\alpha\}) \qquad \alpha \in B, \tag{2.4}$$

$$\mu_B(\cdot,\cdot,\{\Delta\}) = \sum_{\alpha \notin B} \mu(\cdot,\cdot,\{\alpha\}). \tag{2.5}$$

An exemplary *broker view* snapshot is visualized in Figure 2.1b. Yellow and light blue correspond to agents $\alpha \in B$, thus are from the broker's clients. These orders are observed with their particular ID and viewed via $\mu_B(.,.,\{\alpha\})$, while green and red orders would correspond to "all other agents" (Δ) which are not in B, the broker's set of clients. These orders are seen as $\mu_B(.,.,\{\Delta\})$. Furthermore, the broker does not observe the specific time as (2.5) suggests, but rather the relative time since the broker knows its queue position. Hence, the broker can (only) know that an order has been submitted between two orders of its own clients.

Most literature aims to model the *anonymized view* of the LOB. These models have been shown to be useful for describing certain dynamics of limit order books. They include general price moves or distribution of queue sizes. The detailed queue and a specific order's queue position, however, are generally not available in both public data and the majority of the models in the literature. This has led to research aiming to estimate the queue position of an order (Moallemi & Yuan, 2016). As shown by (Moallemi & Yuan, 2016), "queueing effects can be very significant"



(a) Public view (b) Broker view (c) Omniscient view

Figure 2.1: Different views of the same LOB snapshot. Note, only the *omniscient view* has complete information about the queue priority of each order.

and accounting for the queue position should not be ignored when designing trading algorithms. In accordance with this, keeping track of the queue size rather than the entire collection of orders does not allow cancellations to refer to particular orders. This makes keeping track of how orders flow through the queue impossible. Instead, a LOB model that accounts for this heterogeneity allows to do this. We remark that the topic of identifying determinants of limit order cancellations has been extensively studied recently, and would be an interesting research direction to explore, in itself.

2.1.4 Centered limit order book

The *anonymized view* presented also corresponds to the queue size, Q_p, indicating the cumulative quantity to buy or sell a certain stock at price p. For many applications, for instance, the arrival of orders, the relative price level is more important than the absolute price level (Bouchaud et al., 2009). For this, one generally refers to reference prices, i.e. the bid/ask and mid price. In particular, the *bid* price is the highest price at which some agent α is willing to buy a certain asset.

Definition 2.1.4 (Bid price)**.**

$$p_b = \sup_p \{p > 0, Q_p < 0\} \tag{2.6}$$

where $Q_p < 0$ corresponds to $\mu([0,t], p, \cdot) < 0$ for some $\alpha \in A$.

The *ask* price is the lowest price at which some agent α is willing to sell a certain asset.

Definition 2.1.5 (Ask price)**.** The *ask* price is the lowest price at which some agent α is willing to sell a certain quantity of a stock. Hence,

$$p_a = \inf_p \{p > 0, Q_p > 0\} \tag{2.7}$$

where $Q_p > 0$ now corresponds to $\mu([0,t], p, \cdot) > 0$ for some $\alpha \in A$.

Note, for both p_b and p_a the interval $[0,t]$ of the measure μ must be from 0 to t with t corresponding to the current time as for the best price any outstanding order submitted at any time in the past is valid.

The mid price is then just the average between the best bid and ask price and is the price generally depicted for daily time series:

$$p_{mid} = \frac{p_a + p_b}{2} \tag{2.8}$$

Not every level in a limit order book is always populated, i.e. there is no willingness to buy or sell the stock for a certain price. This happens in particular between the bid and the ask price, for instance, if there is more uncertainty about where the price will move.

Definition 2.1.6 (Spread)**.** With p_a being the best ask price and p_b being the best bid price, the spread is defined as the distance between the lowest price of a sell order and the highest price of an ask order

$$s = p_a - p_b, s \in \{k \cdot \delta | k \in \mathbb{N}\}. \tag{2.9}$$

Clearly, $s \geq \delta$ in order to ensure no arbitrage. δ denotes the tick size from 2.1.2.

One common observation is wider spreads in the morning when the information from the night is "processed" in the markets due to the insecurity of where prices are moving.

With the two reference prices p_a and p_b one can define the relative queue size.

Definition 2.1.7 (Centered limit order book)**.** Let Q_p be the *anonymized* view as defined in Definition 2.1.3. Then, for some combination of best prices p_b and p_a, the relative queue size is given by

$$Q_i^b = Q_{p_a - i \cdot \delta}$$

for the bid side and

$$Q_i^a = Q_{p_b + i \cdot \delta}$$

for the ask side. In words, it denotes the quantity available in the book i ticks away from the best opposite price.

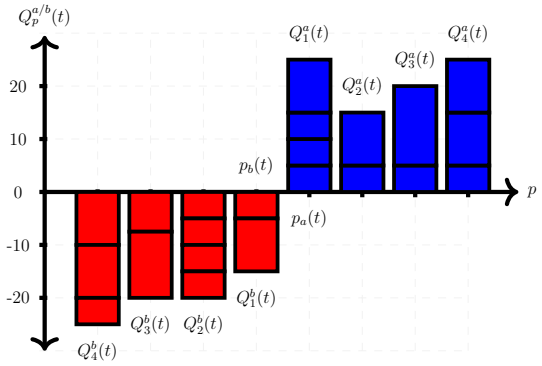


Figure 2.2: Arbitrary LOB snapshot indicating the queues resulting from several orders. Buy queues (red) are negative, sell queues (blue) are positive.

Figure 2.2 shows an exemplary order book indicating the relative quantities and the best ask prices. It is important to note that in the case of a spread $s > \delta$ the quantities overlap. E.g. assume $s = 2\delta$ (i.e. two ticks), then $p_a - \delta = p_b + \delta$. Hence Q_1^a and Q_1^b refer to the same price level and thus $Q_1^a = Q_1^b = 0$.

A full description of the market, the *omniscient view*, is not always available. The *broker view*, however, offers a partially more detailed view of the market, in particular of the broker's set of agents B. Thus, we want to shed some light on what B consists of to better understand the limit order market ecosystem. In case, the ecosystem can be separated into several groups of order flows, modeling the order flow of each group is a much more feasible and tractable task in comparison to modeling every single agent $\alpha \in B$. Chapter 3 analyzes the order flow of a broker and separates a typical set of agents using execution services into different representative groups. These show homogeneous behavior within their group but differ substantially across different groups.

2.1.5 Events in the order book

The LOB, i.e. the measure μ or the queue size Q_p, evolves over time due to order events, which affect these. In particular in Chapter 4 and Chapter 5, we will primarily look at how events affect the *anonymized view* of the limit order book, e.g. the queue size. We will hence consider the quantities at a certain point in time $t \in \mathbb{R}^+$. The order types which are widely modeled in the literature are limit order, market order, and cancellation order, altogether driving the main dynamics in a LOB.

1. **Limit order submission:** An agent submits a limit order x at time t and price $0 < p < p_a(t)$ for bid orders and $p_b(t) < p < \infty$ for ask orders respectively.

From time t on, order x is part of the limit order book. The change of the queue size is given by $\Delta Q_p(t) = q$.

2. **Limit order cancellation:** An agent cancels their existing order x. Since the cancellation sets the order size to $q = 0$, x is not part of the LOB anymore. The change of the queue size at order price p reads $\Delta Q_p(t) = -q$.

3. **Execution of a limit order via market order:** An agent submits a buy market order of quantity q at time t. This quantity is then immediately executed against sell limit orders on the ask side. There may be hidden limit orders, which are not considered here as we are interested in modeling the lit market. In most exchanges, this is done via price-time priority. E.g. a market order of $q = 1$ would be executed against the oldest limit order (in the front of the queue) at the best available price. If the market depth is smaller than the order size (e.g. $Q_{p_a}(t) < q$), the queue is depleted, and the remaining quantity $q - Q_{p_a}(t)$ is executed against limit orders from the succeeding price levels (a phenomenon often referred to as market sweep orders).

2.1.6 Parent orders

As Figure 1.1 visualizes, limit orders sent through a broker part of a larger parent order. Since these parent orders are central in the sequel of the paper, we formally introduce the structure of a parent order.

Definition 2.1.8. A parent order P is defined by a number of variables some of which are determined by the submitting trader, others are only known at the end of the execution.

- a submission or arrival time $t \in \mathbb{R}^+$,
- a target quantity $q^{target} \in \mathbb{Z} \setminus \{0\}$, the total quantity which shall be executed in the market,
- an executed quantity $q^{exec} \in \mathbb{Z}$ which corresponds to the total sum of executed child orders.

The parent order's target quantity q^{target} can be decomposed in its absolute quantity

$$\tilde{q} = |q^{target}| \in \mathbb{N}$$

and sign (buy or sell order)

$$\text{sign}(q^{target}) = \begin{cases} -1, & \text{if } q^{target} < 0 \\ 1, & \text{if } q^{target} > 0. \end{cases} \tag{2.10}$$

Each order has an order placement schedule, which we denote by

$$\mathcal{X} = \{x_1, \ldots, x_N\}, \quad x_i = (t_i, p_i, q_i, \alpha_i) \; \forall \; x_i \in \mathcal{X}, \tag{2.11}$$

where each element has the form of a LO as defined in definition 2.1.2. The schedule contains all LOs placed within the parent order execution. Hence, $\alpha_i = \alpha$ since all limit orders are posted by the same agent. Note, $\mathcal{X}$ may contain effective market orders as executable limit orders. As mentioned above, a buy limit order ($q_i < 0$ with price $p_i = \infty$) would correspond to a buy market order ($q_i > 0$ and $p_i = 0$ to a sell market order respectively).

Additionally, there is an execution schedule

$$\mathcal{X}^{exec} = \{x_1, \ldots, x_N\}, \quad x_i = (t_i, p_i, q_i, \alpha_i) \; \forall \; x_i \in \mathcal{X}^{exec}, \tag{2.12}$$

which is comprised of all market orders affecting the parent order P. This can either be a market order x with $p \in \{0, \infty\}$ sent within the order schedule in eq. (2.11); in this case, $x \in \mathcal{X}$. Or, a market order x is sent by any other agent which (partially) executes an outstanding order from $\mathcal{X}$; in this case, $x \in \mathcal{X}^{exec}$ but $x \notin \mathcal{X}$.

Note, any element $x \in \mathcal{X}^{exec}$ from another agent α' may not be the complete market order. Assume an incoming market order executes several limit orders. In this case, only the fraction affecting a limit order in $\mathcal{X}$ is kept in $\mathcal{X}^{exec}$. If two limit orders of $\mathcal{X}$ are executed by the same market order, then there are two entries with the same time stamp. Alternatively, several market orders affect a limit order from $\mathcal{X}$ currently outstanding. In this case, there are also several elements in $\mathcal{X}^{exec}$, one for each partial execution of an order in $\mathcal{X}$. Note, that both the limit order schedule as well as the execution schedule are only fixed after execution.

2.2 Models of limit order books

This section introduces several approaches to model the LOB. Some of the models are used as benchmark or base models. Namely, Poisson order flow (Section 2.2.1), Hawkes order flow (Section 2.2.2), agent-based models (Section 2.2.3) as well as an SPDE model for LOBs (Section 2.2.4)will be briefly reviewed.

2.2.1 Poisson order flow

Throughout the literature, point processes and in particular Poisson processes have been leveraged to mathematically model the order flow in limit order books. (E. Smith et al., 2003; Abergel & Jedidi, 2013; Cont et al., 2010) amongst others use independent Poisson processes to model three event types: limit order arrivals, cancellations, and market orders. As argued in (Bouchaud et al., 2009), the arrival of the orders depends less on the absolute price level but rather on the distance of the best opposite price. Thus, the object modeled is the relative queue size as in Definition 2.1.7 indicating the cumulative ask (a) or bid (b) quantity i ticks away from the best opposite price at time t.

For each order type, different order arrival rates are assumed – some depending on the distance to the best opposite price:

1. Limit orders arrive with intensity $\lambda_i^{L,\star}$ at side $\star \in \{a/b\}$ and $i \in \{1, \dots k\}$ ticks away from the best opposite price.

2. Cancellation orders arrive with intensity $\lambda_i^{C,\star}(t)$ at side $\star \in \{a/b\}$ and $i \in \{1, \dots k\}$ ticks away from the best opposite price. This intensity is proportional to the current queue size, i.e. $\lambda_i^{C,\star}(t) = \tilde{\lambda}_i^{C,\star} \cdot Q_i^\star(t)$. In essence, the cancellation rate increases with increasing queue size. This is according to the assumption that each unit of an order is equally likely to be canceled with intensity $\tilde{\lambda}_i^{C,\star}$. Furthermore, scaling the cancellation intensity $\lambda_i^{C,\star}(t)$ by the queue size $Q_i^\star(t)$ avoids a potential explosion of the queue as cancellations become more likely with increasing queue size.

3. Market orders arrive with intensity $\lambda^{M,\star}$ at side $\star \in \{a/b\}$

For all intensities, $\lambda \in \mathbb{R}_0^+$ must hold. An exemplary visualization of the model and where the different $\lambda_i^{L,\star}$ "sit" is given in figure 2.3. Note, $\lambda_1^{L,a}$ and $\lambda_1^{L,b}$ are located at the same price level in Figure 2.3. This is to ensure that both buy and sell limit orders can be placed in the empty level.

The whole set of order arrival intensities (parameter set) of the model is given by

$$\Lambda = \{\lambda^{M,a}, \lambda^{M,b}\} \cup \{\lambda_i^{L,\star}, \lambda_i^{C,\star} | \star \in \{a,b\}, i \in \{1, \dots, k\}\} \in \mathbb{R}_0^{+4k+2} \tag{2.13}$$

$$= (\lambda_1, \dots, \lambda_i, \dots, \lambda_{4k+2}) \in \mathbb{R}_0^{4k+2}. \tag{2.14}$$

The second part of (2.13) is a reindexing of the first part for convenience, especially in Chapter 5. Since the event types are assumed independent, the Poisson order flow

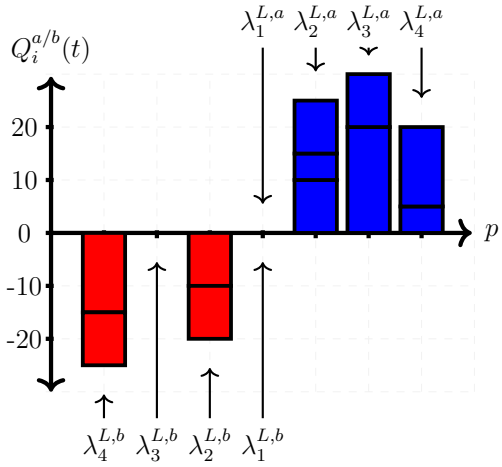


Figure 2.3: Exemplary order book state with indications where arrival rates $\lambda_i^{L,\star}$ currently "sit".

consists of $4k+2$ independent counting processes $\{N_i(t), t \geq 0\}$ for each order type with event times $t_{i,1}, t_{i,2}, \ldots$ for the i-th process (i.e. order type). The number of events are $x_i \sim Po(\lambda_i)$ distributed. Hence, the interarrival times are exponentially distributed with rate λ_i, i.e. $Exp(\lambda_i)$.

For different independent Poisson processes, it holds that $\sum_i x_i \sim Po(\sum_i \lambda_i)$. Hence, the counting process of all orders is

$$Po\left(\sum_{\lambda \in \Lambda} \lambda\right) \tag{2.15}$$

distributed and interarrival times between two consecutive orders follow an exponential distribution with rate $\frac{1}{\sum_{\lambda \in \Lambda} \lambda}$. The single events then depend on the relative size of arrival rates. For example, the probability of the next event drawn of the process $Po\left(\sum_{\lambda \in \Lambda} \lambda\right)$ to be a buy market order is equal to $\frac{\lambda^{M,a}}{\sum_{\lambda \in \Lambda} \lambda}$.

Regarding the order sizes, (Cont et al., 2010) assume a unit order size. This allows to derive quantities such as the probability of the next price move being positive, the probability of an order getting executed before the mid-price moves, and the probability that one buy and one sell limit order are executed before the mid-price moves.

Most models either assume constant order size or some continuous distribution. In fact, there tend to be many orders around even numbers and also large outlier orders. Figure 2.4 displays the distribution of order sizes with the empirical CDF in Figure 2.4a and the histogram for orders of size up to 2000 in Figure 2.4b. Clear spikes for order sizes around even numbers can be recognized.

Algorithm 1 Poisson order flow

Require:

Model parameters: k (levels), Λ (parameters), (Q^a_∞, Q^b_∞) (size of hidden queues), volume distributions

Simulation parameters: N (number of events), $Q^\star_i(0), \forall i \in \{1, \dots k\}, \forall \star \in \{a, b\}$ (initial LOB state)

Ensure:

Initialization: $t \leftarrow 0$

for $i = 1, \dots, N$ **do**

$\lambda^{C,\star}_i(t) \leftarrow \tilde{\lambda}^{C,\star}_i(t) \cdot Q^\star_i(t)$ (Update cancellation intensities)

Draw $\Delta t \sim \text{Exp}(\sum_{\lambda \in \Lambda} \lambda)$

Draw type with probability $\frac{\lambda}{\sum_{\lambda \in \Lambda} \lambda}$, store relative price level i, event type $\{L, M, C\}$ and side $\{a, b\}$

Draw volume q_x from $\mathcal{V}_L, \mathcal{V}_C, \mathcal{V}_M$

$t \leftarrow t + \Delta t$

Update order book

Enforce boundary, i.e. $Q^\star_{k+1}(t) = Q^\star_\infty, \forall \star \in \{a, b\}$

end for

The simulation algorithm is given in Algorithm 1. Extensions include the queue-reactive model (Huang et al., 2015) which introduces dependency of queue size on the order arrival rates.

An estimator for limit and market order submission rates is given by

$$\hat{\lambda}^{L,\star}_i = \frac{N^{L,\star}_{i,T}}{T} \text{ and } \hat{\lambda}^{M,\star} = \frac{N^{,\star}_T}{T}. \tag{2.16}$$

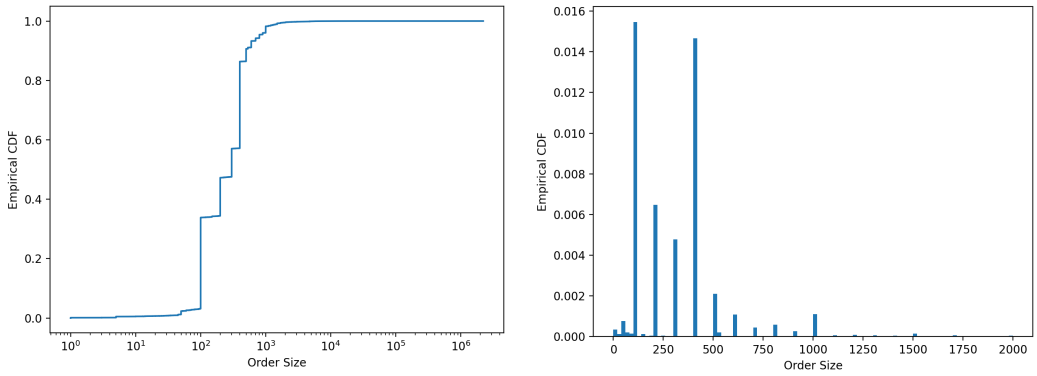
$N^{L,\star}_{i,T}$ and $N^{M,\star}_T$ denote the limit (market) number of orders in the given time period of length T. The intensity of cancellation orders is proportional to the queue size as each order is assumed to have equal cancellation probability. The estimate is thus scaled by the average queue size

$$\hat{\lambda}^{C,\star}_i = \frac{N^{L,\star}_{i,T}}{T \cdot \overline{Q^\star}_i}. \tag{2.17}$$

2.2.2 Hawkes order flow

Empirical data suggests the arrival of orders is not independent but rather shows excitation behaviors of two types.

1. Self excitation: The arrival of an order of a particular type increases the instantaneous intensity for this order type. For instance, assume an ask (a) limit order arrival at time t at relative price level i. Then, $\lambda^{L,a}(t_+) > \lambda^{L,a}(t_-)$



(a) CDF (b) Histogram

Figure 2.4: Distribution of order sizes for one day of limit orders from MU stock. This entails roughly 400,000 events.

2. Cross excitation: The arrival of an order of a particular type increases the instantaneous intensity for *other* order types. E.g., $\lambda^{L,a}(t_+) > \lambda^{L,a}(t_-)$ due to some order other than L, a, i.

Hawkes processes – introduced in (Hawkes, 1971) – have been leveraged in order to account for these excitation effects.

In the Hawkes order flow model, the arrival rates are now given by a Hawkes process and are thus stochastic:

$$\lambda^n(t) = \lambda_0^n(t) + \sum_{m=1}^{p} \int_0^t \phi_{nm}(t-s) dN^m(s), n = 1, \ldots, p \tag{2.18}$$

where $\phi_{nm}(t-s)$ is a kernel modeling the effect from a particular event of type m occurred at time s on event type n at time t. For simulations in this thesis, we use the exponential kernel

$$k_{\alpha,\beta}(t-s) = \alpha_{nm} \exp(-\beta_{nm}(t-s)) \tag{2.19}$$

which is a common choice together with the power law kernel. The exponential kernel makes the system Markovian and hence faster to simulate. The result of the calibration in the case of the exponential kernel is a vector and two matrices.

1. Base intensities

$$\boldsymbol{\lambda_0} = (\lambda_0^1, \ldots, \lambda_0^p)^T \in \mathbb{R}^p$$

describes for each of the Hawkes process dimensions the value to which the arrival rate decays if no further event occurs.

2. Excitation factors

$$\alpha = \begin{pmatrix} \alpha_{1,1} & \cdots & \alpha_{1,p} \\ \vdots & \ddots & \vdots \\ \alpha_{p,1} & \cdots & \alpha_{p,p} \end{pmatrix} \in \mathbb{R}^{p \times p}$$

where the m-th element in the n-th row indicates how much an event of type m excites events of type n.

3. Decay factors

$$\beta = \begin{pmatrix} \beta_{1,1} & \cdots & \beta_{1,p} \\ \vdots & \ddots & \vdots \\ \beta_{p,1} & \cdots & \beta_{p,p} \end{pmatrix} \in \mathbb{R}^{p \times p}$$

where the m-th element in the n-th row indicates how the excitation of event of type m on events of type n decays.

As can be seen, the number of parameters is given by $2p^2 + p$. Hence, with growing dimensionality of the Hawkes processes, the number of parameters grows quadratically. This makes high-dimensional Hawkes processes very hard and slow to calibrate. For $p = 2$ one may simulate the Hawkes process via a thinning algorithm (Lewis & Shedler, 1979; Ogata, 1981, 1998).

Algorithm 2 Hawkes process flow

Require:

Model parameters: Base intensity $\lambda_0^n(t)$ and kernel parameters $(\alpha_{nm}, \beta_{nm})$ for $m, n = 1, \ldots, P$ of the P-variate Hawkes process

Initialization: $t \leftarrow 0$, $I(0) \leftarrow \sum_{\lambda \in \Lambda^C(t)} + \sum_{n=1}^{P} \lambda_n(t + \Delta)$

Time of first event: Draw $\Delta t \sim \text{Exp}(\sum_{i=0}^{4k+2} \lambda_i)$

while $t < T$ **do**

Draw $D \sim U(0, 1)$

$I(t + \Delta t) \leftarrow \sum_{\lambda \in \Lambda^C(t)} + \sum_{n=1}^{P} \lambda_n(t + \Delta)$

if $D < \frac{I(t+\Delta t)}{I(t)}$ **then**

Draw type with probability $\frac{\lambda}{\sum_{\lambda \in \Lambda} \lambda}$, store relative price level i, event type $\{L, M, C\}$ and side $\{a, b\}$

Update order book

Enforce boundary, i.e. $Q^{\star}_{k+1}(t) = Q^{\star}_{\infty}$

Draw $\Delta t \sim I(t)$

end if

$t \leftarrow t + \Delta t$, $I(t) \leftarrow I(t + \Delta t)$

end while

The estimation of Hawkes processes generally is much more involved than the estimation of an independent Poisson process. (Á. Cartea, Cohen, & Labyad, 2021)

introduce a new method for gradient-based estimation of multivariate Hawkes processes via least squares. In this work, the estimation of Hawkes parameters is done with the corresponding package available.

2.2.3 Agent-based models

Agent-based models – so-called ABMs – are computational models that allow the simulation of complex systems. Generally, ABMs consist of several different agents which (inter-)act together.

In finance, different types of agents may be different market participants trading in different fashions which are then used to simulate a process such as a LOB. The agents simulate the behavior of various market participants, such as individual traders, institutional investors, market makers, and High-frequency traders. Each agent in the model is endowed with specific characteristics, strategies, and decision-making rules based on real-world observations or empirical data. These agents interact with each other and respond to market conditions, including the LOB, in order to replicate the dynamics of real financial markets.

By incorporating agent heterogeneity and the interplay between agents, ABMs can – in theory – capture emergent phenomena, such as price fluctuations, liquidity dynamics, herding behavior, and market crashes, that arise from the collective actions of market participants. ABMs potentially allow for the exploration of different scenarios and the examination of the impact of specific agent behaviors or policy interventions as well as changes of these on market outcomes.

In the context of LOBs, ABMs can provide insights into how the order flow, order placement strategies, and the actions of market participants influence the liquidity and price dynamics of the market. ABMs can capture the complex interactions between market participants, such as the impact of different trading strategies on the LOB structure, the formation of bid-ask spreads, the presence of market manipulation, and the overall market efficiency.

(Byrd et al., 2019) build a simulation framework for agent-based simulation of limit order book. It allows for both replay and simulation of LOB data. They assume an exchange agent, which acts as the stock exchange and matches orders with each other. Then, generally noise traders, fundamental or technical traders are assumed. (Paddrik et al., 2012) build an ABM based on the agents extracted in (Kirilenko et al., 2017) in order to simulate the flash crash – also studied in (Kirilenko et al., 2017). These agents are *small*, *fundamental buyers*, *fundamental sellers*, *market makers*,

opportunistic and *high-frequency traders* and differ in number, trade speed, position limits, and market volumes.

While ABMs may appear to be a natural approach to simulate an environment in which many different market participants interact with each other they come with several difficulties. The main problem is the calibration of agent types. First, it is not really clear which agent types exist in financial markets ((Kirilenko et al., 2017), (Cont, Cucuringu, Glukhov, & Prenzel, 2023) and (Á. Cartea et al., 2023) shed some light on this). It is also not clear, how to calibrate these agent types once they are assumed and how the calibrations interact with each other in different market environments. While the interpretation of single agent types is simple and an appealing property of ABMs, the global dynamics of the interactions are often complex and not clear to understand. The result is poorly calibrated models which do not lead to the desired results. This problem goes in line with the robustness of ABMs and possibly large changes in the model outcome when slightly changing parameters.

2.2.4 SPDE model for limit order book dynamics

Usually, a very large number of participants is acting in a LOB. This makes the dynamics very complex and difficult to model. (Cont & Müller, 2021) view the limit order book as a density and model the evolution of this normalized density, denoted as

$$u_t(x) = v(t, p_{mid}(t) + x),$$

via a stochastic partial differential equation (SPDE). (Cont & Müller, 2021) consider several effects impacting the LOB density $u_t(x)$.

1. a rate of buy (sell) order submissions is modeled by a function $f^b(x)$ for bid and $f^a(x)$ for ask orders.

2. complete cancellations are modeled by $\alpha_b u_t(x)$ ($\alpha_a u_t(x)$). This term represents a constant fraction of $u_t(x)$ which gets canceled.

3. a "convection term" (Cont & Müller, 2021) is given by $-\beta_b \nabla u_t(x)$ ($-\beta_a \nabla u_t(x)$) and causes replacement of orders closer to the mid side.

4. a diffusion term is modeled by $\eta_b \Delta u_t(x)$ ($\eta_a \Delta u_t(x)$ where respectively) which causes a flattening of the order book shape via replacing order mass around the particular price level. Δ denotes the Laplacian operator indicating the curvature of the LOB density at x.

5. High-frequency traders are quickly placing and canceling limit orders with a low latency. A stochastic noise term, i.e. $\sigma_b u_t(x) \mathrm{d}W^\star$. (W^a, W^b) models their impact on $u_t(x)$.

The entire SPDE reads

$$\mathrm{d}u_t(x) = [\eta_a \Delta u_t(x) + \beta_a \nabla u_t(x) + \alpha_a u_t(x) + f^a(x)]\mathrm{d}t + \sigma_a u_t(x)\mathrm{d}W_t^a, \forall x \in (0, L) \tag{2.20}$$

for the ask side and

$$\mathrm{d}u_t(x) = [\eta_b \Delta u_t(x) - \beta_b \nabla u_t(x) + \alpha_b u_t(x) - f^b(x)]\mathrm{d}t + \sigma_b u_t(x)\mathrm{d}W_t^b, \forall x \in (-L, 0) \tag{2.21}$$

for the bid side where

$$\eta, \beta, \sigma \in (0, \infty), \alpha < 0, f : (-L, 0) \cup (0, L) \to [0, \infty). \tag{2.22}$$

The constraints ensure that the above-described effects are in the correct direction. A negative $\beta_\star$, for instance, would lead to a shift of the orders away from the mid price which would decrease liquidity and is not very realistic. (Cont & Müller, 2021) show the problem can be framed into a general family of SPDEs for which they provide two tractable examples.

Chapter 3

Analysis and Modeling of Client Order Flow in Limit Order Markets

3.1 Introduction

Price formation in major financial exchanges takes place through the interaction of buy and sell orders sent by a variety of market participants. Orders are submitted to the exchange either through execution services or directly by the agents themselves. These orders are routed through a central Limit order book (LOB) which continuously records the state of outstanding limit orders on the exchange.

The dynamics of the limit order book, which ultimately drives price dynamics, is determined by the flow of buy and sell orders. Market participants may employ a wide variety of trading strategies and intervene at different frequencies, ranging from millisecond to daily. High-frequency traders (HFTs) and market makers (MMs) tend to have direct access to exchanges, while other market participants may route orders through a broker. This results in an aggregate order flow which is the superposition of multiple, heterogeneous components with different frequencies and characteristics, as depicted in Figure 1.1.

In first instance, one might separate traders into two groups – those trading with a proprietary access to exchanges, and those trading through execution services/brokers. The first group primarily contains high-frequency traders and market makers. The remainder are traders which trade through execution services and are not in full control of the actual placement of limit orders. These traders, depicted on the left side, send *parent orders* (also called *meta orders*) to a particular desk of a broker. The broker then uses scheduling algorithms, such as VWAP, TWAP,

POV to slice the parent order into a stream of child orders according to the trader's preferences. The resulting child orders are then sent as *limit* or *market* orders to different venues via a smart order router (SOR). Some of these venues may be lit venues, such as XETRA or LSE exchange, others could be, for example, dark pools or other systematic internalizers. Studies using public LOB data only see what is being sent to exchanges at the end of the order process (right hand-side of Figure 1.1).

This heterogeneity, which is arguably an important feature for risk managers and market regulators (Kirilenko et al., 2017), is challenging to model. Indeed, most stochastic models proposed for limit order book dynamics (E. Smith et al., 2003; Cont et al., 2010) represent the order flow as a homogeneous point process, often for reasons of analytical tractability. These models are generally based on aggregate order flow data such as the LOBSTER database[1], which does not contain information on parent orders or agents submitting them.

Our contributions are twofold. First, we introduce a new granular representation of the limit order book which accounts for the origin of orders and distinguishes different levels of information such as the *anonymized view*, which has been the focus of previous studies, from the *broker view* which is the focus of our analysis.

Second, using detailed order flow data from a major broker, we investigate the structure of the incoming order flow (e.g. the left side of Figure 1.1. We use trade execution data to segment traders into representative groups with similar attributes. These are then analyzed for stability in both the cross-sectional (i.e., cross-asset) and temporal dimensions. To the best of our knowledge, our study is the first one to provide an analysis of a broker's parent order flow.

We find that equity order flow can be segmented into four different components: Quant, Day VWAP, Signal, and Res(idual) order flows. The different groups are stable both over time as well as across stocks. This also holds for the traders themselves. Heterogeneity in both the agents' structure and the order flow they generate in LOBs allows the development of heterogeneous order flow models.

Based on these results, we present a modeling framework for client order flow from the viewpoint of a broker. Without going down to the level of granularity of an agent-based model, our framework decomposes the order flow into components representing different agent types while remaining easy to calibrate and simulate.

[1] https://www.lobsterdata.com/

Outline Section 3.2 presents the data on client order flow and describes its segmentation into different components which we identify as representative agent types. Section 3.3 describes the properties of the order flow for each agent type. A simple model for parent order flow capturing the main heterogeneity between the different agent types is presented in Section 3.4. Section 3.5 summarizes the results and presents future research directions.

3.1.1 Related work

As introduced in Section 1.2, there is a lot of research conducted in the field of limit order markets. Many studies investigate statistical properties or build stochastic models of limit order books, using public data; some references may be found in (Cont, 2011; Gould et al., 2013; Bouchaud et al., 2002).

There are fewer studies on order flow characteristics of particular types of market participants, due to the confidential nature of such data. (Brogaard et al., 2010, 2014) study LOB data in which orders from HFTs and MMs are flagged, in order to analyze the impact of HFT and MM accounts on market quality and price discovery. The studies find evidence that HFTs increase market quality by potentially dampening intraday volatility among other properties. Furthermore, they argue that the trade directions of HFTs are based on public information such as "macro news announcements, market-wide price movements, and limit order book imbalances" (Brogaard et al., 2014). (Hagströmer & Nordén, 2013) analyze LOB data with information about orders from HFTs and MMs, specifically outlining the differences between these two types of market participants in the way they tend to trade. They find MMs to take the majority of limit order traffic and to hold lower inventories compared to HFTs. Along these lines, (Van Kervel & Menkveld, 2019) investigate the behavior of HFTs when large institutional orders are being executed. The study claims HFTs initially trade against institutional orders and after some time change their direction for the "most informed institutional orders". (A. Cartea, Payne, Penalva, & Tapia, 2019), however, find evidence of lower market quality due to ultra-fast trading. This lower quality is manifested by "greater quoted and effective spreads and lower depth". (Hendershott, Jones, & Menkveld, 2011) analyze the effect of HFTs without proprietary data using the "rate of electronic message traffic". The study finds more narrow spreads and less adverse selection. (Hasbrouck & Saar, 2013) attempt to extract HFT trades by identifying the so-called "strategy runs", i.e., periods with similar inter-arrival times and order sizes, which is unique for HFTs, as the authors argue. Their suggested measure of low latency indicates that with increasing low latency trading, spreads

decrease, and the depth at the first level increases. A review of statistical modeling of high-frequency data using public data is given in (Dutta, Karpman, Basu, & Ravishanker, 2023).

The information content of broker order flow has been studied in (Barbon et al., 2019; Di Maggio et al., 2019) and (Hendershott et al., 2020). (Di Maggio, Kermani, & Song, 2017) study brokers' trading behavior claiming brokers tend to set tighter spreads when trading with their closest ties. (Hendershott et al., 2020) analyze transaction costs in over-the-counter corporate bonds markets, depending on the network size of the insurers. (Di Maggio et al., 2019) analyze non-public data to study the network of links between institutional investors and brokers. The study claims brokers obtain information when executing "informed trades" and some of this information is then passed to some of their clients. This is supposed to lead to higher returns for such clients according to (Di Maggio et al., 2019). (Barbon et al., 2019) use similar data focusing on large liquidations of funds, finding that clients of brokers who are aware of such large liquidations tend to execute substantially more during these periods.

Researchers and practitioners often use agent-based approaches when modeling financial markets. Insights into market microstructure and the behavior of different agents are especially interesting for ABMs as they allow for improved calibration and knowledge of different agents. In particular, (Farmer & Foley, 2009) argue about the importance of agent-based models. (Farmer, Patelli, & Zovko, 2005) study and apply a simple "zero-intelligence" agent-based model. (Byrd et al., 2019) build a simulator framework for agent-based modeling of limit order books. (Wang, Hoang, Vorobeychik, & Wellman, 2021) build an agent-based model to study the effect of spoofing on the market.

Clustering, a type of unsupervised machine learning, is suitable for discovering unknown structures in data as it is the target for our study (Hastie, Tibshirani, & Friedman, 2009). Besides clustering agents, clustering algorithms have also been applied in other financial contexts. (Bennett, Cucuringu, & Reinert, 2022) cluster lead-lag networks finding statistically significant clusters in the US market. (Cirulli, Kobak, & Ulrych, 2022) use unsupervised learning for portfolio construction combining hierarchical clustering with momentum methods. (Horvath, Issa, & Muguruza, 2021) cluster market regimes using a novel clustering algorithm based on the Wasserstein distance. The algorithm is extended in (Issa & Horvath, 2023) where a "path-wise method for regime clustering" is introduced. This allows applications to more complex settings, such as non-Markovian ones.

A related study is (Kirilenko et al., 2017)'s analysis on transactions in S&P 500 E-Mini futures during the Flash Crash of May 2010 using audit trail data under the use of an algorithm presented in (Mankad et al., 2013). (Kirilenko et al., 2017) classify the agents into high-frequency traders, market makers, fundamental buyers, fundamental sellers, and opportunistic traders, based on transaction volume and scaled net positions. The behavior of the agent types before and during the flash crash is then compared. Focusing on high-frequency traders and market makers, they find both behave differently and the former especially did not significantly alter their trading behavior albeit the large selling pressure in the market. Despite having access to a very granular data set, the (Kirilenko et al., 2017) focus on high-frequency traders and market makers during the flash crash, and less on the properties of the remaining market participants. (Wright, Reimherr, & Liechty, 2022) apply machine learning in order to classify traders based on their limit order placement behavior under the use of the Wasserstein distance. (Á. Cartea et al., 2023) most recently investigate the structure of the "Euronext Amsterdam" exchange in order to predict trading strategies in electronic markets. Model coefficients are then used in order to cluster trading strategies, namely "directional trading", "opportunistic trading" and "market making". These classes of trading strategies are similar to the classified agents found in (Kirilenko et al., 2017).

Our analysis is broader than (Kirilenko et al., 2017) in the sense that it is focused on multiple tickers across a longer period, using a more detailed data set containing information on parent orders and order flow as well as transactions. In particular, we are able to include order flow from traders who do not have direct market access to the exchanges.

3.2 Segmentation of agent types

This section considers the task of segmenting the population of traders that send orders to the brokers. Traders sending orders build the very left-hand side of Figure 1.1. Our analysis of this process stands in contrast to known works in the literature, since most studies only use anonymized order flow data, as outlined in Section 3.1.1. Few studies use non-public LOB data, and mainly (Kirilenko et al., 2017) has access to the trading accounts. Consequently, the available data structure not only allows us to observe what arrives in the LOB and who sends it, but also whether several limit orders come from the same parent order. Uncovering latent similarities across traders and identifying different trader typologies can potentially facilitate better modeling

of the parent order flow in LOBs. The main implication of our findings is that one can move beyond modeling each agent or trader independently, and pave the way for modeling each homogeneous group or cluster of agents.

3.2.1 Data set and features

For the analysis, we use anonymized trade execution data from a major broker. Market shares can be found in reports from industry providers of strategic benchmarking such as Coalition. The data used in this paper forms a market share mostly in the high single-digit percentage – depending on the stock. This makes it amongst the largest private data sets of broker execution data in European markets. This gives reason to assume that the results are unlikely to be completely different at other large brokers. The motivation for this stems from the fact that traders often do not only trade solely via one broker but also via several major brokers. This data set builds a detailed view of a subset of the entire market, as explained in the previous sections. In particular, it allows one to observe what traders submit to the broker as a parent order, up to where and when different child orders corresponding to this parent order are placed. This holds for both lit and dark venues. The universe is comprised of stocks from *STOXX 600*, and buckets of 1-month duration will be employed in the segmentation.

For each asset i and time period t, the set of traders that execute their trades via the broker is denoted as $B_{i,t}$ where each agent $\alpha \in B_{i,t}$ has sent at least one order to the broker which led to an execution. Clearly, $B_{i,t} \subseteq A_{i,t}$, the whole set of agents active in a LOB of the given ticker. Instruments are primarily analyzed separately; joint analysis is used primarily for matters of comparison and stability, in particular in section 3.2.4.

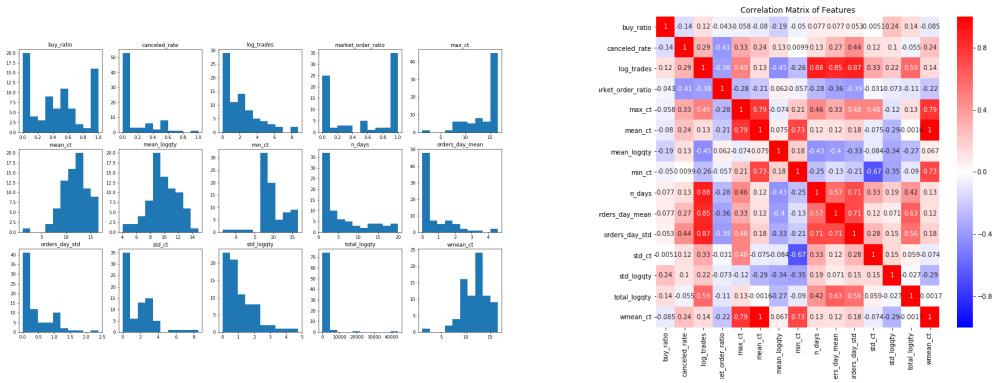
Generally speaking, the broker manages the execution and the child orders which are sent to the LOB. For the analysis, we thus rely on the parent order structure and the corresponding statistics for each agent. Features regarding the single child orders are not included. This means: when do agents send orders, how large are they, and how aggressively shall they be executed? The corresponding data fields are primarily *side* (buy/sell), *number of orders, time of submission*, and *size*. The resulting features/statistics of this information include, for example, the average direction of an agent's trades, the number of orders an agent has submitted, distributional properties of the day time an agent submits orders, or an agent's order sizes' standard deviation. Information regarding external information or market conditions such as momentum,

volatility, etc. are also used. A detailed description of each feature can be found in Table A.1.

Altogether, this amounts to a data set of $n = |B_{i,t}|$ agents with p features

$$X \in \mathbb{R}^{n \times p} \text{ with } x_\alpha = (x_{\alpha,1}, \ldots, x_{\alpha,p}) \in \mathbb{R}^p \; \forall \; \alpha \in B_{i,t}$$

describing $B_{i,t}$. Each vector x_α describes the parent order structure of agent α in instrument i during period t. The data sets, and in particular the various features considered, are of different dimensions and also ranges. Furthermore, some features are heavily skewed, having many smaller values and few very large values. This holds true in particular for features like order size or number of orders. Thus, we use the logarithm of such features and then standardize the data before proceeding with any further analysis. To this end, we apply z-standardization using the sample mean and standard deviation. Alternatively, methods like min-max normalization may be used. In the present case, no major differences could be noted using different normalization techniques.



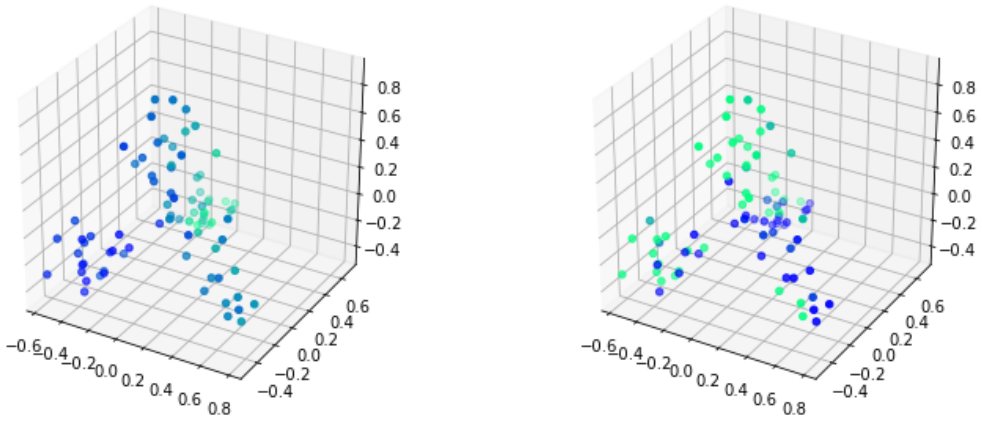
(a) Marginal distribution

(b) Pearson correlation matrix

Figure 3.1: Features of data.

Figure 3.1 shows the distribution and correlation of computed features for one exemplary $B_{i,t}$ from the *STOXX 600*. Many features, such as the ratio to which a particular agent specifies a maximum price for execution (minimum price for sell orders respectively) – *market order ratio* – tend to be rather 0 or 1. I.e., most agents either always or never indicate a limit for executions of child orders. Other features are very skewed, for example *n_days*, the number of days an agent sends an order. This is because most agents only send very few orders during one month for a given ticker.

To visualize agents in a lower-dimensional space and get a first idea of the data structure, methods such as principal component analysis (Hastie et al., 2009) or spectral embedding (Belkin & Niyogi, 2002) may be applied to obtain further insights into the structure of the agents interacting with financial brokers. To this end, Figure 3.2 shows two exemplary features in the embedding space. The mean creation time of a client's orders in Figure 3.2a and the ratio indicating the fraction of orders of a particular trader which contains a maximum price for the execution (minimum price for sell orders respectively) in Figure 3.2b. For both features in Figure 3.2 there are areas of the embedding where traders with similar values of the corresponding features are clustered. For example, there exists an area of traders that have a much higher mean creation time compared to that of other clusters. In Figure 3.2b, a group of traders can be encountered which tends to specify a limit price when sending orders for executions. Additionally, the fact that the point cloud in the embedding is not just a sphere indicates some structure in the underlying data which can be further exploited.



(a) Mean creation time

(b) Ratio of specified limit price

Figure 3.2: Exemplary embedding using first three eigenvectors of the normalized Laplacian matrix using the spectral embedding method from (Belkin & Niyogi, 2002). The color of the nodes indicates the values of the associated feature. The brighter the points, the higher the value of the corresponding feature.

3.2.2 Spectral clustering

As outlined in section 2.1, a broker has a detailed view of a subset of the market, i.e. knows the trader identity and the specific time of an order, for all agents $\alpha \in B_{i,t}$. For ease of notation, we will refer to $B_{i,t}$ as B in the sequel. To better understand the structure of a typical set of traders which use a broker, this section segments B into a disjoint partition $\mathcal{C} = \{C_1, \dots, C_K\}$, in order to obtain representative agent types that best describe the structure of a broker's clients.

Clustering algorithms are designed to do exactly what is this section's purpose. They separate the observations into different, typically unknown, classes or clusters. These types/clusters should be heterogeneous, but traders of the same type should rather form a homogeneous population with similar characteristics. For the present case, each client of the set B obtains an index $i \in I \subseteq \{1, \dots, |B|\}$. I indicates the indices of those agents that have done at least one trade in one month. A partition $\mathcal{C} = \{C_1, \dots, C_K\}$ is sought after, such that

1. $C_k \subseteq B \; \forall \; k \in \{1, \dots, K\}$, every cluster is a subset of the set of agents using the broker B
2. $C_k \neq \emptyset \; \forall \; k \in \{1, \dots, K\}$, every cluster should at least contain one agent
3. $C_k \cap C'_k = \emptyset \; \forall \; k \neq k'$, the clusters should be disjoint.

Furthermore is optimal with respect to some objective. In other words, a disjoint partition $\mathcal{C} = \{C_1, \dots, C_K\}$ is sought after which is optimal with respect to some objective. In our case, this means every trader would belong to a certain type.

Referring to Figure 1.1, we would like to shrink down the number of nodes representing many individual traders submitting parent orders on the left-hand side to just a few nodes representing each trader type. K indicates the number of types the agents shall be separated into. $\bar{x}_1, \dots, \bar{x}_K \in \mathbb{R}^p$ are the corresponding cluster centers indicating the average features (i.e. coordinates) of the data points (i.e. agents) affiliated with the corresponding cluster, so

$$\bar{x}_{k,j} = \frac{1}{|C_k|} \sum_{\alpha \in C_k} x_{\alpha,j} \; \forall \; j \in \{1, \dots, p\}.$$

Essential to each clustering algorithm is a distance matrix $W \in \mathbb{R}^{n \times n}$ which contains the distance $W_{\alpha,\alpha'}$ between observation x_α and $x_{\alpha'}$, for some distance measure d. This distance is often the Euclidean distance.

The spectral clustering technique used in this study combines two methods: spectral embedding and the K-means clustering algorithm (Shi & Malik, 2000; Ng, Jordan, & Weiss, 2002; Meila & Shi, 2001). In other words, it creates the partition via an iterative ascent algorithm on a p'-dimensional, non-linear embedding of the data set, describing the agents' trading behavior.

1. **Construction of the adjacency graph**

   The adjacency graph indicates for each pair of observations whether these are connected or not. In particular,

   $$A \in \mathbb{R}^{n\times n} \text{ where } A_{i,j} = \begin{cases} 1 \text{ if } \|x_i - x_j\|^2 < \epsilon \\ 0 \text{ else,} \end{cases} \qquad \forall\, i,j \in \{1,\dots,n\}. \tag{3.1}$$

   Two vertices are connected by an edge if their Euclidean distance in the actual space $\mathbb{R}^p$ is below a certain threshold ϵ. Alternatively, one may connect a vertex i to its l nearest neighbors where $l \leq n, l \in \mathbb{N}$.

2. **Weighting the similarities**

   This step weighs the connections between observation i (row) and j (column). The common choice is the heat/rbf kernel. In this case, the matrix $W \in \mathbb{R}^{n\times n}$ is computed with

   $$W_{i,j} = \begin{cases} e^{-\frac{\|x_i - x_j\|^2}{\sigma}} \text{ if } A_{i,j} = 1 \\ 0 \text{ else} \end{cases} \qquad \forall\, i,j \in \{1,\dots,n\}, \tag{3.2}$$

   for some user-tuned bandwidth parameter $\sigma \in \mathbb{R}_+$. The closer (i.e. more similar) the observations x_i and x_j are in Euclidean distance, the higher the value W_{ij}. Alternatively, whenever the input matrix is binary, with $A_{i,j} = 1$ for connected vertices, we set $W_{i,j} = 1$ if $A_{i,j} = 1$.

3. **Compute eigenmaps via first p' eigenvectors**

   Next, the generalized eigenvector problem

   $$Lf = \lambda Df, \tag{3.3}$$

   is solved. $D \in \mathbb{R}^{n\times n}$ is a diagonal matrix containing the row-sums of W, i.e. $D_{i,i} = \sum_j W_{i,j}$. We define $L = D - W$ to be the unnormalized Laplacian matrix (also referred to as the Combinatorial Laplacian).

   The solution of equation (3.3) is a set of (sorted) eigenvalues $\lambda = (\lambda_0, \dots, \lambda_{n-1})$ for which $\lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_{n-1}$ holds. For the corresponding eigenvectors

$f = (f_0, \ldots, f_{n-1})$ from equation (3.3) $\lambda_0 = 0$ and $f_0 = (1, \ldots, 1) \in \mathbb{R}^n$ holds (Von Luxburg, 2007). The multiplicity of the $\lambda_0 = 0$ indicates the number of connected components in the graph. Finally, the eigenvectors $f_1, \ldots f_{p'}$ are then used for the p'-dimensional embedding and $y_i = (f_1(i), \ldots, f_{p'}(i)) \in \mathbb{R}^{p'}$.

4. **Clustering the low-dimensional embedding**

   Finally, the agents are clustered in this low-dimensional embedding via the K-means algorithm (MacQueen et al., 1967). The objective function optimized by the algorithm is given by

$$\min_{\mathcal{C}, \{\bar{y}_k\}_k} \sum_{k=1}^{K} |C_k| \sum_{\alpha \in C_k} \|y_\alpha - \bar{y}_k\|^2, \tag{3.4}$$

   where $y_\alpha, \bar{y}_k \in \mathbb{R}^{p'}$, the space of the embedding. The objective function (3.4) corresponds to the minimization of the variance within the clusters with respect to partition $\mathcal{C}$. Optimization is done via iterative descent, alternating between recomputing cluster centers $\bar{y}_k$ and reassigning the observations y_α to the nearest cluster center[2].

In contrast to K-Means, spectral clustering is a non-linear clustering method. This enables the algorithm to potentially uncover clusters which are not convex since the clustering is not performed in the original feature space $\mathbb{R}^p$, but rather in the embedding space $\mathbb{R}^{p'}$ (Von Luxburg, 2007). This capability of uncovering non-linear relationships makes spectral clustering more flexible in comparison to K-Means.

Extracting centroids of a partition when using spectral clustering is not as straightforward as for K-Means. This is because the embedding procedure described above is not invertible for any arbitrary point. Hence, a point $y \in \mathbb{R}^k$ cannot be mapped into the actual feature space $\mathbb{R}^p$. To overcome this, one can use the center of a cluster's observations in the actual space from the data $X \in \mathbb{R}^{n \times p}$, i.e. $\bar{x}_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i \ \forall \ k \in \{1, \ldots, K\}$. Alternatively, the prototype for cluster k can be defined as $x_i \in \mathbb{R}^p$ with $i = \arg\min_{i \in C_k} \|y_i - \bar{y}_k\|^2$. In other words, the observation x_i is the one whose embedding y_i is the closest to the cluster center $\bar{y}_k$ in the embedding space. In this study, the first method is used to obtain cluster centers, as it is less prone to outliers for some feature of the prototype x_i. The cluster centers are then used to give details about properties and provide a comparison of the different agent types.

[2] See (Hastie et al., 2009) for the detailed algorithm.

Several algorithms were used to cluster $B_{i,t}$ and find an optimal partition $\mathcal{C}$. In the following, we particularly outline observations and statistics when comparing the partitions between K-means and spectral clustering in more detail for one exemplary $B_{i,t}$. The number of clusters K ranges from 2 to 5. The number of agents for the ticker is ~ 80.

Cluster sizes: Generally, the number of observations in the clusters is neither very large nor very small. There is no cluster with only very few observations.

Consistency: The partitions using K-means and spectral clustering are very similar. This consistency can quantified by the adjusted rand index (ARI) which indicates the consistency across two partitions. Let $\mathcal{C}^{(1)} = \{C_1^{(1)}, \dots C_K^{(1)}\}$ and $\mathcal{C}^{(2)} = \{C_1^{(2)}, \dots C_K^{(2)}\}$ be two partitions. The ARI reads

$$ARI = \frac{\sum_{k,k'} \binom{n_{k,k'}}{2} - \left[\sum_k \binom{a_k}{2} \sum_{k'} \binom{b_{k'}}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_k \binom{a_k}{2} + \sum_{k'} \binom{b_{k'}}{2}\right] - \left[\sum_k \binom{a_k}{2} \sum_{k'} \binom{b_{k'}}{2}\right] / \binom{n}{2}}, \tag{3.5}$$

where $n_{k,k'} = |C_k^{(1)} \cap C_{k'}^{(2)}|$ denotes the number of observations inside $C_k^{(1)}$ and $C_{k'}^{(2)}$ and $a_k = \sum_j n_{k,k'} = |C_k^{(1)}|$ ($b_{k'} = \sum_k n_{k,k'} = |C_{k'}^{(2)}|$). The index takes a maximum value of 1 if $\mathcal{C}^{(1)} = \mathcal{C}^{(2)}$.

Table 3.1 shows the ARI for different values of K. A particularly high consistency is indicated for 2 and 4 clusters. Noteworthy is the high consistency taking into account that K-means is performed on the actual feature space ($\mathbb{R}^p$), while the spectral clustering is based on the non-linear embedding in $\mathbb{R}^{p'}, p' \ll p$. In other words, regardless of the feature space employed, the recovered partitions are very similar.

# Clusters	2	3	4	5
ARI	**0.7979**	0.5107	**0.7968**	0.6153

Table 3.1: Table indicating the ARI between K-means and spectral clustering. The higher the number, the higher the consistency between to partitions; the maximum value ARI can attain is 1, indicating a perfect matching of the clusters.

Stability of clusters: Increasing K leads to a sequential splitting of the data cloud into clusters. For example, one cluster gets further split when increasing K by one unit. The other clusters and their affiliated traders remain stable.

Number of clusters: Setting $K = 2$ or $K = 4$ leads to the best scores. First of all, the consistency is better than for 3 and 5 clusters indicated by higher ARIs in Table 3.1. Lastly, Table 3.2 shows the variance between agents and their corresponding cluster center. In particular, the more the variance decreases, the more justified the

addition of another cluster. The second differences of the variance within the clusters can indicate how the variance reduction of an additional new cluster changes. This helps to identify a K, for which an increase leads to a much lower variance reduction. In particular, values for K would be either 2 or 4, which goes in line with the other metrics.

# Clusters	1	2	3	4	5	6	7	8	9
Variances	3.914	3.507	3.210	2.981	2.871	2.738	2.640	2.596	2.536
2nd Diffs.	-	**0.108**	0.069	**0.118**	-0.021	0.034	0.053	-0.016	-

Table 3.2: The variance between an observation and its corresponding cluster. The stronger the decrease when K is increased by one cluster, the larger is the marginal effect of the new cluster to separate the data.

3.2.3 Representative agent types

Despite the final goal being the heterogeneity of the aggregated order flow caused by different clusters, the cluster centers $\bar{x}_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i$ are computed to interpret the agent types. Such resulting cluster centers for a subset of the features used are shown in Table 3.3, for one exemplary $B_{i,t}$.

	1: C-Quant	2: C-Day VWAP	3: C-Signal	4: C-Res
Buy ratio	0.63	**0.55**	0.63	**0.65**
Cancellation ratio	**0.26**	**0**	0.15	0.07
# Trades per month	**219.7**	**1.72**	4.08	5.03
Maximum order creation time	**16:10:47**	**08:32:04**	15:02:03	14:34:05
Mean order creation time	12:03:50	**08:25:34**	**14:20:00**	11:31:56
Mean order size (in ADV)	**0.01**	0.03	**0.06**	**0.06**
Mean momentum (bps)	2.285	**-17.87**	29.94	**38.26**
Mean volatility	22.2	23.03	22.42	22
Minimum order creation time	**07:58:08**	08:19:10	**13:34:31**	08:40:12
# Active days per month	**13.47**	**2.1**	3.71	4.95
St. dev. of order creation time	02:24:49	**00:07:12**	00:42:16	**02:40:46**

Table 3.3: Exemplary centroids setting $K = 4$ for one of the 25 most liquid STOXX 600 instruments during December 2019. Liquidity in this case refers to the total number of trades in the data base. The highest (lowest) value of the corresponding features are highlighted in blue (red).

In fact, as shown in Table 3.4, the four agent types may be summarised as follows:

1. **Quantitative agents** (Quant): The most distinguishable agent type is cluster 1, which contains those agents that submit many trades within the period, with

Cluster Name	Cluster Description
C-Quant	Mainly quantitative traders, many trades, small volumes, orders sent throughout the day, execution with few child orders leading to a large POV, net inventory closer to 0
C-Day VWAP	Mostly VWAP as execution, few orders, only sent in the morning, almost no cancellation
C-Signal	Typically trading in the afternoon (especially US market opening), large trades, large amounts traded in dark venues, large POV in general
C-Res	Large orders, medium frequency, sent throughout the day

Table 3.4: Summary of representative agents.

a smaller trade size. Despite trading rather small amounts, the total volume traded is by far the highest in this cluster. Also, the number of days during which the cluster's agents trade is 12 days, which is several times higher than the second-highest value. Cluster 1 exhibits the highest average cancellation rate, indicating that this agent type perhaps tracks the execution of its trades more actively than other types. Also, the time of the trades is uniformly distributed across the whole day, with a mean creation time around noon. This type of agent may be summarized as a "*Quantitative agent*", called Quant in the following sections.

2. **Day VWAP agents** (Day VWAP): The most distinct cluster from Quant is cluster 2 as its features exhibit the largest anti-correlation to features of Quant. The number of trades per agent is the lowest, and all trades are submitted very early in the morning. The average trade size is larger than for Quant, yet not the largest across the agents for this partition. Most noticeable apart from the few number of trades with larger sizes is the creation time of the order, which typically only occurs before market opening. This indicates that these orders are large orders typically sent before market opening, and with an execution target over the entire day. Looking at the execution algorithm, one can primarily find rather passive algorithms such as VWAP, which. This further supports our previous interpretation of the Day VWAP cluster. This trading behavior is further reflected in the cancellation rate, which is the lowest across all clusters while having the most child orders during the execution. This type of agent is summarized as a "*Day VWAP agent*", called Day VWAP in the following

sections.

3. **Signal agents** (SIGNAL): Cluster number 3 is most distinguishable due to its creation time, which shows a minimum of $\sim$ 13:30 and a maximum of $\sim$ 15:00 for the corresponding data slice. Hence, this is an agent type which is typically active in the afternoon (which corresponds to the opening of the US market since it is a STOXX 600 instrument). Similar to DAY VWAP, the cancellation rate is quite small, only $\sim$ 5%, when compared to QUANT which is around $\sim$ 14%. This agent type tends to execute large order sizes (5% of the average daily volume of the last 20 days). Additionally, the cluster in this example has a high percentage of volume, and significant fractions are executed via dark venues which leads us to assume the agent type is less concerned about execution costs, but has a high urgency – hence also executes a lot in dark venues to mitigate traces in the market. The reason may be that this agent type wants to act on trading signals and is thus referred to as a "*Signal agent*", called SIGNAL in the following sections.

4. **Residual agents** (RES): Cluster 4 indicates agents with the largest average trade size and only about five trades per month. The trades are submitted during the entire day, which is also indicated by the high standard deviation of the creation time. Furthermore, the standard deviation of the sizes, as measured in the logarithm of the average daily volume percentage, is the highest. This is the "least distinguishable" agent type and seems to be in between the other three clusters. One possible reason may be that some agents trade different strategies, and thus do not act very homogeneously. Hence, these agents may be referred to as "*Residual agent*", denoted as RES in the following sections.

3.2.4 Stability of clusters

Section 3.2.3 shows that the set of agents $B_{i,t}$, trading asset i during period t through a broker, can be segmented into different clusters. Setting $K = 4$ gives an appropriate number of agent types which can be interpreted, and are also very distinct in their representative features. Moreover, the partitions do not change significantly when changing the clustering algorithm or the feature standardization procedure.

It is, however, unknown how the cluster affiliations and the representative agent types change over time or different instruments. Are the results in section 3.2.3 random or is the partition rather stable? This section addresses the stability of the agents and clusters, both across different instruments as well as different time slices.

One problem arising when comparing two partitions in the present case is that most often $B_{i,t} \neq B_{j,t'}$. In other words, traders which are active in asset i during period t do not necessarily trade asset j during period t'. Nonetheless, the following two questions are of particular interest. First, how stable is the affiliation of an agent to a particular cluster? Do agents change their behavior and act differently across time or different tickers? Second, how stable are the agent types and their features presented in section 3.2.3?

Two approaches are pursued to answer these questions:

1. **Joint clustering of several data slices**:

   As before, $B_{i,t}$ is the set of agents α with at least one trade of asset i during period t. For example, $\bar{x}_{i,\alpha} = \frac{1}{|T_{i,\alpha}|} \sum_{t \in T_{i,\alpha}} x_{i,t,\alpha}$ (where $T_{i,\alpha} = \{t | \alpha \in B_{i,t}\}$) denotes the average of a trader's features across all time periods in which they have traded at least once. If T periods of one asset i, i.e. $B_{i,t}\ \forall\ i \in \{1, \ldots, T\}$ are clustered together, a high concentration of observations from one agent, e.g. $x_{i,t,\alpha}$ for all $t \in T_{i,\alpha}$ in one cluster C_k indicates consistency of the agent across different time spans, and similar for different assets in the same period. Denoting $P_\alpha(\alpha \in C_k)$ as the empirical probability for an observation of agent α to be in cluster C_k, the entropy

   $$H_\alpha = - \sum_{k \in \{1,\ldots,K\}} P_\alpha(\alpha \in C_k) \log(P_\alpha(\alpha \in C_k)) \tag{3.6}$$

   is used to measure a client's concentration in one cluster. Additionally, the maximum affiliation probability, defined as

   $$\max_k P_\alpha(\alpha \in C_k), \tag{3.7}$$

   indicates a more interpretable degree of concentration.

Dimension	Entropy	Maximum Affiliation
Ticker	0.4	0.8
Time	0.5	0.76

Table 3.5: Log-weighted average entropy and maximum affiliation probability of the agents. The first row indicates both values for the consistency across different tickers. The second row indicates the values across different time frames.

Table 3.5 shows log-weighted values for entropy and maximum affiliation in both ticker and time dimension. Log weights with respect to the number of tickers (respectively, time periods) a trader has been active in are used to give higher weights to such traders that span across several distinct sets. The log-weighted entropy is ~ 0.4, which is relatively low compared to the maximum value of entropy for 4 clusters of 1.4. The log weighted maximum affiliation probability is ~ 0.8. In other words, for the cluster C_k which contains most of some agent's observations, the average probability of that agent's observation to be in this cluster is 80%. This indicates some stability of the agent's behavior as well as affiliation to a cluster, across different tickers. For the temporal dimensions, the numbers are similar providing evidence that an agent's trading behavior does not completely change over time either.

2. **Meta clustering**:

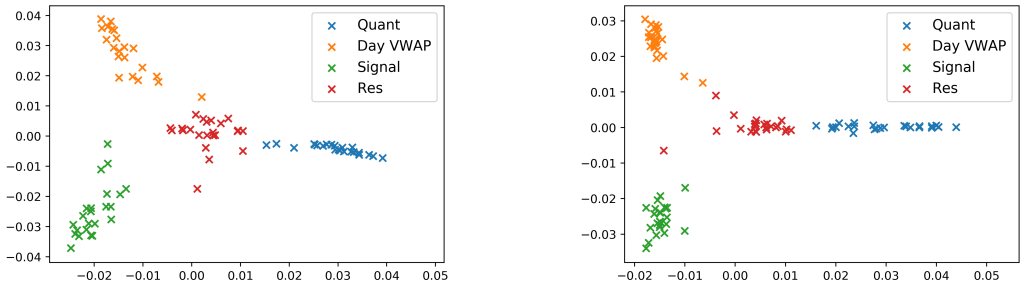
   For stability of the representative agent types, their representative features should be similar over different tickers (or time spans). To investigate this, meta clustering can be applied. In the first step, every $B_{i,t}$ is clustered as before. The result are N different cluster partitions for each instrument (or T for each time period):

   $$\mathcal{C}^{i,t} \text{ with } \mathcal{C}^{i,t} = \{C_0^{i,t}, \ldots, C_K^{i,t}\} \; \forall \, i \in \{1, \ldots, N\},$$

   where each partition $C_k^{i,t}$ has its cluster center $\bar{x}_k^{i,t} = \frac{1}{|C_k^{i,t}|} \sum_{\alpha \in C_k^{i,t}} x_{i,t,\alpha}$. The $K \cdot |N|$ for ($K \cdot |T|$ for time dimension respectively) cluster centers are then clustered again. A high concentration of the first stage cluster centers, i.e. a clear partition, indicates a high stability of the agent types and their features.

   Figure 3.3 shows the cluster centers in their embedding in $\mathbb{R}^2$, with the color indicating the cluster from the first stage. Cluster centers of the same type are very much concentrated – e.g. all QUANT cluster centers are located closely together in the embedding. Just a few points are close to the cluster centers of other types. For some partitions, some clusters are closer to the RES point cloud.

   This is also confirmed by the meta clustering using $K = 4$ meta clusters. Both meta clustering across different tickers as well as different time periods shows very small confusion as shown in Table 3.6. Very few cluster centers are not correctly grouped into their corresponding meta clusters. The most prominent



(a) Different tickers, same time period.

(b) Same ticker, different time periods.

Figure 3.3: Embedding of first stage cluster centers from the single time slice clustering process. Colors indicate the first stage clustering result. As discovered in Table 3.3, C-Res is positioned in-between the other three clusters.

confusion occurs when clustering across different tickers. For three tickers, the Quant cluster is allocated to the Meta Residual cluster.

	Meta Quant	Meta Day VWAP	Meta Signal	Meta Res
Quant	22 (23)	0 (0)	0 (0)	**3 (1)**
Day VWAP	0 (0)	25 (23)	0 (0)	0 **(1)**
Signal	0 (0)	0 (0)	24 (24)	**1** (0)
Res	**1** (0)	0 (0)	0 (0)	24 (24)

Table 3.6: Affiliation of clusters to meta clusters for different tickers.

To further support the evidence for the agent types' stability, one may look at certain representative features from the clusters across different months or tickers. Figure 3.4 visualizes three of the clusters' representative features for 24 different months:

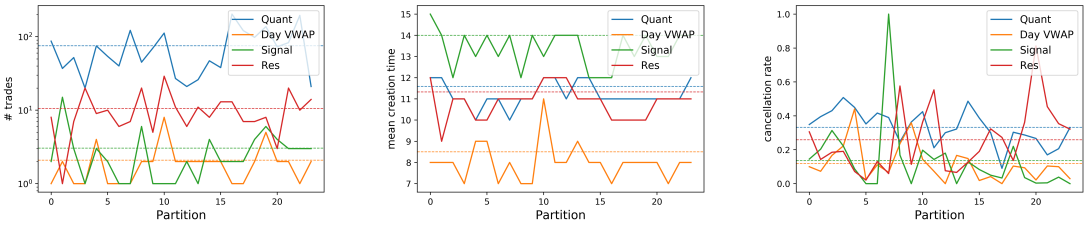
The average number of orders (Figure 3.4a) has shown to be one of the most relevant features in the clustering. Also the centroid values over time show strong discriminative behavior here. Quant is trading by far the most in all months, but one where Res trades similarly often. Day VWAP and Signal show a similar number of trades on the lowest level. This further supports our interpretation above and the embedding in Figure 3.3 that Res lies in-between all clusters, and sometimes much closer to Quant in terms of the number of trades.

The mean creation time depicted in Figure 3.4b shows both a discriminative and stable behavior, similar to the number of trades in Figure 3.4a. Quant and Res have a mean creation time around noon both types typically trade throughout the entire

day. In contrast, Day VWAP trades very early in the morning and thus shows a mean submission time much earlier than the remaining agent types. For Signal, the mean submission time for different months fluctuates to a certain degree around the US market opening. In general, the feature values from Day VWAP and Signal are well separated from Quant and Res.

For the cancellation, the situation differs. In general, the centroids' mean value fluctuates more and the time series of the values throughout different months are more overlapping. The dashed lines indicating the mean values of the different partitions still indicate the same ranking, where in particular Quant tends to have a higher cancellation rate indicating they more actively track the execution. The second highest is Res, which in several periods contains traders which tend to behave like Quant agents, potentially causing the increased cancellation rate. Day VWAP, apart from one large outlier, has the lowest cancellation rate of all clusters. In general, this feature illustrates that not all of the features used in the clustering process are stable or have the same ranking throughout different partitions.

The results of this section indicate a high stability of the agent types presented in section 3.2.3. It consequently leads to assume that the observations are not random, but rather follow a consistent pattern that exhibits different types of traders acting in the limit order market ecosystem. It is, however, not immediately clear whether the results are representative for other brokers. To ultimately verify this, a similar analysis of richer execution data sets consolidated from several brokers would be necessary, but highly unlikely to come by given the sensitivity of the data. Nonetheless, there are several reasons which make it rather unlikely that a similar analysis for another broker would lead to completely different results. Our data set does represent one of the largest market shares. This makes it likely that a broad range of different traders interact with this broker. Furthermore, the stability analysis is done over two years of data and shows very consistent results. Making the assumption that traders change brokers from time to time and do not solely trade via a single broker leads to a varying set of clients. The temporal stability then implies that even though traders may trade more or less with a certain broker, the partition remains quite stable over time. Hence, one may assume that the overall picture in the market is rather similar. A difference may be caused by the particular specialties of certain brokers. Consequently, we expect the agent types to be similar but potentially with different sizes, both in terms of agents but also in terms of traded volume.



(a) # orders (log scale) (b) Mean submission time (c) Cancellation rate

Figure 3.4: Exemplary centroid features of one ticker for several months. The solid lines indicate the actual values over the months, the dashed horizontal lines indicate the average over all months.

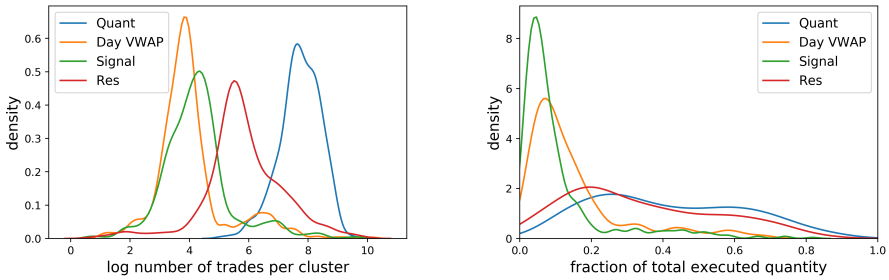
3.3 Decomposition of order flow components

This section reviews properties of the order flow components which were segmented by clustering the agents in section 3.2. This shall answer the question of whether the aggregated flows of each cluster show different also show different dynamics. This section builds a change of perspective as we do not look at different traders but aggregate over a particular cluster as one component. First, we look at the sizes and activities of the components' flow. Following this, child order properties, profitability, and the correlation of inventory with the price moves are analyzed to outline notable differences between the components.

Figure 3.5a illustrates the distribution of the number of orders for each of the order flow components, throughout 25 different instruments of the *STOXX 600* in two years. The distributions differ substantially. While Day VWAP and Signal show a similar distribution in terms of numbers of trades per month, Quant exhibits the largest numbers of orders. Res is once more located in-between the Quant and Day VWAP/Signal. This matches and further intensifies our previous interpretation that Res is some mixture of the first three agent types. Table 3.7 also indicates that the majority of the trades are done by Quant, and only a small fraction by Day VWAP and Signal.

The fraction of the total executed quantity from each of the single data slices is indicated in Figure 3.5b. The distributions of the fractions from Quant and Res show a similar shape. The same holds for Day VWAP and Signal. In fact, the sum of the executed quantity from Quant and Res does not vary much due to their strong negative correlation. The fraction of Res tends to increase when a trader from Res behaves rather like a Quant trader or potentially as a mixture of Quant, Day VWAP and Signal. In comparison to the very small number of orders submitted

by Day VWAP and Signal, the actual executed quantity is much higher. In other words, while Day VWAP and Signal tend to account for only a small fraction of the number of orders, their contribution to the overall executed quantity is much larger since the orders are generally larger and/or lead to higher execution size (for example, due to fewer cancellations and longer execution). The mean fractions are displayed in the "Executed qty" row of Table 3.7.



(a) Number of submitted orders (b) Fraction of executed quantity

Figure 3.5: Distribution of the number of trades and the executed quantity.

	C-Quant	C-Day VWAP	C-Signal	C-Res
Number of orders	0.69	0.06	0.06	0.20
Executed qty	0.41	0.15	0.11	0.34

Table 3.7: Average values of the distributions of clients, trades and executed quantities across different clusters

3.3.1 Heterogeneity of child orders

This section reviews the differences between the four different components regarding the child orders which are sent to different venues. This may give further insights into the degree to which the order flow components differ not only at parent order but also at the child order level. Table 3.8 shows a summary of the orders submitted to exchanges for one exemplary $B_{i,t}$. These sample results are for BATS.L for the month of December 2019. The numbers are normalized either by the mean of the corresponding statistic or by their sum.

In line with the observations regarding the number of orders and executed quantity from Figure 3.5a and Figure 3.5b, the number of child orders submitted by Quant is by far the largest. Furthermore, the quantitative agents have the highest direct market access (DMA) ratio. This means these child orders come from parent orders

that skip the processing of the broker's execution algorithms, hence primarily using the broker as a platform to send their orders to a venue. The DMA ratio is also high for the SIGNAL component. In this particular segmentation, SIGNAL contains one agent which heavily and exclusively trades with DMA orders leading to both an unusually high DMA and cancellation rate. This may be an agent which rather belongs to the QUANT cluster and impacts the statistics of the SIGNAL order flow component. For the RES and DAY VWAP clusters, the DMA ratio is almost zero, and only around 75% of the child orders get canceled.

Regarding the order and execution sizes of child orders, the orders of QUANT have the lowest mean and standard deviation, as indicated in Table 3.8. SIGNAL has by far the highest mean order and execution quantity. A significantly higher mean compared to the median indicates the presence of outliers for all clusters, which might be caused by larger orders placed in dark venues. In particular, component SIGNAL exhibits many executions in dark venues for this month as mentioned in section 3.2.3, hence rendering a large mean order quantity more plausible. As for the median, the execution quantities are significantly lower than the order sizes. One reason for this may be partially filled orders – in particular, in dark venues. Note that we exclude child orders without any partial fills for the present statistics.

Median Distance to Best Price indicates the relative price level at which the child orders of the particular component tend to be placed (i.e., this can be construed as a proxy for aggressiveness or urgency). Again, the median is displayed due to its robustness. QUANT child orders are placed closest to the best price. The median distance from the best price for RES and SIGNAL is roughly similar, while the orders from DAY VWAP tend to be placed when compared to child orders from the other clusters.

Lastly, we look at the daily net inventory of the different components' child orders, the sum of the signed traded sizes (positive for buy, negative for sell) submitted by a cluster during one day. Table 3.9 indicates the mean cumulative inventory for one exemplary month. In particular, QUANT appears to have been a net seller, while the remaining clusters were net buyers in that particular month during which the corresponding stock showed a positive return.

Tables 3.10a and 3.10b indicate the correlation between the components over the whole duration and all instruments for both the net inventory as well as the order flow imbalance[3]. Clusters SIGNAL and RES exhibit the strongest correlation.

[3]The order flow imbalance is computed with the net inventory divided by the total trade volume of the component. A more detailed explanation can be found in section 3.3.3.

	C-Quant	C-Day VWAP	C-Signal	C-Res
# Child Orders	0.88	0.02	0.03	0.06
Mean Order Qty.	0.23	0.42	2.74	0.61
Median Order Qty.	0.29	0.28	3.11	0.33
Std Order Qty.	0.52	0.61	2.02	0.85
Mean Exec Qty.	0.12	0.18	3.53	0.17
Median Exec Qty.	0.70	0.75	1.76	0.79
Std Exec Qty.	0.33	0.40	2.95	0.31
DMA Ratio	2.27	0.03	1.63	0.07
Cancellation Ratio	1.07	0.90	1.09	0.94
Median Distance to Best Price	0.52	1.81	0.90	0.77

Table 3.8: Aggregated statistics of child orders by clusters. This table shows the total number of submitted child orders, the ratio of orders which were sent due to direct market access, the ratio of child orders which were canceled, the mean, the median, and the standard deviation for both order quantity and the executed quantity of orders. The last row shows the median distance of a limit order from the best opposite price. The number of child orders is normalized by the sum of all clusters; the remainder of the statistics are normalized by the mean of the four clusters.

	C-Quant	C-Day VWAP	C-Signal	C-Res
Average inventory	-0.62	0.12	0.11	0.16

Table 3.9: Table indicating the mean cumulative inventory of the clusters. Values are normalized with the absolute inventory of the clusters.

However, the correlations obtained are mostly statistically insignificant. Regressing the net inventory or the order flow imbalance of one cluster on any of the three other components, the coefficients rarely show any significant deviations from zero on an instrument basis. For only very few instruments, weak significant correlations can be found, but the average p-value over all instruments considered here is around 0.25. We furthermore fit regressions over several instruments but a smaller time horizon, under the assumption that the correlations may change over time. Again, few variables show a persistent significance throughout time and the resulting R^2 are very low. This indicates that, despite the significance of a few variable combinations, the explanatory power is small.

The resulting both inconclusive as well as insignificant correlation between the net inventories (OFIs respectively) indicates that not only does the behavior between the clusters differ quite substantially but they are also independent of each other in the way they accumulate inventory. One possible reason for this may be that different

	C-Quant	C-Day VWAP	C-Signal	C-Res
C-Quant	1.00	-0.05	-0.02	-0.04
C-Day VWAP	-0.05	1.00	0.00	-0.03
C-Signal	-0.02	0.00	1.00	-0.09
C-Res	-0.04	-0.03	-0.09	1.00

(a) Net inventory.

	C-Quant	C-Day VWAP	C-Signal	C-Res
C-Quant	1.00	-0.07	0.00	-0.11
C-Day VWAP	-0.07	1.00	-0.02	-0.04
C-Signal	0.00	-0.02	1.00	-0.03
C-Res	-0.11	-0.04	-0.03	1.00

(b) Order flow imbalance.

Table 3.10: Tables indicating the correlation of the order flow between different components.

trader types trade different strategies which are either not correlated at all (leading to insignificant correlations). Another explanation would be that changes depend on the market environment as some strategies may only correlate during certain market conditions. In particular the first makes sense since the trading types seem to act on different time scales in the market. The most persistent observation is a slight negative correlation between Quant and the remainder of the order flow components which, however, is relatively weak.

3.3.2 Profitability

In section 3.2, we showed that traders using execution services may be summarized into different clusters. These trader types have different properties when it comes to the type of orders they send. This leads to the assumption that the objectives of the segmented agent types may differ as well, for example, with respect to their horizon of investment. To this end, we analyze the hypothetical profit and loss (PnL) for each order flow component, in order to investigate structural differences in the returns of the components' trades. We remark that this PnL is hypothetical as it does not refer to the actual inventory of the trader. For example, it may be that a trader is not holding a position for the respective future horizon of time, or that it is actually unwinding a short position instead of building a long position, or the holding period is different. It much rather represents the average PnL of the respective component, at a certain fixed time horizon.

To investigate the hypothetical profitability of each component, we compute the PnL of each trade via

$$PnL_t^l = -\text{sign}(q^{target}) \log\left(\frac{p_{t+l}}{p^{exec}}\right), \tag{3.8}$$

where p^{exec} is the volume-weighted execution price of the corresponding parent order. q^{target} is the target quantity of the parent order, as specified in definition 2.1.8, and $-\text{sign}(q^{target})$ the negative sign of the return. If, for example, $q^{target} > 0$ the order is a sell order, thus we multiply the log return with -1. For some time step $t + l$,

p_{t+l} denotes the closing price at $t+l$, where we consider trading days as increments. For $l = 0$, p_t corresponds to the close price of the day when the corresponding trade happens.

The volume-weighted average price of a parent order's execution p^{exec} is computed via

$$p^{exec} = \frac{1}{q^{exec}} \sum_{x \in \mathcal{X}^{exec}} q_x \cdot p_x, \tag{3.9}$$

where q_x and p_x indicate the quantity and the price of each execution in $\mathcal{X}^{exec}$ of the corresponding parent order. Finally, we compute the expected PnL for each component $k \in \{1, \ldots, K\}$ of trades at day t, as $\mathbb{E}(PnL^l_{t,k})$ by averaging all returns for a given l, t, k. The result is a daily time series for different lags $l \in \{0, 1, 10, 20\}$ and different components $k \in \{1, \ldots, K\}$, where each element is the average PnL of the trades occurred on that particular day. The same computation is done using market excess return, where we subtract the future market return from the instrument's future return, for the PnL computation in Equation (3.8).

Table 3.11 indicates the average expected PnL in basis points for 2018 and 2019 for 25 instruments. From trade to close, Quant appears to be the only order flow component generating a slight profit over the period covered here. This indicates that Quant is potentially pursuing more intraday/short-term strategies. This is supported by the statistical significance of the market excess return of Quant for $l = 1$. For Signal, the 20-day return (both raw return and market excess return) is the largest and furthermore significantly different from 0 to a confidence level of 95%. Even though this is not a realized return, it gives reason to assume this agent type has some medium-frequency edge it is trading. Furthermore, Signal showing the smallest PnL on the same trading day ($l = 0$) supports our interpretation of Signal given in section 3.2 that this component is not aiming for a profit that realizes within the same day. It may well be that the Signal agent type trades mean reversion patterns which realize only after around a month. Apart from that, Day VWAP shows a slight outperformance on the $l = 1$ horizon.

Additionally, Figure 3.6 shows the cumulative hypothetical PnL for each component. In line with the observations from Table 3.11, Signal is clearly outperforming the other agent types on a 20-day horizon (lower right plot) while having the worst performance from trade to close and to $t + 1$ (upper plots). Nonetheless, all Sharpe ratios (risk-adjusted returns) are very close to zero.

	$l=0$	$l=1$	$l=10$	$l=20$	$l=1$, excess	$l=10$, excess	$l=20$, excess
QUANT	**0.38**	0.72	3.39	3.76	0.78*	3.78	2.83
DAY VWAP	-0.25	**1.54**	-2.04	-3.95	**1.42**	0.15	1.44
SIGNAL	-0.47*	-0.43	3.09	**13.17***	0.32	2.65	**10.63***
RES	0.08	0.37	-1.36	-1.51	0.52	-2.24	-2.19

Table 3.11: Table indicating the average $\mathbb{E}(PnL_k^l)$ for agent types in basis points (bps). The suffix *_excess* indicates market excess returns over the *STOXX 600* baseline. A $*$ behind the return in basis points indicates significance using a one-sample t-test and with a confidence level of 95%.

(a) $l = 0$ (b) $l = 1$

(c) $l = 10$ (d) $l = 20$

Figure 3.6: Cumulative PnL $\mathbb{E}(PnL_{t,k}^l)$ of agent types over different time horizons, $l = \{0, 1, 10, 20\}$.

3.3.3 Order flow imbalances during volatile periods

While some components show differences in their expected returns with respect to the trading horizon, it stands to question to which degree the components' order flow on a particular day is correlated with the return of the day. In particular, how does the order flow of different agent types behave when markets move substantially? To this end, we compute the order flow imbalance (OFI) for each instrument via the following measures

$$OFI^{C_k} = \frac{\sum_{p \in \mathcal{X}^k} q^{exec}}{\sum_{p \in \mathcal{X}^k} |q^{exec}|}, \tag{3.10}$$

and

$$OFI^{ADV} = \frac{\sum_{p \in \mathcal{X}^k} q^{exec}}{\text{ADV}}, \tag{3.11}$$

where $\mathcal{X}^k$ is the set of parent orders from cluster k on a given day. We compute the imbalance of the order flow in two ways. The first imbalance shown in Equation (3.10) is normalized by the total executed quantity of the cluster, hence called OFI^{C_k}. The second imbalance shown in Equation (3.11) is normalized by the average daily volume (ADV) from the last 20 days and denoted as OFI^{ADV}. This is to take into account that a large OFI, as defined in Equation (3.10), does not necessarily mean a large impact on the market during the particular day. That is because the total traded volume of the cluster might only be a small fraction of the daily volume. In contrast, Equation (3.11) makes different values of the same day more comparable across different clusters and is large only if a component's net inventory is large in relation to the historical ADV. One cluster might have a large OFI in terms of its own orders but a very small OFI measured on the ADV due to a small traded volume.

Figure 3.7 shows a histogram of the OFI as in Equation (3.10) to simplify comparison. Quant stands in contrast to the other three components. The net order flow peaks around zero, while the other order flow components have two peaks at zero and one. In particular, the aggregated order flow from Quant tends to be rather neutral in terms of order flow imbalance, which can be due to two reasons. First, Quant trades much more often (albeit smaller sizes) and thus facilitates order flow imbalances closer to zero. Second, Quant pursues more intraday/medium frequency strategies without accumulating larger positions.

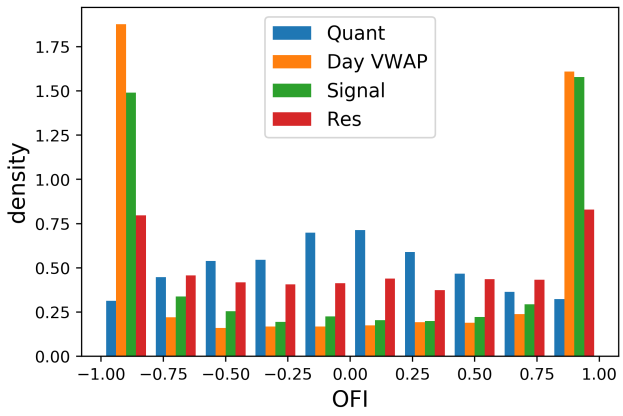


Figure 3.7: Order flow imbalance by cluster, during days of returns with large magnitude.

The correlation of the OFI with the log return from open to close (logOC) is shown in Table 3.12a (via Equation (3.10)) and in Table 3.12b (via Equation (3.11)). Day VWAP and Res show the clearest picture. For both the case of days of large

returns of the particular instrument (left columns), as well as days of large returns of the index (right columns), Day VWAP and Res exhibit a fairly strong positive correlation. This holds for the net inventory scaled by the traded volume of the particular component (Table 3.12a), as well as the net inventory scaled by the market volume (Table 3.12b). In fact, the correlations of OFI^{ADV} with both the instrument and the index return are larger than the correlation of OFI^{C_k} with the return. In other words, when these clusters accumulate net inventory which is also large in terms of the average daily volume, the instrument is likely to also exhibit a larger return. A possible reason for this may be that the order flow becomes a driving factor in the market.

cluster	logOC	logOC_index
C-Quant	-0.0301	0.0113
C-Day VWAP	**0.1590**	0.0229
C-Signal	**-0.1506**	0.0499
C-Res	**0.1102**	**0.1616**

(a) OFI^{C_k} as in Equation (3.10).

cluster	logOC	logOC_index
C-Quant	0.1368	0.0833
C-Day VWAP	**0.2044**	0.1416
C-Signal	**-0.0441**	**-0.0794**
C-Res	**0.2189**	0.2390

(b) OFI^{ADV} as in Equation (3.11).

Table 3.12: Correlation between OFI and returns of stocks (column logOC) and return of the index (column logOC_index), for two different types of normalized OFI.

Signal is the only cluster exhibiting a fairly large negative correlation with negative returns of the instruments. This correlation seems to be less strong when the net inventory is large as defined in Equation (3.11). A reason for this may be that due to the high participation rate, Signal becomes a driving factor of the market, similar to Day VWAP and Res above. The correlation with the index is less clear and varies between OFI^{C_k} and OFI^{ADV}. However, it seems that the index is more likely to decrease if the net inventory is large, also based on the average daily volume of the corresponding stock.

Quant shows the lowest correlation of net inventory scaled with total trade size, with both the daily return as well as the index (-0.03 and 0.01, respectively). This is to be expected, taking into account Figure 3.7 that shows the tendency of Quant to keep a net inventory closer to zero. For net inventories which are large if measured on the total market volume following Equation (3.11), however, the correlation also seems to turn stronger. Similar for Day VWAP and Res, if the net inventory is large when measured on the ADV, the cluster becomes more of a market driver, and a higher correlation with the daily return can be observed.

3.4 Heterogeneous parent order model

As illustrated in the order process in Figure 1.1, limit orders sent through execution services are usually part of a larger parent order. This section proposes a simple model which captures the most important heterogeneous properties for each of the components extracted in Section 3.2. Once these parent orders can be modeled, their scheduling and execution in the LOB may be simulated. This can be done by replicating the flow of the parent orders into the LOB as depicted in Figure 1.1 via known execution and order routing algorithms, some of which are well studied in the literature. Modeling the flow of these components itself without the direct scheduling and execution can additionally be of high interest to brokers, as this can possibly improve the broker's knowledge and service to clients. Rather than aiming to replicate and fit the data to the full extent, the following model is designed to outline a starting point which has good fits from a marginal perspective for several stocks, despite being very simple. It also further underpins the structural differences between the different clusters which have been outlined in the previous sections. Furthermore, we assume independence between the flows presented in the following, due to the absence of a significant correlation structure between the components, as detailed in Table 3.10.

3.4.1 Quant

As outlined in Section 3.2, the QUANT order flow component is submitting orders throughout the day. Figure 3.8c suggests the U-shaped intra-day pattern commonly observed in trading behavior of limit order books (Cont, 2011). In the morning and towards the closing time of the market, the intensity of the orders increases. QUANT order sizes are of mostly small size and typically executed in just a few child orders. This, however, does not imply that only one order is sent to the exchange.

As per modeling the QUANT order flow component, we suggest a non-homogeneous Poisson process. In particular, $\{N^{Quant}(t), t > 0\}$ denotes the counting process of the parent order submissions indicating the number of submitted parent orders up to time t. The arrival time of the n-th parent order is denoted as t_n^{Quant}. New orders arrive proportionally to the conditional intensity rate $\lambda^{Quant}(t)$, $t \in [0, T]$ similar to the shape in Figure 3.8c. This is to properly reflect the intraday pattern of the QUANT order arrivals. It must hold that

$$\int_0^T \lambda^{Quant}(s)ds = 1,$$

so that the expected number of orders within $[0, T]$ equals one. The conditional intensity function $\lambda^{Quant}(t)$ is then scaled by the number of expected order submissions for the corresponding day

$$\lambda^{Quant}(t) \cdot n^{Quant}.$$

Each of the arriving parent orders additional "marks" or properties as specified in Definition 2.1.8, in particular a sign and a target quantity. For the QUANT parent orders, the following considerations are in place

- the logarithm of the expected number of parent orders, parameter N follows a skewed normal distribution, $\log(n^{Quant}) \sim sN(a, \mu, \sigma^2)$ with skewness a, mean μ and variance σ^2 (Figure 3.8a),
- the target quantities are fitted with a Laplacian distribution, i.e. $q^{target} \sim \text{Laplace}(\mu, b)$, as illustrated in Figure 3.8b,
- the signs of the order is Bernoulli distributed $sign_i \sim B(1, p)$,
- the probability of an order being a buy order, p, differs across days and is modeled by a Beta(a, b) distribution; Figure 3.8d indicates the fit.

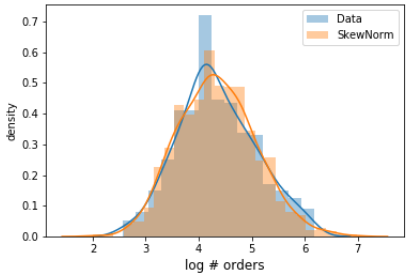
The execution of the QUANT orders is done with the Almgren-Chriss optimal execution framework, first presented in (R. Almgren & Chriss, 2001). The execution generally consists of only very few if not just one child order and the execution time is very short.

3.4.2 Day VWAP

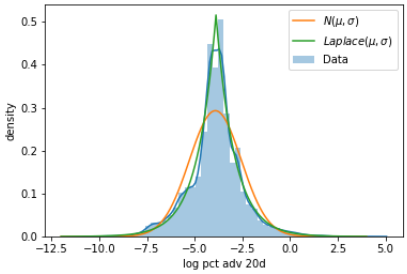
As outlined in Table 3.3, DAY VWAP mainly sends parent orders in the early morning around the market opening. The total number of parent orders is quite small, while execution generally takes place throughout the entire day. A model for these orders is thus quite simple and can be done without any time dependence since we assume all orders to be submitted at market open (i.e. $t = 0$). The orders are then executed throughout the day proportional to some estimated volume profile.

For DAY VWAP, it suffices to know how the sum of all buy (respectively, sell) orders is distributed, which is then executed as one large buy order (respectively, sell order). Denoting $X^{DayVWAP}$ as the set of all parent orders from the DAY VWAP component for a given day, the quantities of interest are

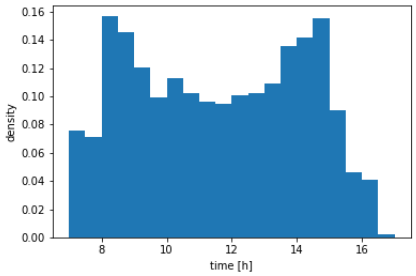
$$\sum_{p \in X^{DayVWAP}} (q^{target})_- \quad \text{and} \quad \sum_{p \in X^{DayVWAP}} (q^{target})_+,$$



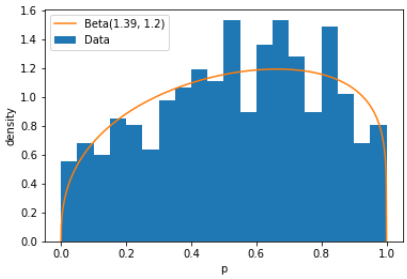
(a) Order counts: QUANT order flow.



(b) Order size distribution: QUANT order flow.



(c) Submission times



(d) Fraction of buy orders

Figure 3.8: Aggregated parent order flow distributions for one stock, over two years, for QUANT.

where the first term builds the cumulative size of all DAY VWAP buy orders, and the second term all DAY VWAP sell orders. The DAY VWAP component hence consists of two (aggregated) parent orders with

- submission time $t = 0$,
- target quantities for both orders, where the logarithm of the absolute value, $\log(q^{target}) \sim sN(a, \mu, \sigma^2)$ where $sN(a, \mu, \sigma)$ follow a skewed normal distribution with skew a, mean μ and variance σ^2 (as shown in Figure 3.9),
- the sign of the order, which is -1 for the aggregated buy order and 1 for the aggregated sell order.

The aggregated buy and sell DAY VWAP orders are executed following the volume profile of the previous day.

3.4.3 Signal

The SIGNAL cluster as outlined in Table 3.3 sends quite large orders and executes them in a relatively short horizon. This gives reason to assume SIGNAL is a more

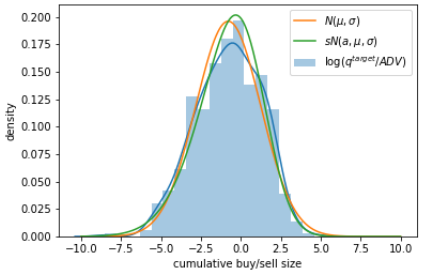


Figure 3.9: The aggregated parent order sizes (on a log scale) for one stock over two years for DAY VWAP.

aggressive player in the LOB, moving the LOB in a certain time horizon.

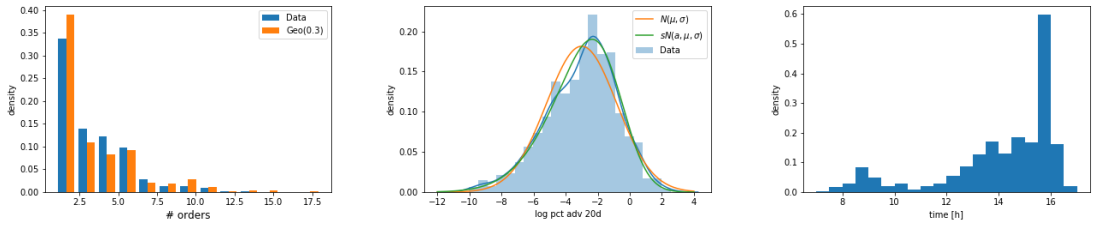
As per modeling the SIGNAL component, we suggest a non-homogeneous Poisson process similar to QUANT. As before, $\{N^{Signal}(t), t > 0\}$ denotes the counting process of the parent order submissions indicating the number of submitted parent orders up to time t. New orders arrive proportionally to the conditional intensity rate $\lambda^{Signal}(t),\ t \in [0, T]$, similar to the shape in Figure 3.10c. The conditional intensity function $\lambda^{Signal}(t)$ is then scaled by the number of expected order submissions for the corresponding day

$$\lambda^{Signal}(t) \cdot n^{Signal}.$$

The SIGNAL parent orders come with the following "marks" or properties

- the expected number of arriving parent orders at a day, n^{Signal}, follows a geometric distribution, i.e. $n^{Signal} \sim \text{Geo}(p)$ (Figure 3.10a),
- the logarithm of the target quantities SIGNAL parent orders $\log(q^{target})$ is skewed normal distributed, $\log(q^{target}) \sim sN(a, \mu, \sigma^2)$ with skew a, mean μ and variance σ^2 similarly to DAY VWAP (Figure 3.10b).
- the sign of the orders $sign_i \sim B(1, p)$, where p denoted the probability of a buy orders.

A large proportion of the SIGNAL component consists of close related orders. These orders are removed because they do not form part of the continuous trading session. We suggest the execution of the SIGNAL orders is done similarly to the QUANT orders using the Almgren-Chriss framework (R. Almgren & Chriss, 2001). The detailed execution, however, is not part of this work. Note, orders from SIGNAL are substantially larger than those from QUANT, which – together with the frequency of orders and the intraday pattern – constitutes the main difference between the two clusters.



(a) Order counts (b) Order sizes (c) Submission times

Figure 3.10: Aggregated parent order flow distributions for one stock over two years for SIGNAL.

3.4.4 Res

Both the representative features in Table 3.3 as well as the embedding of the first stage clustering in Figure 3.3 indicate that RES lies in between the remainder of the other clusters. In addition, the confusion matrices in Table 3.6 confirm that whenever there exists instability in the market segmentation, one of the remaining clusters is mistaken with RES. To this end, we model RES as a random mixture of QUANT, DAY VWAP and RES. We suggest random attribution of the mixture following a Dirichlet distribution

$$f(x) = \frac{1}{B(\alpha)} \Pi_{i=1}^{3} x_i^{\alpha_i - 1} \quad \text{where} \quad B(\alpha) = \frac{\Pi_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{K})} \tag{3.12}$$

and $x = (x_1, x_2, x_3), \alpha = (\alpha_1, \alpha_2, \alpha_3)$ are the weights between the three clusters. The random allocation is then proportionally added to the first three components.

3.4.5 Clustering synthetic parent order flow

This section aims to illustrate the parent order flow model and support the results from Section 3.2. To this end, we simulate parent orders of the three different types described above, namely QUANT, DAYVWAP, and SIGNAL flows. We do not include RES as this cluster is assumed to be a composition from the first three clusters as explained above. From the generated parent orders, we extract features as those shown in table Table 3.3. Then, the features of the simulated flows are then clustered. This is to show that flows of the same type will end up in one cluster in order to provide further validation for the results of this work. Furthermore, the clusters should have a similar interpretation. We remark, that not all features can be extracted such as the cancellation rate of orders as we did not study under which circumstances orders of a particular agent type are canceled. Hence, not all of the features can be constructed

for which we either use the representative feature from Table 3.3 or do not cluster on that feature. This can be specified in the code which is available online.

Table 3.13 shows the confusion matrix of the partition and indicates in which cluster the simulated flows end up. The rows indicate the simulated agent flows and the columns indicate the clusters in which these end up. Table 3.13 shows no confusion at all. This means all simulated flows are allocated to their corresponding cluster as expected. This holds for both K-Means as well as spectral clustering.

	Quant	DayVWAP	Signal
Quant agents	25 (25)	0 (0)	0 (0)
DayVWAP agents	0 (0)	25 (25)	0 (0)
Signal agents	0 (0)	0 (0)	25 (25)

Table 3.13: Affiliation of simulated flows with the corresponding clusters.

Figure 3.11 shows the spectral embedding of the simulated agent flows similar to Figure 3.3. Similar to the confusion matrix, the simulated agent flows are well separated in the embedding space which explains the clear partitioning in Table 3.13. Both the confusion matrix as well as the embedding support the heterogeneity of the flows extracted in this work which persist also when simulating these.

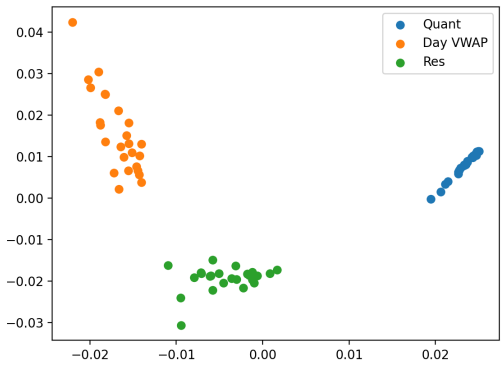


Figure 3.11: Spectral embedding of simulated agent flows.

3.5 Conclusion

Many different agents act together in limit order books with different intentions, trading horizons, and information sets. This gives reason to assume that order flow in limit order books is not homogeneous but rather of different types. To account

for different agents submitting orders an alternative notation for limit order books was given in Section 2.1. This notation allows to derive different views on the order book with different granularity ranging from an anonymized *public view* to the fully detailed *omniscient view*.

To investigate the heterogeneity of those agents which make use of brokers, trade execution data was analyzed. Results show these agents may be summarized in four representative clusters which differ substantially in both their trading behavior as well as the order flow induced in the limit order book by the parent orders. In particular, trading frequency, trade size but also order submission time and execution strategies show notable differences between these agent types which gives evidence that some heterogeneity may be assumed. The insights were used to propose a simple model for the parent order flow of each cluster in order to capture some of the heterogeneous dynamics of the different trader types in Section 3.4.

In relation to (Kirilenko et al., 2017), the results may be seen as additive rather than comparative. In particular, the results presented in this work refer more to the "fundamental buyers/sellers" and "opportunistic traders" classes from (Kirilenko et al., 2017) as the data in this study excludes HFT and market maker agents which have their own execution platforms. Kirilenko et al. (Kirilenko et al., 2017) mainly focus on HFTs and MMs in their study. Looking at the entire market one thus would have to aggregate both studies to get the full picture of the heterogeneity in limit order markets.

In contrast to (Kirilenko et al., 2017), this study is able to distinguish agents on their actual parent orders for several tickers and a longer time horizon. This enables us to show that the agent types presented in Section 3.2.3 are consistent over longer time periods and also exist in different stocks. The only noteworthy confusion factor is between C-Res and some of the other clusters which seem to move towards C-Res under certain conditions.

The results of this study and (Kirilenko et al., 2017) provide evidence that order flow in limit order books shows strong heterogeneity indicating that modeling LOBs under the assumption of homogeneity is not very valid. In contrast to most order flow models in the literature, such models should hence include some degree of heterogeneity.

This study builds a first step toward a better understanding of limit order markets and their heterogeneity. We focus on the parent order process, the very left-hand side of the order process depicted in Figure 1.1. Our results provide a foundation for many future research directions. The interplay between traders with proprietary access

to the exchange (HFTs and MMs) and traders acting through brokers is yet to be analyzed in detail. The authors of (Kirilenko et al., 2017) remark, for instance, that HFTs keep their trading patterns stable, also in times of increased market volatility. More detailed studies are yet missing. It is also to be investigated how exactly orders are processed and arrive in the LOB. This would correspond to the center part of Figure 1.1. While this study provides indications in which fashion each agent type tends to execute orders, it is not exactly clear how much liquidity from each venue tends to go to the lit or dark venues, etc. Lastly, the combination of parent order flow and HFTs and MMs may be used to create heterogeneous models for entire LOBs. Under the assumption of similarity of the trader structure between different brokers, these order flows may be scaled up to the entire volume of all brokers. It remains to be investigated whether realistic LOB models can be developed, which intend to incorporate the entire process shown in Figure 1.1.

Chapter 4

Limit Order Book Simulation with GANs

4.1 Introduction

Most exchanges nowadays organize trading activities via a centralized limit order book (LOB). These LOBs are records of outstanding limit orders (LOs) indicating the willingness of an agent to either buy or sell a certain quantity of an asset at a certain price. Many different agents with different objectives and trading styles interact in the LOB, thus leading to complex dynamics. Such dynamics and their statistical properties have been extensively studied in the literature. Models have been built to capture and reproduce the observed dynamics. In particular, synthetic data is of high value to practitioners, allowing them to build more robust execution algorithms, and also more rigorously test the performance of various models and trading strategies.

In most settings, either event-based or queue-based models have been developed in the literature. In the setting of ultra-high frequency applications, it is indispensable to model the entire event stream, since algorithms act on these single events. However, for many practical applications in mid-to-high frequency settings, it suffices to model the order book snapshots at certain regular time intervals.

In this study, we address the modeling of time series of LOB snapshots which has received less attention in the literature. More precisely, we leverage Generative Adversarial Networks (GANs) in order to learn the conditional probability distribution of the transition between two consecutive LOB snapshots, and thereby implicitly the price changes in the LOB. The aim is to match both unconditional as well as conditional metrics.

1. Unconditional metrics: Using the trained GAN to generate synthetic data, are unconditional stylized facts recovered?

2. Conditional metrics: The primary aim of a generative LOB model is to interact with it in order to model the effect of the interaction on the system. Hence, interacting with the LOB state should have similar effects as studied in the literature. This particularly concerns the market impact of executions, including the well-known square root law (Gatheral, 2010). This is studied via both limit order submission (liquidity provision) as well as the submission of market orders (liquidity extraction).

Our proposed approach to model LOB snapshots is able to reproduce a number of statistical properties of typical LOB snapshot time series, such as distributions, correlation structures, and price dynamics. Additionally, the price process is implicitly in line with the quantity process due to the design of the network and its input structure. Finally, the model – once trained to a sufficient accuracy – is able to reproduce commonly known patterns from the market impact literature. Interacting with the LOB state shows effects such as the expected market impact and the square root law (Gatheral, 2010) when changing the quantity of injected orders. We find this particularly interesting as the model is not explicitly being trained to do so, and furthermore, it only learns the dynamics of the LOB snapshots. This work opens many future research directions for the improvement of LOB snapshot simulation, mostly regarding the inclusion of more history, stabilization of the model convergence, support of sparse LOB snapshots for empty levels, and intraday effects.

The remainder of the paper is structured as follows. Section 4.2 presents related work from the LOB literature. Limit order books and the object modeled are presented in Section 4.3. Section 4.4 reviews Wasserstein Generative Adversarial Networks and outlines the design for the model in the present study. Unconditional metrics of stylized facts are presented in Section 4.5 and results of interaction with the simulator are shown in Section 4.6. Section 4.7 concludes and describes further extensions of this research direction.

4.2 Related work

Every action in a LOB on an exchange is recorded and stored which leads to an extensive amount of data. The availability of such LOB data for a large number of years has led to a vast amount of research articles in many directions as presented

in Chapter 1. One part of the research directions deals with modeling the dynamics of LOBs and simulating the entire order book instead of only e.g. the price process. The behavior of such a model when interacting with it and what effect it has on the order book is often of central question in these studies.

An overview of limit order book modeling is given in (Cont, 2011; Gould et al., 2013). Empirical studies on properties of LOB include (Cont, 2001; Bouchaud et al., 2002; Potters & Bouchaud, 2003; Cont, 2011). Due to its large relevance for industrial applications, prediction of future price changes in LOBs and traded volumes have been conducted. More recently, (J. A. Sirignano, 2019; J. Sirignano & Cont, 2019; Zhang et al., 2019) have used large amounts of LOB data in order to predict future price movements using different neural network architectures.

Synthetic data becomes more and more important in financial applications and it is important to both avoid mistakes when building models to create synthetic data and also to assess whether the data is realistic enough. (Vyetrenko et al., 2019; Cohen, Snow, & Szpruch, 2021; Assefa et al., 2020; Jordon et al., 2022) among other works deal with these issues. (Vyetrenko et al., 2019) review particular metrics when building synthetic limit order book data. (Cohen et al., 2021) deals with several risk factors when applying black-box models in finance, and also when building simulators. (Jordon et al., 2022) shed light on synthetic data from a broader perspective. (Assefa et al., 2020) emphasize the importance of realistic synthetic data in the financial world.

The main focus of many studies in the literature is the development of methods for modeling the dynamics in limit order books on an *event* base. To this end, researchers predominantly used point processes, namely Poisson processes for example in (Cont et al., 2010; E. Smith et al., 2003; Huang et al., 2015). Order flow models driven via Poisson processes do not perfectly reproduce the market dynamics, in particular with respect to inter-arrival times and cross-excitation behavior. However, their simplicity allows to derive quantities which are of potential interest. For instance, (Cont et al., 2010) derive analytical solutions for the probability of a price increase, the execution of a limit order before a mid price move, and the execution of both a buy and sell limit order before a mid price move. To account for dynamics dependent on the queue size, (Huang et al., 2015) introduce a queue-reactive model with arrival intensities dependent on the queue state. (Huang et al., 2015) argue order arrival rates in particular depend on the queue size and furthermore whether there is a spread larger than one tick.

Hawkes processes (Hawkes, 1971) have been used in finance for many applications. They offer the possibility to model an intensity process in which events of a certain type increase the intensity for the same type of events (self-excitation) or other events (cross-excitation). This intensity process is modeled under the use of a kernel describing the excitation effect as well as its decay. (Abergel & Jedidi, 2013, 2015; Morariu-Patrichi & Pakkanen, 2022) use Hawkes processes in order to model the order flow in order to account for the clustering and excitation behavior in LOBs. (Morariu-Patrichi & Pakkanen, 2022) introduced state dependency to better model dependence properties of the order flow conditioned on either spread or order imbalance.

On the other end, the computer science community has developed agent-based models (see Chapter 2) which assume different types of agents that act together in the LOB (Paddrik et al., 2012; Byrd et al., 2019). The main difficulties with agent-based models consist of defining the agents and calibrating them such that the joint simulation leads to realistic markets. Despite each single agent type generally being interpretable and rather simple, the joint dynamics are usually very complex which makes calibration a challenging task. (Paddrik et al., 2012) is particularly interesting due to the fact they use agent types and calibrations baked by an empirical study of the flash crash (Kirilenko et al., 2017). Further studies of agent-based models include (Farmer & Foley, 2009) and (Wang et al., 2021). (Farmer & Foley, 2009) build a zero-intelligence agent-based model and (Wang et al., 2021) empirically the effect of spoofing in an agent-based model. (Cont, Cucuringu, Glukhov, & Prenzel, 2023) attempt to shed light on different types of agents analyzing and modeling the order flow of a major broker.

All the previously mentioned studies model the queue sizes on an order-by-order basis. Alternatively, (Cont & Müller, 2021) model the LOB state as a density which evolves over time via a stochastic partial differential equation (SPDE). The SPDE is meant to explain the net changes of the density over time instead of creating a discrete order stream which leads to changes in the LOB. In particular, they assume that events in LOBs may be summarized into a net change in the order book. Each factor in the SPDE represents certain properties, such as high-frequency traders which lead to fluctuations similar to Gaussian noise or traders replacing their orders closer to the mid price.

With recent advances in machine learning and the increased availability of computational resources, the focus has recently shifted to more data-driven methods. In particular, generative adversarial networks (GANs), introduced in (Goodfellow et al.,

2014) have gained a lot of attention for synthetic data generation, mostly in the context of speech and image data. The GAN framework consists of two neural networks competing against each other, thereby improving the quality of the generated data until a Nash equilibrium is reached. Properly trained, GANs have shown the ability to learn complex structures in the data in very high-dimensional settings. Since their introduction, a large body of work has been conducted to improve well-known issues regarding their training instability (Arjovsky, Chintala, & Bottou, 2017; Gulrajani, Ahmed, Arjovsky, Dumoulin, & Courville, 2017; Mescheder, Nowozin, & Geiger, 2017; Mescheder, Geiger, & Nowozin, 2018; C.-L. Li, Chang, Cheng, Yang, & Póczos, 2017). Most commonly known are variations of the Wasserstein GAN (Arjovsky et al., 2017; Gulrajani et al., 2017), which use the Wasserstein distance approximation via the Kantorovich-Rubinstein duality to measure the distance between the data and the generated distribution. (C.-L. Li et al., 2017) develop a GAN in which the discriminator learns an optimal kernel to learn the Mean-maximum discrepancy (MMD) distance. (C.-L. Li et al., 2017) argue this corresponds to matching all moments instead of just the first moment when using the Wasserstein GAN. (Mescheder et al., 2017) propose analyze the training algorithm of GANs and suggest a new training method "consensus optimization". Different GANs are analyzed with respect to their local convergence in (Mescheder et al., 2018) finding. Importantly, (Mirza & Osindero, 2014) first introduces conditional GANs in order to condition generated samples on variables (e.g. one-hot encoding of digits to be generated). This offers the possibility to condition samples which may be important for certain use cases as well as potentially combats mode collapse.

The ability of generative networks to learn complex distributions and generate samples from them has drawn significant attention from academics and practitioners in the financial area. GANs have been applied in 1-dimensional settings amongst others in (Wiese, Knobloch, Korn, & Kretschmer, 2020; Da Silva & Shi, 2019; Takahashi, Chen, & Tanaka-Ishii, 2019; K. E. Smith & Smith, 2020; Ni, Szpruch, Wiese, Liao, & Xiao, 2020). In particular, (Wiese et al., 2020) design a generator architecture including temporal convolutional neural networks outperforming conventional GARCH models. In the multidimensional setting, (Wiese, Bai, Wood, & Buehler, 2019) deploy GANs to generate time series discrete local volatility surfaces outperforming classical generators. (Wiese et al., 2021) tackle the problem of multi-asset market simulation for both spot and option prices. (Wiese & Murray, 2022) present a framework in order to simulate risk-neutral markets, leveraging results from (Buehler, Murray, Pakkanen,

& Wood, 2021). (Limmer & Horvath, 2023) build a novel GAN-like structure in order to "automate robustification in our hedging objective". (Vuletić, Prenzel, & Cucuringu, 2023) use conditional GANs in order to obtain probabilistic forecasts of financial time series incorporating economic terms into the GAN loss function. Apart from GANs, Variational Autoencoders (VAEs) have also been deployed to build market generators. (Lin, Liu, & Huang, 2022) leverage GANs in order to predict Credit Default Swaps (CDS). (Zhou, Pan, Hu, Tang, & Zhao, 2018) deploy them to predict stock price with high-frequency data.(Buehler, Horvath, Lyons, Perez Arribas, & Wood, 2020) build a VAE simulator for time series using signatures. Using rough path theory, the study learns to generate the next step in the signature space and uses inverse transform to obtain the next point of the financial time series. A comprehensive overview of GAN applications in finance is given in (Eckerli & Osterrieder, 2021).

Hybrid approaches under the use of GANs are presented in (Marti, 2020; Prenzel et al., 2022). (Marti, 2020) learn to generate realistic correlation matrices with GANs that can be used in downstream tasks. (Prenzel et al., 2022) generate synthetic calibration of order flow models introducing new dynamics into conventional, analytically tractable models. With respect to direct modeling of LOBs, (J. Li et al., 2020; Coletta et al., 2022) deploy a conditional WGAN to generate the event stream in an order book, conditioned on the previous order stream. Each order is a tuple of price, quantity, type of the order, time since past order, ± 1 side of the book. However, GANs are used in these studies to output discrete values (e.g. the type of the order) while GANs are generally designed to learn continuous probability distributions.

This work aims to model the net transition of the order book state using GANs. More precisely, we design and train a probabilistic model to learn the distribution of future LOB states given some state as condition. To the best of our knowledge, this is the first study to achieve this and to design a model with realistic market impact patterns.

4.3 Problem setting

This section formulates the problem we are dealing with. Many market participants without access to more detailed data sets see the LOB in forms of queue sizes which build the cumulative demand and supply for a certain price at a certain time (see Chapter 2 for more information on different views of the LOB). Moreover, many market participants act rather on order book snapshots than the detailed event stream.

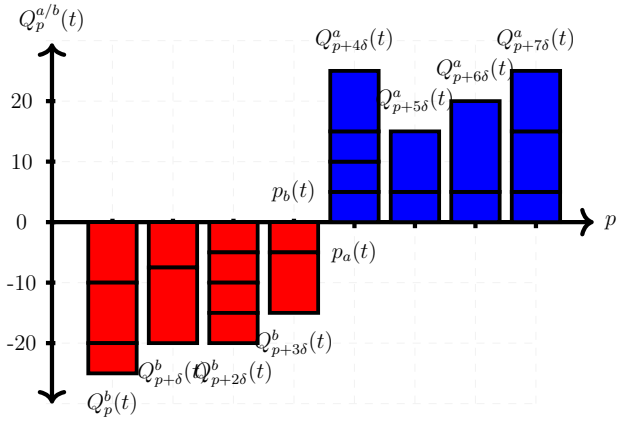


Figure 4.1: Arbitrary LOB snapshot indicating the queues resulting from several orders. Buy queues (red) are negative, sell queues (blue) are positive.

4.3.1 Limit order book snapshots

Hence, we focus on modeling the public view of the order book as introduced in Chapter 2, i.e. Definition 2.1.3

$$Q_p(t) = \sum_{\mathcal{L}_p(t)} q_x \tag{4.1}$$

for some point in time t. This denotes the cumulative sum of all outstanding shares at price p.

We reemphasize that as with the signs of single orders (Definition 2.1.2) a positive queue size indicates a sell (ask) queue and a negative queue size indicates a buy (bid) queue. The best bid and ask price correspond to

$$p_a(t) = \inf_p \{p > 0, Q_p(t) > 0\} \tag{4.2}$$

and

$$p_b(t) = \sup_p \{p > 0, Q_p(t) < 0\}. \tag{4.3}$$

The so-called mid-price is then just the average between bid and ask price, $p_{mid}(t) = \frac{p_a(t)+p_b(t)}{2}$. For no arbitrage reasons, $p_a(t) > p_b(t)$ holds at any time, as well as $Q_p(t) \geq 0 \; \forall \; p > p_a(t)$, and $Q_p(t) \leq 0, \; \forall \; p < p_b(t)$.

A limit order book snapshot is then the collection of queue sizes, usually for $k \in \mathbb{N}$ levels around the mid price. An exemplary centered LOB snapshot is depicted in Figure 4.1.

Definition 4.3.1 (Centered LOB snapshot)**.** For some price grid of length $2k$, i.e. $(p, p+\delta, \ldots, p+(2k-1)\cdot\delta)$, the centered order book at time t is defined as

$$X_t = \left(\underbrace{Q_p(t), \ldots, Q_{p+(k-1)\cdot\delta}(t)}_{\leq 0 \rightarrow \text{ bid}}, \underbrace{Q_{p+k\cdot\delta}(t), \ldots, Q_{p+(k-1)\cdot\delta}(t)}_{\geq 0 \rightarrow \text{ ask}} \right) \tag{4.4}$$

$X_t \in \mathbb{Z}^{2k}$. At every t, the price grid is chosen such that $Q_{p+i\delta}(t) \leq 0 \; \forall \; i \in \{0, \ldots, k-1\}$ and $Q_{p+i\delta}(t) \geq 0 \; \forall \; i \in \{k, \ldots, 2k-1\}$, hence "centered" LOB snapshot.

In this representation, $Q_{p+(k-1)\delta}(t)$ and $Q_{p+(k)\delta}(t)$ correspond to the best bid (ask) queue at the corresponding best bid (ask) price

$$p_b(t) = p + (k-1)\delta(t), \quad \text{and} \quad p_a(t) = p + k\delta(t).$$

The above holds under the assumption that queues are not empty, i.e. $Q_p(t) \neq 0$.

4.3.2 Conditional order book density

Market participants' actions lead to (1) submission, (2) cancellation, and (3) execution of limit orders in the LOB, which are discrete events in continuous time. Each of these actions leads to one change in the book. Its cumulative change over some appropriate time span Δt may be seen as a continuous transition of the different queue sizes from state X_t to $X_{t+\Delta t}$. In the present work, we aim to model the distribution of the centered order book state at time $t + \Delta t$. The future LOB snapshot for the same fixed price grid $(p, p+\delta, \ldots, p+(2k-1)\cdot\delta)$ takes one of the following three forms.

1. No price change: The sign of the queue at price p, $Q_p(t)$, remains the same if $|Q_p(t+\Delta t) - Q_p(t)| < |Q_p(t)|$, i.e. the change is smaller than the queue size. If there is no price change, $X_{t+\Delta t}$ has the same expression as in Equation (4.4), but at time $t + \Delta t$. The future state has the following form

$$X_{t+\Delta t} = \left(\underbrace{Q_p(t+\Delta t), \ldots, Q_{p+(k-1)\cdot\delta}(t+\Delta t)}_{\leq 0 \rightarrow \text{ bid}}, \underbrace{Q_{p+k\cdot\delta}(t+\Delta t), \ldots, Q_{p+(k-1)\cdot\delta}(t+\Delta t)}_{\geq 0 \rightarrow \text{ ask}} \right). \tag{4.5}$$

2. Price decrease: For a price decrease, one or more bid queues have to turn positive, thereby turning into ask queues. In case of Equation (4.4), at least

$Q_{p+(k-1)\cdot\delta}(t+\Delta t)$ would be positive, turning it into an ask quantity. A price decrease of one tick (δ) leads to the following shape of the LOB snapshot

$$X_{t+\Delta t} = \left(\underbrace{Q_p(t+\Delta t), Q_{p+\delta}(t+\Delta t)}_{\leq 0 \rightarrow \text{ bid}}, \underbrace{Q_{p+2\cdot\delta}(t+\Delta t), Q_{p+3\cdot\delta}(t+\Delta t), Q_{p+4\cdot\delta}(t+\Delta t), Q_{p+5\cdot\delta}(t+\Delta t)}_{\geq 0 \rightarrow \text{ ask}} \right). \tag{4.6}$$

Similarly, a price change of k ticks would be induced by a sign change of k - former - bid quantities.

3. Price increase: Vice versa, for a price increase, one or more bid quantities have to turn negative, thereby turning into bid queues. E.g., for a one tick price increase, $Q_{p+k\cdot\delta}(t+\Delta t)$ has to turn negative and for $k=3$ the order book state would read

$$X_{t+\Delta t} = \left(\underbrace{Q_p(t+\Delta t), Q_{p+\delta}(t+\Delta t), Q_{p+2\cdot\delta}(t+\Delta t), Q_{p+3\cdot\delta}(t+\Delta t)}_{\leq 0 \rightarrow \text{ bid}}, \underbrace{Q_{p+4\cdot\delta}(t+\Delta t), Q_{p+5\cdot\delta}(t+\Delta t)}_{\geq 0 \rightarrow \text{ ask}} \right). \tag{4.7}$$

The advantage of the LOB representation in Equation (4.4) is that price changes are inherently modeled via the process of the LOB snapshots. Especially for large tick stocks, price changes are discrete and generally a few ticks, for an appropriate Δt. This is because typically only a few queues get depleted and filled by either the same side (no price change) or the other side (price change). Hence, Equation (4.4) is much more natural than, for example, adding the price change as an additional, discrete variable. Furthermore, price changes are by design consistent with the queue size process. At the same time, it is important to consider that at most k price changes during Δt can be modeled. This is important to keep in mind when choosing k and Δt and should be chosen appropriately depending on the particularities of each individual stock. Especially for small tick stocks, one may want to consider binning several levels into a single one.

As mentioned above, we are interested in the distribution of future centered LOB snapshots $X_{t+\Delta t}$, and aim to draw samples from the distribution

$$X_{t+\Delta t} \sim \mathbb{P}_r(x_{t+\Delta t}), \tag{4.8}$$

where subscript r refers to the probability distribution of the data assumed to be the "real" distribution. Clearly, LOB snapshots depend on each other, as certain realizations of $X_{t+\Delta t}$ are more likely than others depending on the current state X_t (and possibly depending on much larger history, as for instance, (J. Sirignano & Cont,

2019) find longer path dependency in LOBs). Modeling the unconditional distribution $\mathbb{P}_r(x_{t+\Delta t})$ is thus not sufficient.

Instead, a more realistic target is to learn the conditional distribution of centered LOB snapshots at $t + \Delta t$,

$$X_{t+\Delta t}|S_t \sim \mathbb{P}_r(x_{t+\Delta t}|s_t) \tag{4.9}$$

where $S_t \in \mathbb{R}^s$ is some state which serves as a condition for the order book sample.

Such a condition S_t can take many forms, but primarily should contain

1. Recent history in the form of previous lagged order book states

   $$X_t, X_{t-\Delta t}, \ldots, X_{t-h\cdot\Delta t};$$

   such states may be compressed with a suitable encoding function

   $$f(X_t, X_{t-\Delta t}, \ldots, X_{t-h\cdot\Delta t}),$$

   or form a raw condition.

2. "Global" conditions such as time of day, value of the volatility index VIX, etc.

Note that in a Markovian setting, S_t may just be X_t or a compression of X_t.

4.4 Generative adversarial networks

Sampling from the distribution in Equation (4.9) is not straightforward because the distribution is most likely not tractable and entails complex dependencies. Generative adversarial networks may be used as a data-driven method to learn to draw samples from a generated distribution $\mathbb{P}_g(x_{t+\Delta t}|s_t)$ which is similar to the data distribution $\mathbb{P}_r(x_{t+\Delta t}|s_t)$.

In generative modeling, one is confronted with "real" (denoted with subscript r) data X following a probability measure $\mathbb{P}_r$, i.e., $X \sim \mathbb{P}_r$ and $X \in \mathbb{R}^d$, where d denotes the dimension of the input data. The aim is to generate samples $\tilde{X} \in \mathbb{P}_g$, where $\mathbb{P}_g$ is the induced distribution of some generative model such that $D(\mathbb{P}_r, \mathbb{P}_g) \to 0$ for some distance or divergence measure D. In many applications, such as image or speech generation, this task is fairly difficult due to the high dimensionality and complexity of $\mathbb{P}_r$. In the setting of time series, this particularly applies to the dependencies on the past and other conditions.

4.4.1 Entropy GAN

Generative Adversarial Networks (GANs) – first introduced by (Goodfellow et al., 2014) – opened an entirely new area of research in generative modeling. GANs allow to draw samples from an unknown probability distribution which is learned in a data-driven fashion.

In particular, two agents are competing with each other in a two-player min-max game.

1. **Generator:** The generator is given a random noise seed and maps it into the sample space. The generated samples should be real enough so that the discriminator cannot distinguish between real and fake samples.

2. **Discriminator:** The discriminator is tasked with differentiating between real and generated/fake data samples by classifying them accordingly.

The generator model can be defined as follows:

Definition 4.4.1 (Generator (g))**.** Let $Z \sim \mathbb{P}_z, Z \in \mathbb{R}^z$ be a sample drawn from some prior/seed and $\Theta^{(g)}$ the parameter space of g (usually a set of trainable weights of a neural network). The generator g is then a function parameterized by $\theta^{(g)} \in \Theta^{(g)}$, transforming the noise seed Z from the latent space $\mathbb{R}^z$ into the sample space $\mathbb{R}^d$

$$\begin{aligned} g : \Theta^{(g)} \times \mathbb{R}^z &\to \mathbb{R}^d, \\ (\theta^{(g)}, Z) &\mapsto g_{\theta^{(g)}}(Z). \end{aligned} \tag{4.10}$$

Similarly, the discriminator in its classical sense is defined as:

Definition 4.4.2 (Discriminator (d))**.** Let $X \in \mathbb{R}^d$ either $\sim \mathbb{P}_{data}(X)$ or $\sim \mathbb{P}_g(X)$ and $\Theta^{(d)}$ parameter space of d (e.g. trainable weights of neural network).
The discriminator d is a function parameterized by $\theta^{(d)} \in \Theta^{(d)}$ discriminating between real and generated samples:

$$\begin{aligned} d : \Theta^{(d)} \times \mathbb{R}^d &\to [0, 1] \\ (\theta^{(d)}, X) &\mapsto d_{\theta^{(d)}}(X). \end{aligned} \tag{4.11}$$

Both generator (g) and discriminator (d) are generally neural networks with their corresponding parameters $\theta_{(g)}$ and $\theta_{(d)}$. The models are then trained via simultaneous optimization (alternating gradient decent) in a joint min-max game

$$\min_{\theta_{(g)}} \max_{\theta_{(d)}} \mathbb{E}_{x \sim \mathbb{P}_r(x)}[\log d_{\theta^{(d)}}(x)] + \mathbb{E}_{z \sim \mathbb{P}_z(z)}[\log 1 - d_{\theta^{(d)}}(g_{\theta^{(g)}}(z))] \tag{4.12}$$

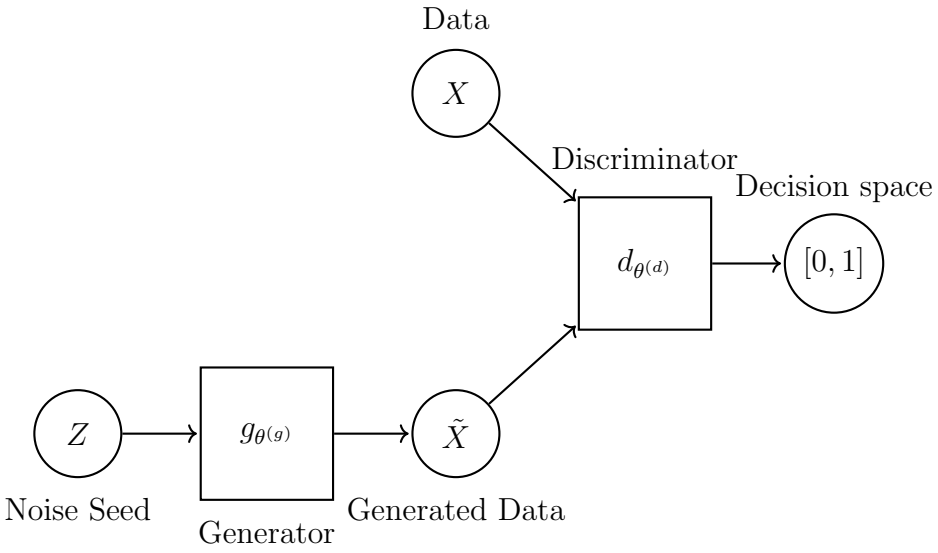


Figure 4.2: GAN scheme

which is the classification accuracy of the discriminator d. The discriminator wants to maximize the expected log-probability of correctly identifying real samples over the real distribution $\mathbb{P}_r$, i.e.$\mathbb{E}_{x\sim\mathbb{P}_r(x)}[\log d_{\theta^{(d)}}(x)]$, and the expectation of the log-probability of correctly identifying fake samples over the prior distribution $\mathbb{P}_z$, i.e. $\mathbb{E}_{z\sim\mathbb{P}_z(z)}[\log 1 - d_{\theta^{(d)}}(g_{\theta^{(g)}}(z))]$. In other words, the discriminator maximizes with respect to $\theta_{(d)}$ to better distinguish between $X \sim \mathbb{P}_r$ and $\tilde{X} \sim \mathbb{P}_g$. This can also be viewed as minimizing the binary cross-entropy of a classification task. On the other side, the generator aims to minimize the ability of the discriminator to correctly classify fake samples with respect to its parameters $\theta_{(g)}$.

Figure 4.2 shows the workflow of a classic GAN. The two squares are the generator and discriminator. A noise seed Z is fed into the generator, which outputs some synthetic data $\tilde{X}$. The discriminator then gets both real samples $X \sim \mathbb{P}_r$ and generated samples $\tilde{X} \sim \mathbb{P}_g$ and intents to classify the samples as real or generated ones. During training, d is trained to improve its classification accuracy. The error from the discriminator is passed through to the generator, however, then tries to minimize the accuracy instead of maximizing it to improve the sample quality and thereby fool the discriminator.

(Goodfellow et al., 2014) show that a discriminator for any state of the generator g is optimal if

$$d_g^* = \frac{p_r(x)}{p_r(x) + p_g(x)} \tag{4.13}$$

which is $\frac{1}{2}$ if $p_r = p_g$.

By simultaneous training of both d and g, d becomes better and better in discriminating between real and fake samples which is then used by g to further and further improve its sample quality. In its optimum which is a saddle point on the loss surface, a Nash equilibrium is reached. This means that given the parameter of the opponent, the present strategy (i.e. $\theta_{(\star)}$) is optimal and the player is not able to improve its strategy. This point is reached when $d(x) = 0.5 \; \forall \; x \sim \mathbb{P}_r, \; x \sim \mathbb{P}_g$. In this case, the d is not able to improve its classification accuracy given the sample quality of the g. Vice versa, the g can't improve its sample quality given the classification accuracy of the d.

It can be shown, that the original GAN setting introduced in (Goodfellow et al., 2014) minimizes the Jensen-Shannon (JS) divergence (Arjovsky et al., 2017), which is an adjusted Kulback-Leibler (KL) divergence to obtain an actual distance satisfying the usual properties.

Definition 4.4.3 (Jensen-Shannon Divergence (JS Divergence))**.** For two probability distributions $\mathbb{P}_r$ and $\mathbb{P}_g$, the Kulback-Leibler divergence reads

$$D_{KL}(\mathbb{P}_r \parallel \mathbb{P}_g) = \int_x \mathbb{P}_r(x) \log \frac{\mathbb{P}_r(x)}{\mathbb{P}_g(x)} dx. \tag{4.14}$$

The symmetric version of the KL divergence, the Jensen-Shannon divergence is then defined by

$$D_{JS}\left(\mathbb{P}_r \| \mathbb{P}_g\right) = \frac{1}{2} D_{KL}\left(\mathbb{P}_r \middle\| \frac{\mathbb{P}_r + \mathbb{P}_g}{2}\right) + \frac{1}{2} D_{KL}\left(\mathbb{P}_g \middle\| \frac{\mathbb{P}_r + \mathbb{P}_g}{2}\right). \tag{4.15}$$

For the optimal discriminator, $d_g^* = \frac{1}{2}$ for any real or fake input, hence (4.12) reduces to $\log \frac{1}{2} + \log \frac{1}{2} = -\log 4$. Using this result and (4.12), the generator cost $C(g) = V(d_g^*, g)$ reads

$$C(g) = -\log(4) + D_{KL}\left(\mathbb{P}_r \middle\| \frac{\mathbb{P}_r + p_g}{2}\right) + D_{KL}\left(\mathbb{P}_g \middle\| \frac{\mathbb{P}_r + \mathbb{P}_g}{2}\right). \tag{4.16}$$

A major drawback of the JS divergence is its discontinuity. I.e., for some sequence $\theta_t \to \theta$, where θ denotes the true parameter setting, the sequence of probability measures $(\mathbb{P}_{\theta_t})_{t \in \mathbb{N}} \not\to \mathbb{P}_\theta$ under the use of JS-divergence (Arjovsky et al., 2017). Due to this discontinuity, gradients for the discriminator can't be used easily in order to make $\mathbb{P}_g$ converging to $\mathbb{P}_r$ which makes training more difficult in many settings. This is in particular the case when two distributions are disjoint.

4.4.2 Wasserstein GAN

To combat the problems that come with training via the JS distance, (Arjovsky et al., 2017) and (Gulrajani et al., 2017) first make use of the Wasserstein-1 which originates from the theory of optimal transport in order to obtain better gradient properties during the training process.

The Wasserstein distance is defined as follows.

Definition 4.4.4 (Wasserstein-1 (W1) Distance)**.** For two probability measures $\mathbb{P}_r$ and $\mathbb{P}_g$, the Wasserstein-1 distance is defined as

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma} \left[\|x - y\|\right] \tag{4.17}$$

where $\Pi(\mathbb{P}_r, \mathbb{P}_g)$ denotes the set of joint probability measures with $\mathbb{P}_r$ and $\mathbb{P}_g$ as their marginal distributions. The W1 distance is also called earth mover distance and can be interpreted as the optimal map to transform one probability measure, e.g. $\mathbb{P}_r$, to another, e.g. $\mathbb{P}_g$.

Despite being expensive to numerically compute, the W1 distance has better properties in terms of continuity than the JS-divergence. More precisely (Arjovsky et al., 2017) show that under certain conditions $W(\mathbb{P}_r, \mathbb{P}_g)$

1. continuous
2. differentiable almost everywhere

which makes the W1 distance a more sensible distance for the GAN framework.

To deal with the intractability of the infimum in (4.17), (Arjovsky et al., 2017) use the Kantorovich-Rubinstein Duality to approximate the dual problem

$$W(\mathbb{P}_r, \mathbb{P}_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim\mathbb{P}_r}[f(x)] - \mathbb{E}_{x\sim\mathbb{P}_g}[f(x)] \tag{4.18}$$

where f needs to be 1-Lipschitz for the duality to hold.

The problem now entails the challenge of maximizing over all 1-Lipschitz functions. In the Wasserstein GAN setup, this is done via a neural network (often dubbed as the *critic*) approximating all 1-Lipschitz functions in order to estimate the dual of the Wasserstein-1 distance from Equation (4.18).

Definition 4.4.5 (Critic (f))**.** Let $X \in \mathbb{R}^d$ be either $\sim \mathbb{P}_r$ or $\sim \mathbb{P}_g$, and $\Theta^{(f)}$ the parameter space of f (e.g. trainable weights of a neural network). The critic f is a function parameterized by $\theta^{(f)} \in \Theta^{(f)}$, such that

$$\begin{aligned} f : \Theta^{(f)} \times \mathbb{R}^d &\to \mathbb{R}, \\ (\theta^{(f)}, X) &\mapsto f_{\theta^{(f)}}(X). \end{aligned} \tag{4.19}$$

This model is replacing the discriminator from Definition 4.4.2 and the new Wasserstein GAN objective function using (4.17) now reads

$$\min_{\theta_{(g)}} \max_{\|f\|_L \leq 1} \mathbb{E}_{x\sim\mathbb{P}_r}[f_{\theta^{(f)}}(x)] - \mathbb{E}_{z\sim\mathbb{P}_z}[f_{\theta^{(f)}}(g_{\theta^{(g)}}(z))]. \tag{4.20}$$

The new min-max game may be understood as a constant aim from the critic f to correctly estimate the W1-distance via maximizing Equation (4.20) while the generator uses the gradient of the approximated W1-distance in order to improve its sample quality and thereby minimizing that distance. Both f and g, as before are trained via simultaneous optimization.

One challenge that arises is the 1-Lipschitz condition of the dual problem from Equation (4.20), which must hold for f for the correct estimation of the Wasserstein-1 distance. To this end, (Arjovsky et al., 2017) suggest the rather rigorous approach to clip the weights of the critic, i.e. restrict $\Theta^{(f)} = [-c, c]^l$, with l denoting the number of trainable weights of the critic, and $c \in \mathbb{R}$ denoting a clipping threshold often chosen relatively small (0.01 in (Arjovsky et al., 2017)). This approach, however, is very constraining in many cases which potentially leads to a critic that is too restrictive. This renders the WGAN approach to fail in many simple examples. To mitigate this issue, (Gulrajani et al., 2017) suggest a gradient penalty which penalizes gradients above 1 for linear combinations between arbitrary points drawn from $\mathbb{P}_r$ and $\mathbb{P}_g$. The updated critic's objective is adjusted with a gradient penalty and reads

$$\min_{\theta^{(g)}} \max_{\theta^{(f)}} \mathbb{E}_{x\sim\mathbb{P}_r}[f_{\theta^{(f)}}(x)] - \mathbb{E}_{z\sim\mathbb{P}_z}[f_{\theta^{(f)}}(g_{\theta^{(g)}}(z))] - \lambda\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[(\|\nabla_{\hat{x}} f_{\theta^{(f)}}(\hat{x})\|_2 - 1)^2\right], \tag{4.21}$$

with the penalty coefficient $\lambda \in \mathbb{R}$ leading to a more efficient enforcement of the Lipschitz condition (Arjovsky et al., 2017). $\mathbb{P}_{\tilde{x}}$ is the distribution of

$$\tilde{x} = ux + (1-u)g(z),$$

where $U \sim U(0,1)$ is a uniformly distributed random variable in the interval $[0,1]$. Note the generator only impacts the second term of Equation (4.21) and the penalty is subtracted since the critic is trained to maximize the equation.

4.4.3 GAN Formulation for LOB simulation

The aim is to learn the probability distribution introduced in (4.9). As outlined above, unconditional sampling will likely not suffice because the future LOB snapshot will

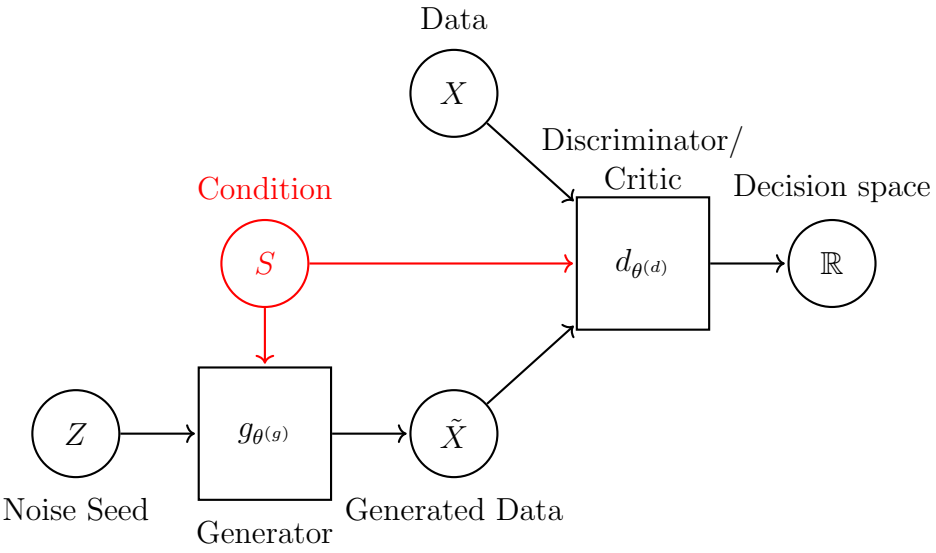


Figure 4.3: Conditional GAN scheme

depend on other variables, such as previous snapshots of the LOB, volatility, and time-of-day effects. We therefore use the setting of conditional GANs first introduced in (Mirza & Osindero, 2014) to condition the samples to be drawn on some state $S_t \in \mathbb{R}^s$. Hence, the generator used including the state S_t has the following form

$$\begin{aligned} g : \Theta^{(g)} \times \mathbb{R}^z \times \mathbb{R}^s &\to \mathbb{R}^d \\ (\theta^{(g)}, Z, S_t) &\mapsto g_{\theta^{(g)}}(Z_t, S_t) = X_{t+\Delta t}. \end{aligned} \tag{4.22}$$

The simplest setting for a model is a Markovian system which will be used as a baseline for future improvement, hence $S_t = X_t$. Similar to the generator, the discriminator (critic) is also given S_t as input to "judge" whether, given a certain condition, a future state is realistic or not. In the case of the entropy GAN, this amounts to

$$\begin{aligned} d : \Theta^{(d)} \times \mathbb{R}^d \times \mathbb{R}^s &\to [0, 1] \\ (\theta^{(d)}, X_{t+\Delta t}, S_t) &\mapsto d_{\theta^{(d)}}(X_{t+\Delta t}, S_t). \end{aligned} \tag{4.23}$$

The updated scheme for a conditional GAN is depicted in Figure 4.3 with the condition shown in red which is fed both into g and into d.

For the present setting, six fully connected layers are used for both g and d (respectively f, in the Wasserstein GAN case). The architecture of the generator is visualized in Figure 4.4, and entails two fully connected layers, each of which separately transforms Z_t and S_t. The transformations ($h_{2,1}$ and $h_{2,2}$) are then concatenated and in two further fully connected layers transformed into the output space.

For g, we use ReLU for hidden layers and linear activation for the output layer because any queue size may be positive (for sell queues) or negative (for buy queues).

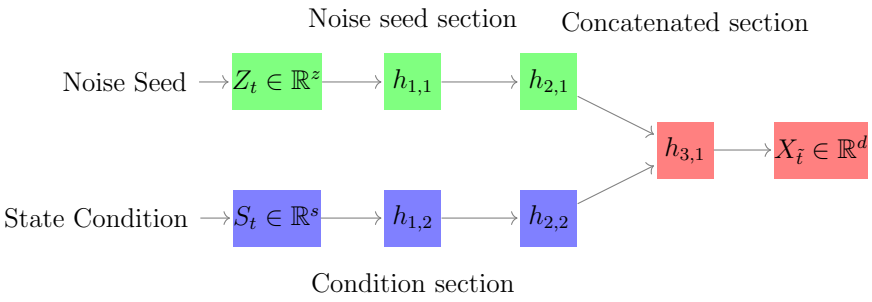


Figure 4.4: Conditional GAN scheme with three hidden layers of which the first and the second ($h_{1,i}$ and $h_{2,i}$) are separated. $h_{2,1}$ and $h_{2,2}$ are then concatenated to combine the network's noise seed and condition section to generate $X_{\tilde{t}}$.

Training is performed using the RMSprop and ADAM optimizer. Further hyper-parameter specifications can be found in Table 4.1. In order to stabilize training, we further apply gradient clipping during the training process to the discriminator to avoid fluctuations particularly due to gradient noise in the later stages of the training.

Activation Hidden Layers	ReLU
Activation Final Layer	Linear
# of hidden layers	3
Hidden Layer Size	32-64
Δt	10 seconds
Days of data	20 days
# Samples	$\sim 40k$
Optimiser	Adam
Learning Rate	$1e-5$
Iterations	$\sim$ 300k

Table 4.1: Table indicating attributes of network and data used for training.

4.5 Empirical results

This section introduces the data used throughout this work, reviews the training process, and presents numerical results of (recurrent) simulation without interaction.

4.5.1 Data set description

The data we used in this research is obtained from lobsterdata.com. The platform provides L3 data with a message file containing the orders sent to an exchange (primarily market, limit, and (partial) cancellation orders), as well as a file containing the corresponding order book states.

To train the neural network, 20 days of MU tick data from 2016 are used. MU is a large tick stock which rarely contains empty levels and mostly has a spread equal to the tick size. This makes it rather easy to learn for our approach in comparison to other small tick names, as the GAN does not have to produce zero-inflated distributions with many ticks mostly close to zero.

Limit order books exhibit non-stationary patterns, particularly at the beginning and at the end of the trading day. We therefore discard the first and last 30 minutes of the continuous trading session, in order to remove market open and closing effects. Setting Δt to 10 seconds results in ∼40k order book transitions for 5.5 hours of trading during 20 days. We choose $k = 3$, in other words, we focus on the first three bid and ask levels. In this case, more than 99.9% of the price changes fall within this range. Furthermore, $4 \cdot \hat{\sigma}_{\Delta t} < 3$ where $\hat{\sigma}_{\Delta t}$ is the volatility of the price changes measured in ticks over 10 seconds (assuming the price behaves like a Brownian motion).

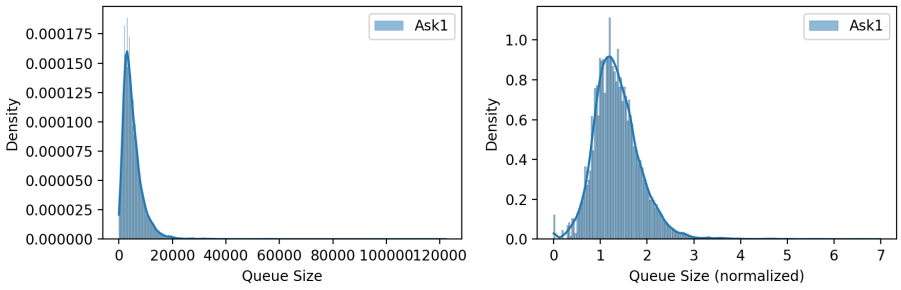


Figure 4.5: Distribution of queue sizes (left) and normalized queue sizes (right).

As for most data-driven tasks, normalization is crucial. While a log transformation seems sensible, problems may occur for queue sizes around or equal to zero. Furthermore, due to the input structure of the state Equation (4.4), the sign of the queue must be preserved. We hence normalize via square root and a normalization constant

$$\tilde{Q}_p(t) = \frac{sign(Q_p(t))\sqrt{|Q_p(t)|}}{c} \tag{4.24}$$

where

$$sign(Q_p(t)) = \begin{cases} 1, & \text{if } Q_p(t) \geq 0 \\ -1, & \text{if } Q_p(t) < 0 \end{cases} \tag{4.25}$$

preserves the correct sign of the queue. Furthermore, c is a constant to scale the data into a certain range, which is particularly important for numerical aspects and if one wants to use data from different names. In Figure 4.5, the normalized distribution is shown on the right-hand side.

4.5.2 Training process

As in every unsupervised learning task, there is no direct metric to observe the quality of the generated samples. In particular, the loss of the entropy GAN does not indicate sample quality, and while the Wasserstein distance in the setting of (Arjovsky et al., 2017; Gulrajani et al., 2017) can indicate the quality over the course of the training, the distance cannot be used to compare different models. Instead, metrics generally found to be important in real data are observed and tracked during the training process to obtain insights into what the model learns as well as how fast and accurately it does so.

For most GAN specifications used in this study, the model learns the metrics quickly and shows some convergence without much tuning. However, some metrics start to fluctuate once they have reached a certain level – especially in the setting of the entropy GAN as introduced first by (Goodfellow et al., 2014). This is particularly critical for the price paths. The shape of the marginal distributions is shown in Figure 4.7a. Small differences in the probability mass can imply different probabilities for price increases or decreases can lead to drifts in the generated price paths. Hence, in particular, when using a balanced training set, the generated distributions should be as symmetric as possible in order to ensure the stability of the synthetic data. Due to the aforementioned fluctuations, this is not a trivial task to handle. The reason for these fluctuations is potentially based on the GAN structure which, in theory, converges to a saddle point (Nash equilibrium). The iterative gradient descent has as a consequence that this saddle point slightly moves around with every step, and hence the GAN never really is quite optimal on a small enough scale, which is enough for these imbalances to manifest. In particular (Mescheder et al., 2017, 2018) analyze these dynamics in more detail. The issue of this symmetry and how to enforce it are addressed in Section 4.5.4.

We find WGAN-GP as presented in (Gulrajani et al., 2017) to work best and lead to the smoothest training process. In particular, the use of gradient clipping of the critic leads to less "extreme" changes at each gradient step. This builds a more consistent base to learn from for the generator. For example, if a certain gradient becomes too large during optimization, clipping prevents the discriminator/critic from moving too far away. Note that we clip by value, not by norm.

Figure 4.6 shows several tracked metrics during the training process. The upper left plot indicates the W1-distances of the marginal distributions which are learned fairly quickly. This also holds for the deviation of the mean and the volatility from the generated samples. In comparison, these lines were much more volatile for the

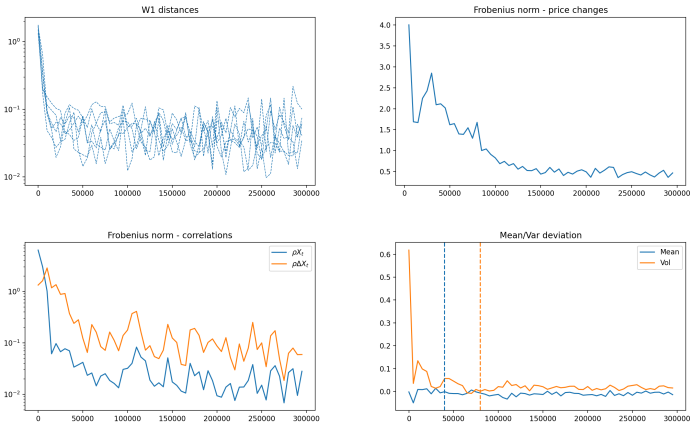


Figure 4.6: Convergence WGAN-GP with Gradient Clipping.

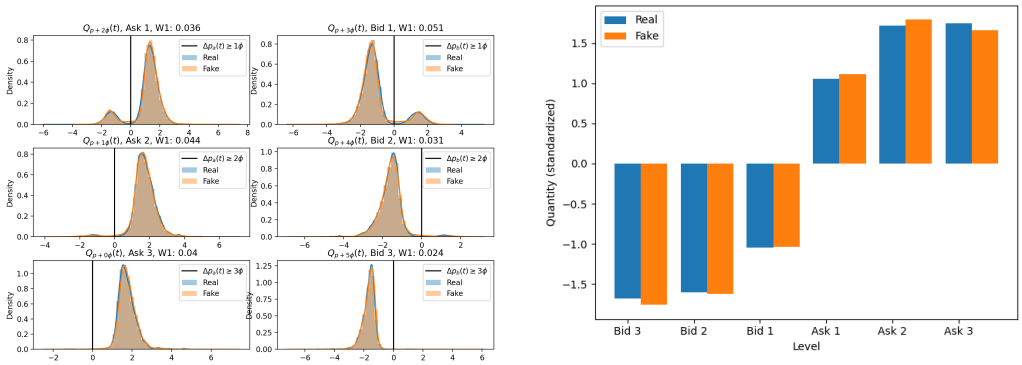
entropy GAN version and without the use of gradient clipping. Perhaps most interesting to observe are the conditional price changes, which are only really learned by the GAN over time. This indicates that the GAN first matches the marginal distributions without much effort, and then needs more time to also capture the conditional properties of the LOB snapshots. The next section analyses each of the metrics in more detail.

4.5.3 Stylized facts

1. **Marginal distributions:** Figure 4.7a indicates the marginal distributions of $X_{t+\Delta t}$. The shape of the distributions is learned well. For Ask-1 (Bid-1), the negative (positive) values correspond to the samples with price changes, i.e. the queue is depleted and becomes a queue of the opposite sign. The mass of the generated samples is not as large as for the real samples, which indicates fewer generated price changes. The small bumps in Ask2, Bid2 depicted in Figure 4.7a correspond to price changes of two ticks, which show less probability mass. This indicates that price changes of two ticks (or more) are much less likely to occur.

2. **Average centered order book shape**: Figure 4.7b shows the average LOB shape for the real and the generated data, which coincide very well. Looking closer, the average queue sizes tend to be slightly larger for ask quantities.

3. **Correlation structure:** The correlation matrices of both the generated order quantities themselves, as well as its changes (e.g. $Q_p(\tilde{t}) - Q_p(t)$), are shown in Figure 4.8. In particular, the correlation of the actual order book values are



(a) Marginal distributions $X_{t+\Delta t}$. (b) Average LOB shape.

Figure 4.7: Marginal distributions and average order book shape for both real and fake data.

very close to those observed in real data. Also, the first differences show a large degree of accordance.

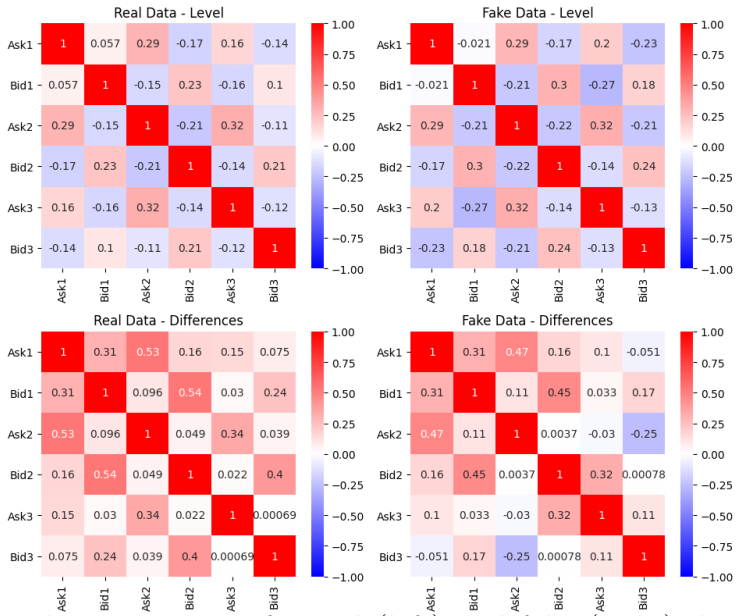
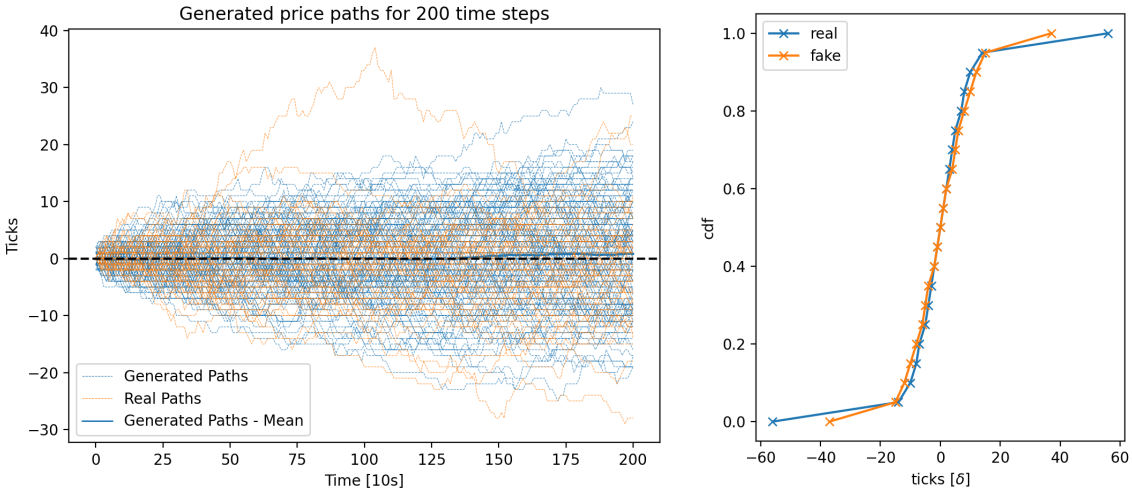


Figure 4.8: Correlation matrices for real (left) and fake (right) data. The top row pertains to the generated order quantities, while the bottom row to the increments $Q_p(\tilde{t}) - Q_p(t)$.

4. **Price paths:** We now turn our attention to properties of the price paths using g in recurrent fashion to generate entire time series. In particular, the generated state $X_{t+\Delta t}$ builds the new condition, $S_{t+\Delta t}$, which is then used to generate $X_{t+2\Delta t}$. After a sign change – i.e. price change – the state is re-centered and the shift is stored as the corresponding price change, as explained in Section 4.3.

Figure 4.9 shows multiple real and generated paths, as well as the mean of the generated paths. The initial order book states X_0 are sampled from the data set. It is also possible to use the same initial order book state X_0. Table 4.2 shows statistics regarding the price changes. The values for real and fake price paths coincide well. For both real and fake data, the drift is close to zero. Some of the mean's deviation is to sample variance. We also observe a slightly higher volatility for generated price paths. One reason for that is the slightly higher percentage of transitions with more than one tick. While the real distribution 200-step returns show significant excess kurtosis, the generated ones do not, indicating the GAN is not producing price paths with fat tails. This is expected as the model has a Markovian structure when setting $S_t = X_t$



(a) Real and fake price paths (in ticks).

(b) CDF of $\Delta p_{mid}(200)$ in ticks.

Figure 4.9: Price paths and corresponding percentiles of terminal prices for real and fake data.

Figure 4.9b supports previous statements. The center percentiles coincide very well between real and generated price changes over 200 transitions. However, especially the lowest and highest points in the figure – corresponding to the 0% and 100% quantile and hence indicating the minimum and maximum – show significant deviation further. This indicates that the paths generated by the GAN do not show the same tail properties as the real data. Despite the 0% and 100% quantile being very noisy values, this observation is very consistent.

Besides the differences in the tail behavior of the generated paths, the distribution of price changes over 200 transitions is fit quite well. The quantiles in

Figure 4.9b – particularly from 5% to 95% – show very little deviation apart from the mentioned tail behavior. Furthermore, it is to emphasize the price paths show a surprisingly good fit taking into account of being purely generated by transitions of the order book's queue size process X_t. This indicates proper modeling of the process of queue size transitions can produce realistic price paths.

	μ	σ	0δ	1δ	-1δ	2δ	-2δ	3δ	-3δ
$\mathbb{P}_r$	-0.24	9.05	0.7373	0.1125	0.1125	0.0156	0.0158	0.0019	0.0020
$\mathbb{P}_g$	-0.55	9.19	0.7214	0.1145	0.1145	0.0194	0.0152	0.0021	0.0027

Table 4.2: Table summarizing price moves. The first two columns indicate 200-step mean and volatility, with the remainder columns showing the frequency of particular price changes.

5. **Conditional price changes:** In the literature, the order imbalance or book imbalance, defined as

$$OI(t) = \frac{Q_{p_b}(t) - Q_{p_a}(t)}{Q_{p_b}(t) + Q_{p_a}(t)}, \tag{4.26}$$

is well known to have predictive power for future price changes. In particular, the higher the imbalance between the bid and the ask side of the order book, the more likely is a price change toward the direction of the smaller queue. This can be explained economically by the idea that there is a weaker support for the price on the side of the smaller queue. It thus stands to question, whether the model's probability for price changes differs depending on the state and furthermore, whether this coincides with what is observed in real data. One quantity to look at is the probability of a positive change of the mid price $p_m(t)$, e.g.

$$\mathbb{P}(p_m(\tilde{t}) > p_m(t) \,|p_m(\tilde{t}) \neq p_m(t), Q_1^a(t) \in [q_k, q_{k+1}], Q_1^b(t) \in [q_l, q_{l+1}]), \tag{4.27}$$

where $k, l \in \{0, \ldots, 9\}$ indicate the number of the quantile corresponding to 0% to 90% quantile. Figure 4.10 indicates these probabilities for both real and generated data. For instance, the upper left square indicates the probability for a price change to be positive if both bid and ask queues are in the bucket between the corresponding 0% and 10% quantile.

Figure 4.10 shows that the generator, solely learning the process of the queues in the order book, accurately reflects the different probabilities of price changes.

In view of the queue sizes distributions, it verifies whether the probability mass on the negative and positive side in Figure 4.7a – but conditioned on the queue sizes – are correct, as these correspond to the particular price changes. The result indicates that conditioning on history, i.e. at least on the previous state X_t, is necessary and learned well by the generator.

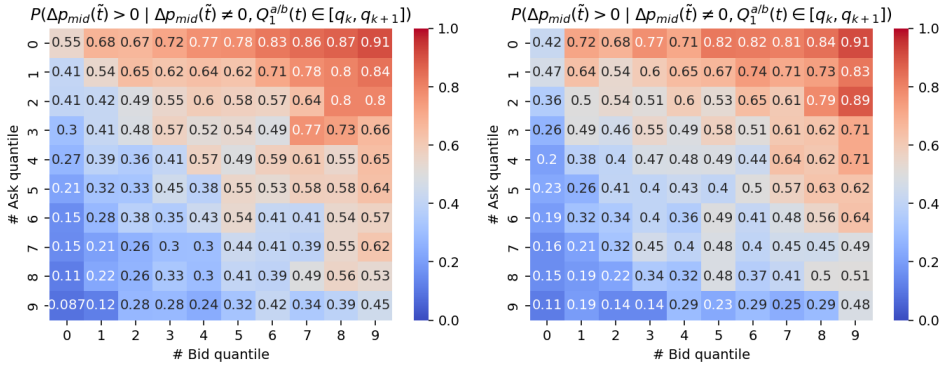


Figure 4.10: Matrix indicating the probability of a positive price change, given there is a price change, as a function of the quantiles of the best bid/best ask queues. Left: Real, right: Fake.

4.5.4 Symmetry of simulations

As previously mentioned, training stability is a notorious problem GANs suffer from (Arjovsky et al., 2017; Gulrajani et al., 2017; Mescheder et al., 2017), which has also been documented in Section 4.5.2. For the present representation, the symmetry of the simulations is important for several reasons. First, the data distributions are close to symmetric, as can be seen in Section 4.5.3. Second, on shorter time scales, it is economically sensible that the mechanics in the LOB are generally symmetrical. Lastly, imbalances between opposite generated distributions (e.g. Ask-1 and Bid-1) may introduce drifts.

In particular, it is problematic if

$$\mathbb{P}_g(Q_{p_b}(t+\Delta t) > 0) \neq \mathbb{P}_g(Q_{p_a}(t+\Delta t) < 0),$$

and more generally if

$$\mathbb{P}_g(Q_{p_b - i\delta}(t+\Delta t) > 0) \neq \mathbb{P}_g(Q_{p_a + i\delta}(t+\Delta t) < 0).$$

In other words, price increases and price decreases of certain ticks are not equally likely (in an unconditional sense). The consequences are drifts which potentially offer arbitrage opportunities. To this end, we may make the following assumption.

Assumption 4.5.1. For any state S_t, the probability distribution should be symmetrical, in the sense that

$$\mathbb{P}_g(Q_{p+i\delta}(t+\Delta t)|S_t) = \mathbb{P}_g(r(Q_{p+(2k-1-i)}(t+\Delta t))|r(S_t)), \tag{4.28}$$

where $r : \mathbb{R} \to \mathbb{R}$ denotes the point reflection around the mid-price.

This mainly states that state dynamics in the LOB are assumed to be symmetric, e.g. $r(Q_{p_b}(t)) = -Q_{p_a}(t)$. Hence, for each step in the simulation, the state S_t is flipped with probability 0.5.

$$X_{t+\delta t} = \begin{cases} g(Z_t, S_t), & \text{with } p = 0.5 \\ r(g(Z_t, r(S_t))), & \text{with } p = 0.5 \end{cases} \tag{4.29}$$

By design, the generated probability distribution is then symmetrical around the mid-price, which ensures that there is no structural drift in the simulations that could be exploited by an agent.

4.5.5 Benchmarks: Poisson and Hawkes order flow

The results in Section 4.5.3 indicate good fits of distributional quantities of the queue sizes, as well as (conditional) price paths. A natural question to consider is how these results compare to traditional methods, such as Poisson or Hawkes processes-driven order flow which model the entire event stream.

To this end, we emphasize that our model generates snapshots of limit order books, and not the entire event stream. Consequently, some granularity is lost in the generative process. However, as mentioned in the introduction, this loss is not of critical importance for many market participants that operate outside of the high-frequency setting, but rather focus on mid-frequency time scales.

Both Poisson and Hawkes models are briefly explained in the appendix. We refer the reader to (Abergel, Anane, Chakraborti, Jedidi, & Toke, 2016) for a detailed explanation of the models and their simulation, in particular chapters 6 and 8 therein. Details on the simulation are given in chapter 9.

The figures in Appendix B.2 show marginals, average order book shape, as well as price paths and conditional price changes of the snapshots taken from the Poisson order flow. Figure B.1a reveals that the queue size distribution of the generated snapshots differs substantially from that of the real distribution shown in blue. Furthermore, the indicated Wasserstein distances are one order of magnitude higher than they are for the proposed GAN-based model. Most importantly, it can be seen that

the probability mass for price changes (i.e. a negative Ask 1 or positive Bid 1 queue respectively) is much smaller in the generated data. This indicates a lower frequency of price changes compared to the real data. Consequently, the average order book shape in Figure B.1b also differs substantially in comparison to the GAN-based model. The same figures for the Hawkes-driven order flow are shown in Appendix B.3.

Figure B.2 shows the price paths of the same number of simulations as in Figure 4.9. A closer look reveals that the generated real paths exhibit significantly lower volatility, about half of the volatility seen in real price paths and those generated by the suggested GAN model. This is a general issue with Poisson order flow, which has a mean-reverting queue because the cancellation intensity decreases with queue sizes, which is not necessarily realistic.

Finally, Figure B.3 shows the equivalent matrix on the right-hand side as the right one in Figure 4.10 but for Poisson order flow. The pattern is somewhat similar to what can be observed in the real data, on the left side. However, the fit is much worse than for the GAN model. Furthermore, the indicated probability is conditioned on that a price change occurs. As mentioned above, the frequency of price changes itself is much lower for the benchmark which is not reflected in this matrix.

4.6 Interaction with the simulator

Throughout the last sections, we presented an approach to use generative adversarial networks to simulate snapshots of LOB for a specified time interval Δt. The results in Section 4.5 demonstrate that the proposed model reproduces unconditional stylized facts very accurately, including price time series.

When it comes to synthetic LOB data/simulators, the ultimate eventual target is to obtain a realistic response of the simulator when the user interacts with it. Most importantly, the so-called *market impact* of the interaction should be realistic. One important study has been conducted by (Gatheral, 2010), which in particular deals with the "square-root law". The market impact, as a function of the quantity normalized by trade volume, takes the shape of a square-root function, defined as

$$\text{Cost} = \text{Spread term} + c\sigma\sqrt{\frac{n}{V}}. \tag{4.30}$$

This interaction generally takes place when either submitting new limit orders, canceling existing limit orders, or executing against existing limit orders via a market order. In this section, we analyze the effect of the

1. execution of a larger parent order of different sizes via market orders
2. liquidity provision via submission of limit orders
3. execution of parent orders via market orders with different slicing.

4.6.1 Market order execution

In this section, the execution of a parent order via several market orders is considered. In particular, we assume

- a parent order size $Q^{parent} \in \mathbb{Z}$
- an execution horizon of $n \in \mathbb{N}$ time steps LOB transitions (i.e. $n\Delta t$ seconds)
- some frequency f.

Q^{parent} is the total order size to be executed over n time steps and frequency f. If $f = 1$, a market order is sent every time step (Δt), and the amount of liquidity $\frac{Q^{parent}}{n}$ is removed from the book. A market order can be expressed as a 3-tuple (t, p, q), with $q > 0$ and $p = 0$ for sell market orders, and $q < 0$ and $p = \infty$ for buy market orders respectively.

Example 4.6.1 (Market order execution)**.** Assume some current order book state X_t with ask side $Q_{p+i\cdot\delta}(t) < 0,\ \forall\ i \in k, \dots, 2k-1$, as described in Definition 4.3.1. A buy market order (t, p, q), with $p = \infty$ and quantity $q < 0$, is then executed instantaneously at time t, which leads to an instantaneous change in the LOB snapshot. For instance, if the market order does not exceed the queue, i.e. $|q| < |Q_{p+(k-1)\cdot\delta}(t)|$, the altered best queue size reads

$$\tilde{Q}_{p+k\cdot\delta}(t) = Q_{p+k\cdot\delta}(t) + q.$$

In case the order is larger than the available liquidity at the best price, the queue gets depleted and the remainder of the order gets executed against available liquidity at the second-best level, a phenomenon commonly referred to as a multi-level sweep event.

Then, the general altered queue on any bid level reads as

$$\tilde{Q}_{p_b-i\delta}(t) = Q_{p_b-i\delta}(t) + \left(q + \sum_{j=0}^{i-1} Q_{p_b-j\delta}(t)\right)_+, \tag{4.31}$$

where $p_b = p+(k-1)\delta$ is the best bid price as outlined in Definition 4.3.1. The second term is the remaining quantity from the parent order which has not been executed by depleting the previous $i-1$ bid levels. The difference is placed into $(\dots)_+$ brackets, as a negative quantity means that the child order was smaller than the cumulative quantity of the previous levels.

Vice versa, the altered queue level on any ask side reads as

$$\tilde{Q}_{p_a+i\delta}(t) = Q_{p_a+i\delta}(t) + \left(q + \sum_{j=0}^{i-1} Q_{p_a+j\delta}(t) \right)_- . \tag{4.32}$$

Also here, the second term is the remaining quantity of the order after depleting the previous $i-1$ ask levels. This time, the $(\dots)_-$ brackets are in the equation as only a negative remainder indicates a partially executed order.

Figure 4.11 shows the result of 10000 executions with increasing Q^{parent}. The parent orders are multiples of quantiles of the distribution of queue sizes. The number of time steps n is set to 60, and $f = \frac{1}{3}$. Each child order is hence of size $\frac{Q^{parent}}{n}\frac{1}{f}$. The mean price paths are shown in Figure 4.11. To improve comparability the paths are created with the same initial order book states S_0 and also the noise $Z_0, \dots, Z_T$ is identical. This aims to provide a more realistic counterfactual scenario.

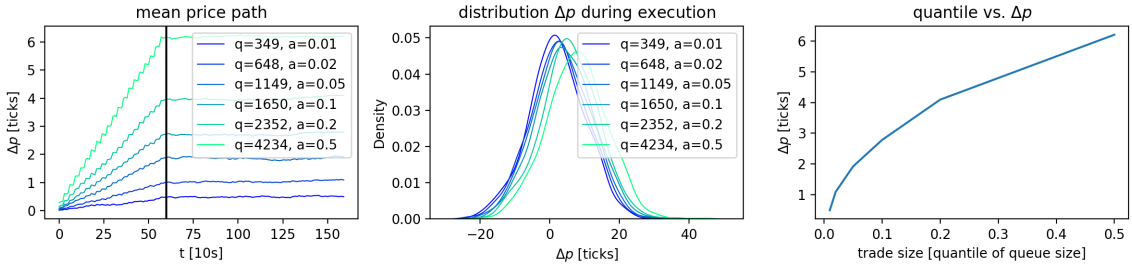


Figure 4.11: Execution of a parent order with different sizes based on the empirical distribution of the queue size.

- The leftmost plot in Figure 4.11 shows the average price paths for increasing buy parent orders with increasing sizes. The execution ends after $60\Delta t$ and a total of $n \cdot f$ market orders. It can clearly be seen that the model entails an amount of market impact that aligns well with several expectations.

  1. Buy orders have a positive price impact, i.e. the mean price path of the execution is positive.

2. The average price impact increases with increasing parent order (child order) size.

3. The impact stops after the execution, which occurs because of two reasons. First, the extra drift induced by sequentially taking liquidity from one side stops. Second, due to the Markovian structure of the model presented here, a persistent market impact would reveal a lack of stability of the simulator.

- The middle plot in Figure 4.11 indicates the distributions of the prices at the end of the execution. The shifts of the distribution are clearly visible.

- The right plot in Figure 4.11 displays the average price impact of the order executions depending on the order size

$$\mathbb{E}_g[\Delta p_{mid} | Q^{target}]. \tag{4.33}$$

  As already mentioned above, price impact increases with increasing queue size. Furthermore, the curve on the right plot in Figure 4.11 appears to be concave, which gives the price impact curve a square-root-like shape, as often found and argued in the literature, especially in the seminal work of (Gatheral, 2010). The model inherently reproduces this pattern without being specifically trained to do so.

Figure 4.12 shows 200 of the 10000 execution paths. While the effect of the execution is not strongly visible for the smallest child order size, the effect quickly becomes much stronger. In particular, the two largest order sizes avoid stronger downward moves from the paths. After the end of the execution (indicated by the red vertical line), the trend disappears and the paths continue without any noticeable drift, as one would expect.

For comparison, Figure B.4 shows the equivalent experiment with Poisson-based event simulation. Overall, an impact can be seen which tends to be increasing with increasing child order size. However, this impact is very small for the smaller child order sizes and not too conclusive. Only for the largest child order size, we observe an impact similar to the one illustrated in Figure 4.11. This behavior is, however, more of a mechanical nature, because depletion of price level is quite likely in that setting. The price impact curve on the right plot has a more convex instead of a concave pattern, which stands in contrast to observations from the market impact literature (Gatheral, 2010).

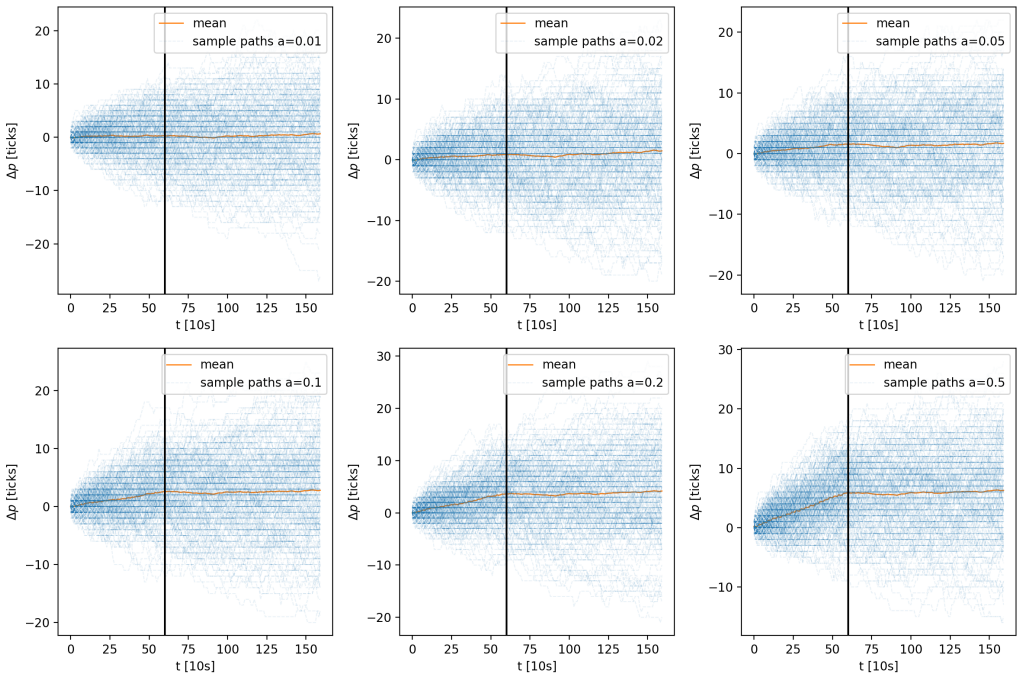


Figure 4.12: Price paths during buy market order execution. The orange horizontal line indicates the end of execution.

Figure B.9 displays the impact patterns with Hawkes-based event simulation. The simulation is performed as described in (Abergel et al., 2016). Similar to our model and the Poisson order flow, an impact is observable with increasing child order size. Moreover, the impact for smaller child order sizes appears more realistic than using Poisson arrival times. However, no decreasing marginal impact can be observed, and the impact of smaller trade sizes becomes less clear in comparison to the results in Figure 4.11. Lastly, as for Poisson order flow, it can be seen that the distribution of terminal prices with execution also shows much less volatility than the real data, in comparison to our proposed GAN model.

4.6.2 Liquidity provision

The previous section provided insights into how the model reacts when interacting with it by "taking liquidity" via instantaneously changing the order book state X_t. Market participants also provide liquidity by sending limit orders. Similar to liquidity extraction, the submission of limit orders has an impact. As before, we consider an order submission schedule where a quantity Q^{parent} is submitted over n time steps ($n \cdot \Delta t$ seconds) using different child orders. This results in $n \cdot f$ limit order submissions

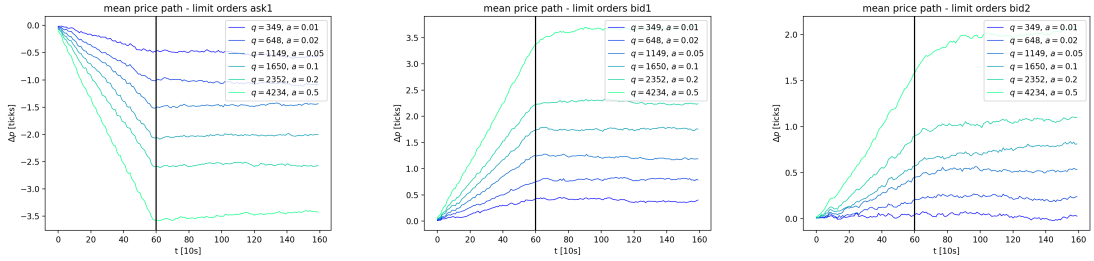
every $\frac{1}{f}$ time steps. The altered queue size reads

$$\tilde{Q}_p(t) = Q_p(t) + q, \tag{4.34}$$

and $q = Q^{parent} \cdot \frac{1}{n \cdot f}$. Note that for limit orders, the queue and the order quantity q have the same sign. Only market orders have a different sign. Hence, a queue never gets depleted here and one submission only affects one queue.

Figure 4.13 shows the mean price paths of 10,000 limit order submissions. For comparability with the experiments from Section 4.6.1, the order sizes are kept the same with $n = 60$ and $f = \frac{1}{3}$, leading to 20 limit order submissions. Sell order submissions at the best ask price $p_a(t)$, i.e. queue $Q_{p+k\cdot\delta}(t)$ are shown in the leftmost plot, and buy order submission at the best bid $p_b(t)$, i.e. queue $Q_{p+(k-1)\cdot\delta}(t)$ in the middle plot. The rightmost plot shows the effect of providing liquidity at the second-best bid price $p_b(t) - \delta$.

Once induced in the order book, we assume the order might be canceled or executed according to the general dynamics learned by the model. As before, we keep on simulating for 100 transitions after the execution. Furthermore, we use the same state initialization S_0 and noise sequence $Z_0, \ldots, Z_T$ as for the market order execution in Section 4.6.1, in order to preserve the counterfactual quality of the scenarios.



(a) First Ask (b) First Bid (c) Second Bid

Figure 4.13: Market impact of liquidity provision on both the ask and the bid side of the LOB. The black vertical line indicates the stop of the trade.

1. Similar to simulated market orders via liquidity extraction, the sequential liquidity provision also exhibits price impact. This price impact has the expected direction; buy limit orders at the bid drive markets up (or avoid price decreases) and vice versa. Limit orders support a particular price level for two main reasons. First, from a technical point of view, more quantity must be executed in order to deplete the particular queue. Second, it indicates demand for a particular price reinforcing the belief that the price of the corresponding asset should

be higher. This makes a price move towards the direction of the limit order less likely.

2. The observed price impact for the same quantity placed into the book is lower than when extracting liquidity. E.g. for a child order size of $q = 4234, Q^{parent} = 84,680$ shares, the price impact for limit order submission is around 3.5 ticks in comparison to roughly 6 ticks for the market order execution of the same size and execution style.

3. The mean price path seems smoother than the mean price path under liquidity extraction in Figure 4.11. One reason for this is that no price changes are directly caused by liquidity provision, in contrast to the extraction of liquidity. Liquidity provision instead has more of an effect of price support avoiding the depletion of the queue.

4. The magnitude of the impact when providing liquidity is the same for both the bid and the ask side. This holds both for the symmetric sampling as well as normal sampling.

5. Similar to before, the impact stops after execution (at time step 60) and the mean price path continues with zero drift. This is expected, in particular considering the Markovian structure of the generator.

6. Lastly, Figure 4.13c shows the effect of placing limit orders at the second best bid level $p_b(t) + \delta$ $(p + (k-2) \cdot \delta)$. Interestingly, the model also learns some price impact of placing liquidity deeper in the book. The price impact of the same order stream is yet smaller than placing limit orders at the best level – in line with our expectations. Figure 4.13c shows a slight continuation for a few steps after the execution. This may be interpreted as a "wall of liquidity", built up during the provision for large child order size, which then has a more persistent impact.

Figure B.5 shows the equivalent results using Poisson order flow. The picture here is very unclear. The addition of limit orders appears to have a rather diffuse impact, which is not quite clear in comparison to the GAN model. Again, the largest order size on the first Ask level tends to have some negative impact. This impact however is very small, and smaller child order sizes do not seem to have any impact at all in comparison. The dynamics when placing at the first bid tend to be the same for all order sizes in this simulation. Limit orders at the second Bid show a very slight

impact. It stands to question whether this is significant or not. Overall, the picture is very noisy and not as clearly structured as Figure 4.13.

Similarly, for the Hawkes process-driven order flow, the patterns of liquidity provision are fairly unclear, as shown in Figure B.10. Similarly to the Poisson order flow, a qualitatively recognizable drift can be seen, in particular in the simulations on the bid side. However, there is no real distinguishable effect between the order sizes in comparison. The overall effect however is much smaller than the one from liquidity provision. Also, the impact does not necessarily appear to be monotone here. We also emphasize that the simulation takes significantly more time than our proposed model, in line with findings from the Hawkes processes literature.

4.6.3 POV execution

The previous two subsections provide evidence that the proposed simulator automatically learns to exhibit some form of market impact, which is in line with economic expectations, findings in the literature, and expectations of industry practitioners. This holds for both liquidity extraction via market orders as well as liquidity provision via limit order submissions.

This section analyzes how the model behaves when the same quantity Q^{parent} is executed over different frequencies and lengths. In particular, executing with different frequencies implies that, over the execution horizon, the percentage of volume (POV) is the same despite the simulator not tracking executed volume. It thus stands to question what difference it has on how Q^{parent} is sliced into larger and smaller child orders, which are then sent during n time steps and at different frequencies.

In particular, we assume a quantity Q^{parent} over $n = 100$ time steps and then execute Q^{parent} via market orders, as outlined in Section 4.6.1 in two ways

1. Execution with even rate, i.e. $f_{even} = 1$: This strategy executes $q = \frac{Q^{parent}}{n}$ at every Δt. The mean price path of the execution is shown in orange in Figure 4.14. The execution schedule is depicted on the right side in Figure 4.14.

2. Slow (quantized step) execution with $f_{quantized}$: Executes larger chunks $q = \frac{Q^{parent}}{n \cdot f}$ every $f_{quantized}^{-1}$ time steps. The mean price path and execution schedule are shown in blue, in the same Figure 4.14.

Figure 4.14 shows the results for both execution styles and $Q^{parent} = 10,000$. The quantized frequency $f_{quantized}$ is set to $1/10$, hence, we execute 10 times as much as during the even execution but only after every 10th transition. Both execution styles

lead to roughly the same price impact, suggesting that POV is the main driver in this model – despite not being explicitly modeled and incorporated into the learning process. This is in line with studies such as (R. Almgren & Chriss, 2001), stating that POV is the primary driver of market impact. The price paths of the quantized execution lead to more price changes during the placement of larger market orders, both by inducing price changes directly or causing large imbalances, making price changes very likely to occur.

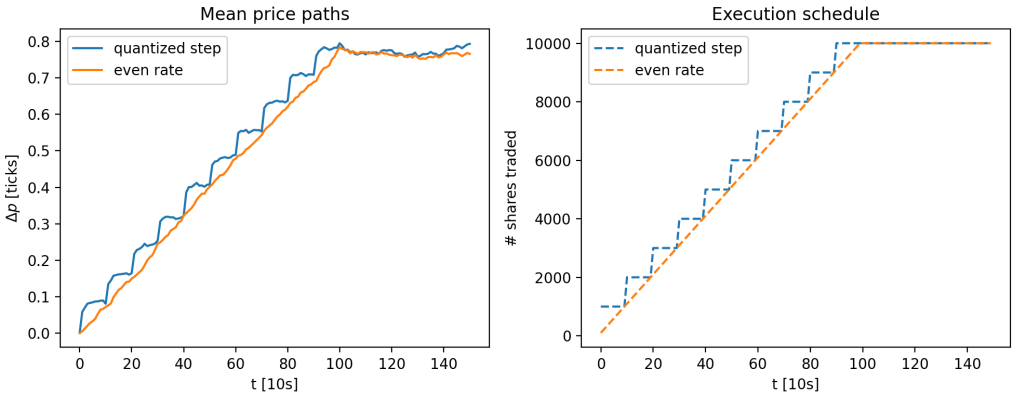


Figure 4.14: Execution of the same quantity Q^{parent} with larger and smaller child orders, leading to the same POV both over the entire time interval.

We remark that by increasing Q^{parent}, decreasing n, or decreasing $f_{quantized}$, child orders may become very large – in particular in the quantized execution setting. For these child orders, which consistently deplete more than the queue, the model appears to have difficulties absorbing these larger orders. The consequence is that quantized execution tends to have a smaller price impact. This shortcoming may be improved by increasing the training data and inducing more samples with several empty levels. However, as the book is mostly populated, frequent acting which leads to level depletion is rather unrealistic.

4.7 Conclusion

Ever since limit order books have been used first to organize electronic exchanges, a vast amount of studies have been conducted. In particular, the construction of realistic models for generating synthetic data is an ongoing challenge and has been attempted to solve with a variety of methods, ranging from stochastic point processes to agent-based models. For many applications, it is sufficient to learn the dynamics of the public LOB view – the LOB snapshots for a certain price grid around the best

price – rather than the entire order-by-order event stream. This in particular holds true for mid to high-frequency trading because many investors base their decisions on the time series of LOB snapshots.

In this work, we consider the modeling of the time series of these LOB snapshots. We seek a generative model that learns the probability distribution of future order book states, conditioned on previous LOB snapshots, aiming to

1. reproduce certain stylized facts, i.e., match distributions, correlations, auto-correlation properties, etc.

2. reproduce market impact when interacting with the model.

The proposed model is based on GANs (Goodfellow et al., 2014) that learn the transitions between LOB snapshots for some time step Δt. The current Markovian system only uses the snapshot at time t in order to learn $\mathbb{P}_r(x_{t+\Delta t}|s_t)$. The framework is proposed in a way that enables the GAN to generate discrete price changes via the process of queue sizes, thereby circumventing the weakness of GANs to learn discrete probability distributions. Moreover, generating LOB snapshots by sampling from a conditional probability distribution directly via GANs is much faster and more efficient than event-based modeling and simulation.

The obtained results show that GANs are able to learn a wide range of dynamics of LOBs in a simple Markovian setting. As analyzed in Section 4.5, most statistical properties such as marginal distributions and correlations are learned very fast. Also, the drift and volatility of the price paths quickly match the properties of the real data. By using gradient penalty and clipping of the critic's gradient in the Wasserstein GAN setting, training can be further stabilized, as otherwise, the generated distribution tends to fluctuate. This accuracy is indispensable, as slight inaccuracies can lead to drifts in the price paths and other issues. To this end, we adjust the sampling method to make the model's samples symmetric.

Apart from matching statistical properties when using the model to generate synthetic data, the GAN may also be used for actual interaction of the user with the simulator. More importantly, the conditional probability distribution is altered by changing the state which corresponds to simulating trades. Among general patterns, we demonstrate that the model

1. shows market impact which is expected from economic intuition,

2. recovers the square-root law with respect to trade size, as commonly known in the literature,

3. reproduces a smaller impact of liquidity provision compared to liquidity extraction,

4. exhibits even less impact of liquidity provision deeper in the book.

We compare the results to two order flow-based models, namely Poisson and Hawkes order flow. The latter is often considered as the benchmark in the field of order book modeling. As mentioned in the introduction, many applications require the modeling of the state of the book, and not of the entire stream. That being enough, our model outperforms in most metrics. Despite losing certain granularity, we show that the quantity process of the queues is modeled much more accurately, which inherently leads to more realistic price paths in comparison to Poisson and Hawkes processes which exhibit much less volatility compared to the real data. While the market impact due to liquidity extraction shows patterns not too different, our model produces a more concave market impact curve. A striking difference is observed in liquidity provision, which does not show a clear picture in the case of the benchmark models, whereas our proposed approach shows a clear impact pattern.

Future research directions. As anticipated in the introduction, this work opens up plenty of directions for future research, some of which are indispensable to be researched in order to use the approach as a realistic backtesting environment and other use cases.

GANs are known for their instability during training. In particular, in the later training stages, when the GAN is close to the solution, the GAN tends to fluctuate around the target. These fluctuations can be enough to cause slight imbalances in the generated distributions, in particular the marginals of the best levels in Figure 4.7a. The end result is that the probability of price increases and decreases are not balanced. This can result in drifts in the generated price paths. It would thus be of great use to further improve the convergence of the model and/or enforce symmetry within the model. Potentially, normalizing flows (Papamakarios, Nalisnick, Rezende, Mohamed, & Lakshminarayanan, 2021) may be used to improve this stability.

The presented results were obtained with a Markovian setting, as outlined in Section 4.4. However, there is path dependency well documented in previous studies, e.g. (J. Sirignano & Cont, 2019). In particular, the impact of trade sequences may be different, as one would expect a decaying market impact when accounting for history. Whether or not the inclusion of more history leads to improved and/or different results remains to be investigated.

This study models the state X_t and how it evolves over time. It does not contain any information on what happens during the transition from t to $t + \Delta t$. In future work, the state may be extended/enlarged such that more information could be modeled, such as the cumulative quantity of order submissions, cancellations, and executions, in order to obtain insights on whether a submitted order has been executed or not. This could then better inform the fill probability modeling, which is of interest to both practitioners and academics.

We mainly consider dense LOBs where $Q_p(t) \neq 0$. This means that there are no empty levels, and in particular, the spread $s(t) = p_a(t) - p_b(t) = \delta$. Currently, the network is designed with a linear activation function in the final output layer. Assuming the stock modeled contains many snapshots with $s(t) > \delta$, the distribution of the queue sizes will be zero-inflated. The GAN would have to learn to map many values directly (or sufficiently close) to zero which is inconvenient when using a linear activation function in the final layer. It would hence be beneficial to adjust the design such that the model supports zero-inflated distribution. This could be achieved by either changing the output structure or using an activation function that supports and promotes the generation of zeros.

Lastly, we have disregarded stationarity issues in our study. In particular, we use 20 subsequent days of a stock regardless of its trading volume, time of the day, volatility, etc. To this end, one could use a volume-based or tick-based clock, as an alternative to the calendar clock we used throughout this paper. Furthermore, increasing the size of data and adding further conditions to account for, e.g. intraday patterns, may improve the data generation process.

Chapter 5

Dynamic Calibration of Limit Order Book Models

5.1 Introduction

Limit order books (LOBs) provide the mechanism via which most stock trading is organized at most exchanges. For many applications, it is not only of interest to simulate the LOB state for a certain period of time, but also to model order-by-order data (the so-called MBO - market by order data). Such order flow models have been built under heavy use of point processes (E. Smith et al., 2003; Cont et al., 2010; Abergel & Jedidi, 2015; Huang et al., 2015; Morariu-Patrichi & Pakkanen, 2022). These point processes then produce every event affecting the LOB state. In most of the literature, order-by-order data of several days is used to obtain one single calibration of the model. These are then kept fixed which leads to stationary dynamics in the modeled order flow.

This study aims to improve this shortcoming of one single calibration by making use of existing approaches to simulate order flow and improving those with a hierarchical model on top of them. In particular, we build a probabilistic model based on generative adversarial networks (GANs), first introduced in (Goodfellow et al., 2014), to learn to generate new, unseen calibrations of different order flow models. This allows to introduce new, realistic dynamics into existing models, while preserving their theoretical properties and, in some cases, interpretability.

We show in this study that GANs can learn high-dimensional distributions of calibrations of smaller time periods. Furthermore, these can be efficiently generated conditioned on other covariates, such as time-of-day or the current value of the volatility index (VIX). After training, the synthetic – but realistic – calibrations can be subsequently used to simulate order streams using the baseline model. This induces stylized

behaviour like temporary drifts, different volatility regimes and intraday patterns one does not observe with one single calibration.

5.2 Related Work

Ever since exchanges have started using LOBs as the mechanism for electronic trading, vast amounts of publicly available historical data have led to a wealth of studies analyzing their empirical properties (Bouchaud et al., 2002; Cont, 2011), modeling their dynamics (Cont et al., 2010; Lehalle, Guéant, & Razafinimanana, 2011; Abergel & Jedidi, 2013) or predicting price moves (J. Sirignano & Cont, 2019; Zhang et al., 2019). Overviews are given in (Cont, 2011; Gould et al., 2013).

One main line of research has been to build models for the order flow of LOBs. This is to better understand their dynamics and to obtain analytical quantities for certain properties of the order book such as volatility or the expected time for queue depletion. (E. Smith et al., 2003) were among the first to use independent Poisson processes to model the inter-arrival times of different orders. (Cont et al., 2010) built a stochastic model for limit order books also based on Poisson processes, allowing for different intensities depending on the depth of the book and derive several analytic quantities, such as the probability that the mid-price increases (rather than decreases) or the probability of execution before mid-price movement. (Huang et al., 2015) propose a queue reactive model in which intensities depend on the current state of the order book. They claim the model is able to reproduce realistic market impact. (Muni Toke & Yoshida, 2017) present a parametric approach to model the state-dependent arrival rates of order book events.

To account for dependencies between the arrivals of limit orders, in particular self and cross-excitation of certain events, (Abergel & Jedidi, 2013, 2015) use Hawkes processes to model inter-arrival times of orders. Hawkes process, incorporating an excitation kernel, are able to capture the salient clustering effect of orders, especially after the occurrence of a market order. (Abergel & Jedidi, 2015) also study the long-term behavior of the LOB modeled with Hawkes process dynamics. The use of state-dependent Hawkes processes for LOBs is introduced in (Morariu-Patrichi & Pakkanen, 2022) to account for the different dynamics depending on the spread or the order imbalance. (Lu & Abergel, 2018) use non-linear Hawkes processes to model the order flow. Purely data-driven event streams are generated in (J. Li et al., 2020).

Agent-based models have been used to model the dynamics of different agents in the market and then simulate the joint order flow in (Paddrik et al., 2012; Byrd et

al., 2019). (Paddrik et al., 2012) use a decomposition of traders previously obtained by an analysis of the flash crash (Kirilenko et al., 2017). (Byrd et al., 2019) provide a general frame work for this type of event stream modeling. However, in agent-based modeling it is often difficult to define the agents and obtain a calibration which leads to realistic effects of the synthetic data.

In this study, we attempt to combine data-driven methods and commonly known LOB models – in particular Poisson order flow – in order to combine the best of both worlds. This line of work is in the spirit of (Marti, 2020) and (Storchan, Vyetrenko, & Balch, 2020). (Marti, 2020) uses GANs to sample realistic correlation matrices. (Storchan et al., 2020) presents a method called MAS-GAN in order to adversarial calibrate multi-agent simulators.

5.3 Generating realistic calibrations

Typical order flow models such as the Poisson order flow are often estimated using several days of data, see e.g. (Cont et al., 2010). Once estimated, Λ generally remains fixed. This leads to stationary modeling of order flow, regardless of, for instance, the time of the day or the volatility of the given day. The flow purely depends on the calibration in Equation (2.13).

Our present work suggests to generate "synthetic" calibrations in order to induce different, unseen but realistic dynamics, thereby rendering the order flow non-stationary. Instead of estimating order intensities for an entire time horizon (e.g. one day or month), one may estimate many calibrations over smaller time periods in order to obtain an entire data set of calibrations with distribution $\mathbb{P}_r(\Lambda)$. This, consequently entails the assumption that while markets are not stationary over an entire day, they may be more stationary for a short time interval.

The aim then is to approximate this distribution and draw samples from it. To this end, one may suggest a parametric distribution with a certain correlation structure. This study, instead, leverages generative adversarial networks (GANs) to learn generating calibrations without any assumptions on the form of distribution.

As introduced in (Goodfellow et al., 2014), a GAN consists out of two models (generally neural networks):

1. Generator (g): uses a noise seed from e.g. $Z \in \mathbb{R}^z, Z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_z)$ to map it into the sample space of Λ

2. Discriminator (d): gets real samples and "fake" samples – generated by g – with the aim to distinguish between those.

One design for the g is a simple unconditioned case

$$\Theta^{(g)} \times \mathbb{R}^z \to \mathbb{R}^{4k+2}_{\geq 0} \tag{5.1}$$

$$(\theta^{(g)}, Z) \mapsto g_{\theta^{(g)}}(Z) = \Lambda. \tag{5.2}$$

$\Theta^{(g)}$ is the parameter space of a neural network and $\mathbb{R}^Z$ the space of the noise input.

A generator as in Equation (5.1) is unconditional. Yet, the calibrations potentially differ substantially depending on variables such as time of day, momentum or volatility in the market. Conditional GAN (Mirza & Osindero, 2014) can be applied to condition the generated distribution on external (market) conditions, e.g.

$$\Theta^{(g)} \times \mathbb{R}^z \times \mathbb{R}^s \to \mathbb{R}^{4k+2}_{\geq 0}, \tag{5.3}$$

$$(\theta^{(g)}, Z, S) \mapsto g_{\theta^{(g)}}(Z, S) = \Lambda. \tag{5.4}$$

The vector $S \in \mathbb{R}^s$ is used by the generator to condition the generated calibrations of the order flow. The time of day and the state of the volatility index VIX of the trading day are used below, i.e. $S = (time, vix), S \in \mathbb{R}^2$.

The conditional discriminator is given by

$$\begin{aligned} d : \Theta^{(d)} \times \mathbb{R}^{4k+2} \times \mathbb{R}^s &\to [0, 1] \\ (\theta^{(d)}, \Lambda, S) &\mapsto d_{\theta^{(d)}}(\Lambda, S). \end{aligned} \tag{5.5}$$

In its first version from (Goodfellow et al., 2014), the GAN is then trained in a minmax game with loss function

$$\min_{\theta^{(g)}} \max_{\theta^{(d)}} \mathbb{E}_{x \sim \mathbb{P}_r(x)}[\log d_{\theta^{(d)}}(x)] + \mathbb{E}_{z \sim \mathbb{P}_z(z)}[\log 1 - d_{\theta^{(d)}}(g_{\theta^{(g)}}(z))]. \tag{5.6}$$

In words, the discriminator tries to correctly classify the calibrations as real or fake while the generator tries to fool the discriminator. In theory, the two players reach a Nash equilibrium in which the discriminator is unsure about every sample and outputs 0.5 for every sample (Goodfellow et al., 2014). In practical applications, however, GANs are difficult to train and to tune which has led to a lot of research trying to improve their training stability.

This setting and the condition S can further be extended with other conditions (e.g. previous calibrations, average traded volume during the time period etc.), and will constitute future work.

5.4 Results

5.4.1 Evaluation metrics

To track the similarity of the correlation, the Frobenius norm of the two matrices' differences is used. Let $\Sigma \in \mathbb{R}^{p\times p}$ be a correlation matrix, where $\sigma_{i,j} \forall i,j \in \{1,\dots p\}$ is the correlation between the i-th and j-th variable. The total number of variables is p. The Frobenius norm reads

$$\|\Sigma\|_F = \sqrt{\sum_i^p \sum_j^p |\sigma_{i,j}|^2}. \tag{5.7}$$

In the present case, $\sigma_{i,j} = \mathbb{E}(\lambda_i\lambda_j) - \mathbb{E}(\lambda_i)\mathbb{E}(\lambda_j)$ and $p = 4k+2$. During training, $\|\Sigma_{fake} - \Sigma_{real}\|_F$ is computed every 5 epochs.

To observe the convergence of the marginal distributions, the W1 distance (5.8) is used

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma\in\Pi(\mathbb{P}_r,\mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\left[\|x-y\|\right], \tag{5.8}$$

where $\mathbb{P}_r$ is the real and $\mathbb{P}_g$ is the generated distribution. Both the maximum as well as the mean of the W1 distances of each variable are tracked, i.e.

$$\frac{1}{4k+2}\sum_{i=1}^{4k+2} W(\mathbb{P}_{r_i}, \mathbb{P}_{g_i}) \quad \text{and} \quad \max_{i\in\{1,\dots,4k+2\}} W(\mathbb{P}_{r_i}, \mathbb{P}_{g_i})$$

where $\mathbb{P}_{r_i}$ ($\mathbb{P}_{g_i}$) is the real (generated) marginal distribution of λ_i.

To track the effect of the state conditions S_t, samples are generated conditioned on the range of the data. E.g. for daytime, the trading time of the NASDAQ stock (9:30 - 16:00) is split into even intervals with boundaries

$$(t_0,\dots,t_j,\dots,t_n) \text{ with } t_j - t_i = (j-i)\cdot c \;\forall\; j,i \in \{0,\dots,n\}, j>i \tag{5.9}$$

where $t_0 = 9.5$ (market open) and $t_n = 16$ (market close). For each interval (t_{j-1}, t_j), times are uniformly drawn from $U(t_{j-1}, t_j)$. All other conditions are randomly sampled from the range of the training data. It is of particular interest how the expected value of a certain variable λ_i deviates from the observed data. So in particular $\mathbb{E}_{\mathbb{P}_{g_i}}(\lambda_i | t \in [t_{j-1}, t_j)) - \mathbb{E}_{\mathbb{P}_{r_i}}(\lambda_i | t \in [t_{j-1}, t_j))$, the difference between the conditional expectations for λ_i drawn from the generated, marginal distribution $\mathbb{P}_{g_i}$ and the data distribution $\mathbb{P}_{r_i}$. Similar to the W1 distance, we compute the (squared) sum of all

deviations and the maximum in order to keep track of the general convergence, as well as the "worst" case

$$\sum_{i=1}^{4k+2}\sum_{j=1}^{n}\left[\mathbb{E}_{\mathbb{P}_{g_i}}(\lambda_i|t\in[t_{j-1},t_j))-\mathbb{E}_{\mathbb{P}_{r_i}}(\lambda_i|t\in[t_{j-1},t_j))\right]^2 \tag{5.10}$$

and

$$\max_{i\in\{1,\dots,4L+2\}}\sum_{j=1}^{n}\left[\mathbb{E}_{\mathbb{P}_{g_i}}(\lambda_i|t\in[t_{j-1},t_j))-\mathbb{E}_{\mathbb{P}_{r_i}}(\lambda_i|t\in[t_{j-1},t_j))\right]^2. \tag{5.11}$$

For the VIX index and the condition effect using the volatility index, we assume an equidistant grid similar to (5.9). The deviation is then computed equivalently to (5.10).

5.4.2 Parameter Simulation

To create a data set containing many different calibrations, intensities up to 5 ticks away (i.e. $k = 5$) from the best opposite price were estimated for 6 minute buckets using LOBSTER data for one year of the INTC ticker. The result is a data set of $\sim$ 16000 calibrations. Setting the intervals to 6 minutes implies the assumption of arrival rates to be stationary within this time period. The length of the time interval is subject to discussion and potentially even deserves an independent study. Due to the non-negativity of Λ, we model the standardized log intensity, i.e.

$$\tilde{\lambda}_i = \frac{\log(\lambda_i)-\mu_{\log(\lambda_i)}}{\sigma_{\log(\lambda_i)}}. \tag{5.12}$$

This standardized data set is used in a first step to train a GAN to learn (5.3). The results shown below use the loss setting from (Goodfellow et al., 2014). Both networks are fully connected neural networks with three hidden layers and either 32 or 64 hidden neurons. We train for up to 2000 iterations through the training data set with a learning rate of 0.00002 which takes about 30 minutes. Batch size is set to 32 and the noise dimension is 10. The ADAM optimizer is used for training.

Figure 5.1 visualizes all metrics during the training process on a logarithmic scale. Several things can be observed.

First, longer training tends to improve results. Looking at any of the metrics, there is a global decreasing trend with increasing number of epochs up until epoch 1000. Then the metrics start to fluctuate their current level. The first glimpse of the data is learned very fast by the GAN, which then more and more improves on smaller scales.

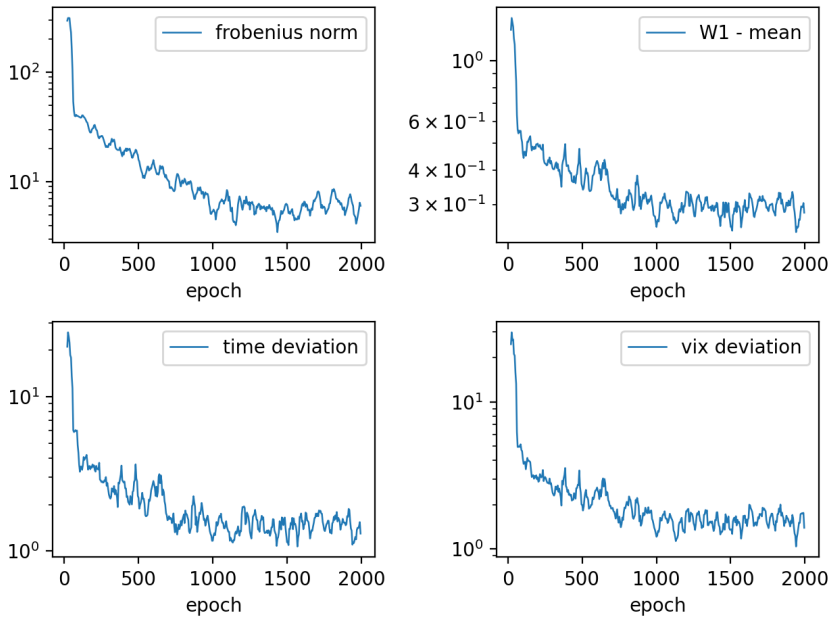


Figure 5.1: Rolling mean (5 epochs) of observed metrics on a log scale during training.

GANs are known to be unstable during optimization (see e.g. (Arjovsky et al., 2017; Gulrajani et al., 2017; Mescheder et al., 2017)). Also, training this model to generate the calibrations shows some fluctuations in the tracked metrics. These fluctuations appear to be aligned across different metrics, in particular to be observed in the later training stage.

Figure 5.2 shows the generated unconditional distributions for six of the 22 intensities (corresponding to $k = 5$), namely $\lambda_1^{C,a}$, $\lambda_1^{C,b}$, $\lambda_1^{L,a}$, $\lambda_1^{L,b}$, $\lambda^{M,a}$ and $\lambda^{M,b}$;

1. *lca1* corresponds to the "lambda cancellation ask" 1 tick away from the best bid price, $Q_1^a(t)$;

2. *lla1* corresponds to the "lambda limit order" submissions at $Q_1^a(t)$;

3. *lma* denoted the intensity "lambda market order" on the ask side $\lambda^{M,a}$.

In line with the mean W1 distance shown in the upper right plot in Figure 5.1, the distributions are mostly well fit, apart from slight deviations.

The order intensities, i.e. any λ, are non-stationary and in particular dependent on the time of the day. This impacts strongly the dynamics in the LOB. Figure 5.3 visualizes the expected value of the generated and the real order intensities condition on some time bucket. The solid orange like shows $\mathbb{E}_{\mathbb{P}_{r_i}}(\lambda_i | t \in [t_{j-1}, t_j))$, the expected value of the estimates for the bucket of the data, and the solid blue line

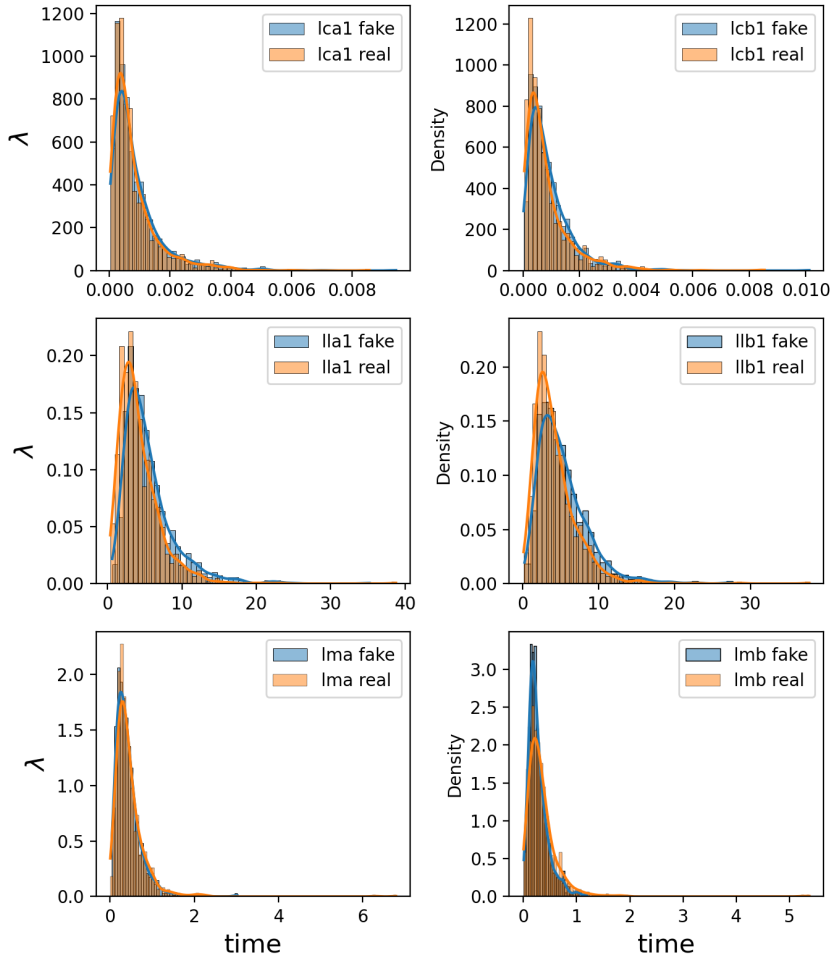


Figure 5.2: Intensities for cancellation, submission and market order intensities (one bid and one ask).

shows $\mathbb{E}_{\mathbb{P}_{g_i}}(\lambda_i | t \in [t_{j-1}, t_j))$, the mean generated value of the GAN for the particular point in time. The dashed lines indicate

$$\mathbb{E}_{\mathbb{P}_{g_i}}(\lambda_i | t \in [t_{j-1}, t_j)) \pm \sqrt{VAR_{\mathbb{P}_{g_i}}(\lambda_i | t \in [t_{j-1}, t_j))},$$

i.e. mean $\pm$ standard deviation of the value from the conditional order intensity.

In particular, the cancellations, *lca1* and *lcb1* are well fit, capturing the decreasing pattern throughout the day. Also the lambdas of limit order submissions, i.e. *lla1* and *llb1* show the U-shaped trend throughout the day, though with a slight over estimation. The most notable deviation is during the morning, which shows a significantly larger expected value for market orders on both bid as well as ask side.

Figure 5.4 shows the analogue plot to Figure 5.3 but when conditioning on different values of the volatility index VIX. As before, the orange (blue) line shows the mean

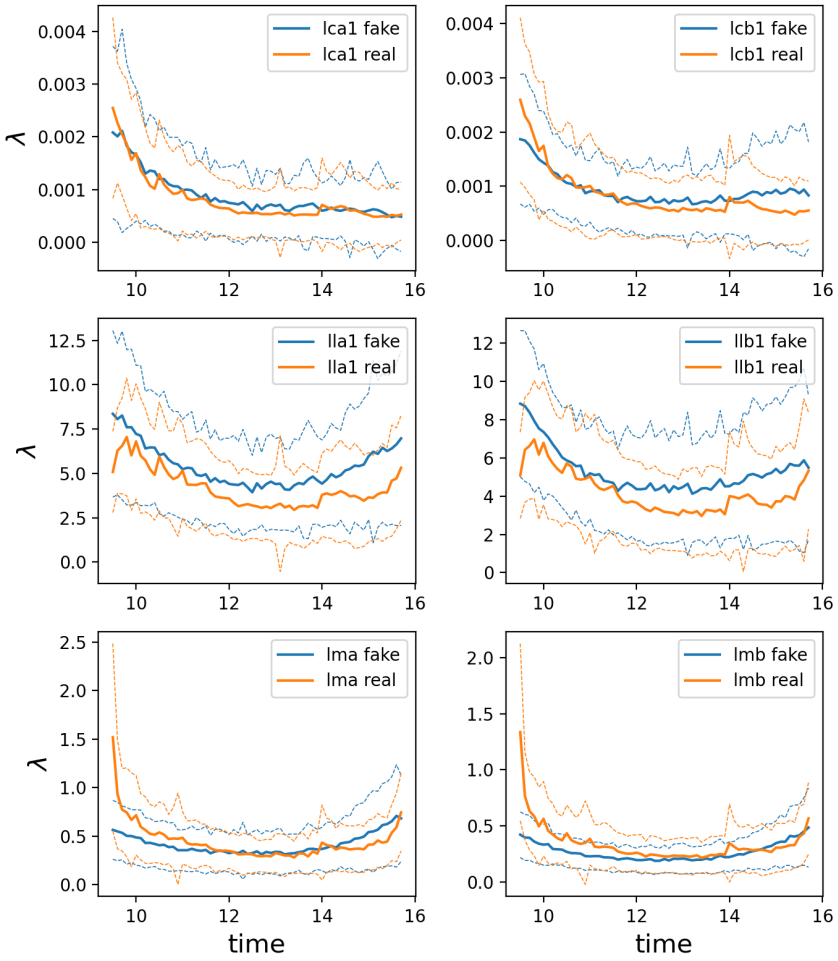


Figure 5.3: Mean intensities and standard deviations for six different intensities conditioned on the time of the day.

value for the real (fake) calibrations conditioned on a given value of the volatility index. Again, particularly for *lca1* and *lcb1*, the cancellation intensities one tick away from the best opposite prices are very well fit as the upper two plots indicate. Generally, for all intensities, the increasing pattern is clearly recognizable indicating the GAN easily picks up the pattern to condition effectively on time and volatility.

5.4.3 Event Stream Simulation with Dynamic Parameters

Results show that GANs automatically learn much of the structure of the calibrations with promising fits and without tuning of hyper parameters, as outlined above. Lastly, we want to use the generated calibrations Λ to simulate the even stream and compare it to both real data as well as the Poisson order flow without parameters.

To this end, we will simulate entire trading days of LOB data and compare the

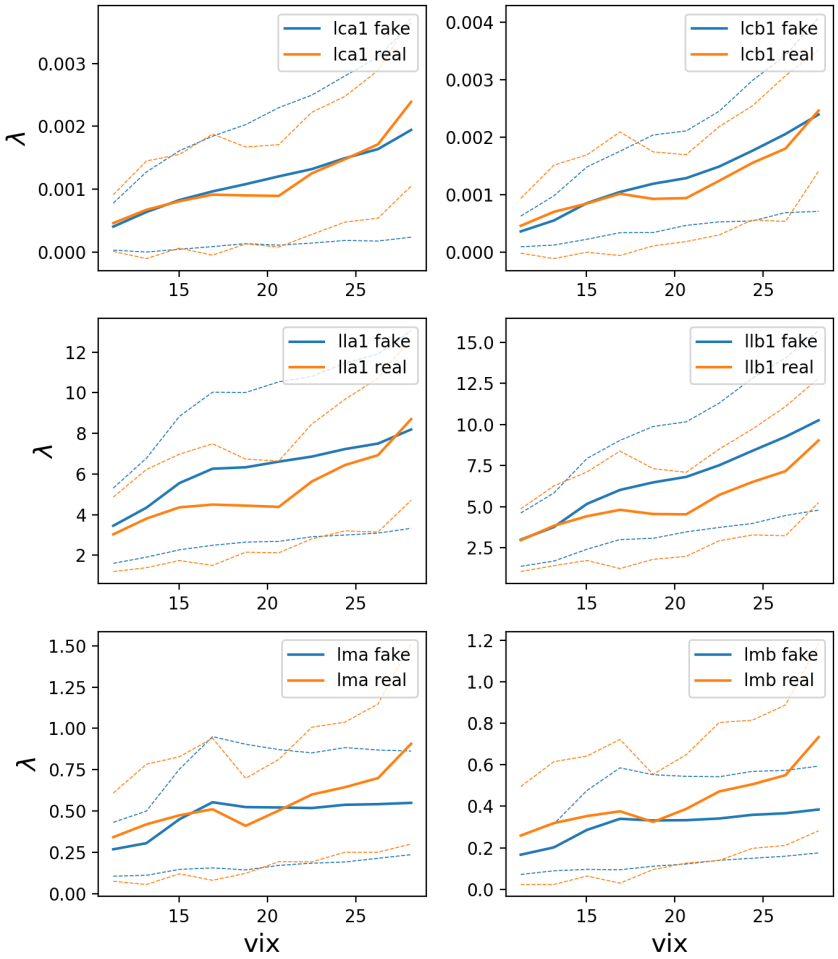


Figure 5.4: Mean intensities and standard deviations for six different intensities conditioned on the volatility index VIX.

statistics of the behavior of different paths of the LOB. This is done in two steps, which are then repeated in an iterative fashion.

1. The condition S for the sample/calibration is set and a calibration is generated.

2. The generated calibration is used to simulate the next time interval. For this time interval, the calibration does not change.

The time frame for which the calibration is kept constant in this example at the same value as for the estimation for the data set – 6 minutes. This implies the parameters change abruptly which is not necessarily optimal and subject for future research to investigate different (smoother) transitions between two consecutive calibrations or other time horizons.

Figure 5.5 shows the average of 100 exemplary simulations in comparison to the statistics of one exemplary day. The upper left plot shows the number of orders throughout the day. The simulations clearly exhibit the U-shaped pattern which can also be seen in the data of the real day. Furthermore, the high VIX conditioned simulations exhibit a higher level of orders that the lower VIX conditioned simulation. This is in line with Figure 5.4 which shows larger order intensities for higher VIX values and hence also more orders. For the Poisson order with a constant calibration, the number of orders is flat.

In the upper right, Figure 5.5 shows exemplary price paths from the simulation. It can be seen that the dynamic calibration clearly induces drifts and non-stationary dynamics in particular for the high VIX conditioned price path. It can also be noted that the real price path is more volatile despite a lower number of events. Part of the reason is the mean reverting behavior of the Poisson order flow which leads to decreasing cancellation intensities for smaller queue sizes. This lets the queue rather increase than decrease and the consequence are fewer price changes.

The lower left plot in Figure 5.5 indicates the volatility of the price changes throughout the day. In the present case, the price change is the mid price change during one second. Hence, the volatility for calibration bucket i reads $\sigma_i = \frac{1}{360}\sum_{j=1}^{360}(r_{i,j} - \bar{r}_i)^2$ where $r_{i,j} = \frac{p_{i,j} - p_{i,j-j}}{\delta}$ the j-th price change in the bucket i and $\bar{r}_i$ its mean. For comparison, the rolling volatility from the real exemplary day using 360 price changes is shown, which exhibits a decreasing pattern throughout the day. As mentioned, the calibrations conditioned on a high VIX value lead to higher volatility of the one-second price changes. Both simulations show a similar decreasing pattern as the real data (with a slight increase towards the end of the day), however, they are at a lower level than the sample real day. Again, it can be noted that the Poisson order flow tends to exhibit a less volatility than the real data (despite more events). For constant calibration, the volatility throughout the day is again constant.

The lower right plot in Figure 5.5 shows the distribution of the spread for the simulations and reference data. For all settings, the spreads of the model tend to be tighter. The reason for this is the baseline model. Assuming a spread of two ticks 2δ, we have the situation in Figure 2.3. Both $\lambda_1^{L,a}$ and $\lambda_1^{L,a}$ now sit at the empty level. The distribution shows that these intensities are much larger than in deeper levels. Thus, the probability of a limit order being posted in the empty level,

$$\frac{\lambda_1^{L,a} + \lambda_1^{L,b}}{\sum_{\lambda \in \Lambda} \lambda}, \tag{5.13}$$

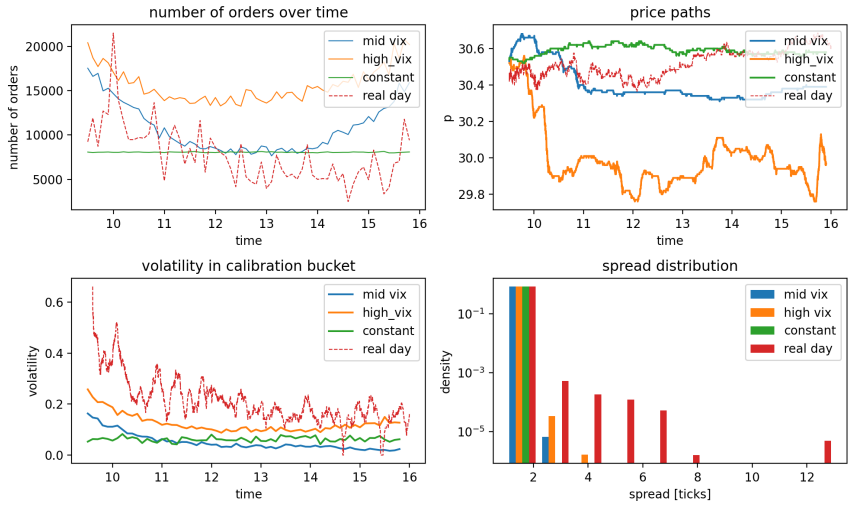


Figure 5.5: Order book simulation over the entire trading day for both a high and a low vix trading day.

is rather large. This leads to quickly disappearing spreads in the Poisson order flow. In the future one could add some state dependency to improve the spread behavior of the model. Table 5.1 shows the average intensities from the data set scaled by their sum, e.g.

$$\frac{\lambda_i^{L,\star}}{\sum_{i=1}^{k}(\lambda_i^{L,a}+\lambda_i^{L,b})} \tag{5.14}$$

which is equivalent to the probability where the next limit order submission will occur. It can be seen that in case of a spread equal to 2δ (as in Figure 5.5), the next limit order will be placed in the empty level between $p_b(t)$ and $p_a(t)$ with a probability of more than 76%. The consequence are quickly disappearing spreads which is not necessarily realistic.

i	1	2	3	4	5
$\lambda_i^{L,a}$	0.379	0.058	0.026	0.016	0.015
$\lambda_i^{L,b}$	0.384	0.059	0.026	0.016	0.015

Table 5.1: Table indicating limit order submission intensities for different levels scaled by their total sum.

5.5 Conclusion

Many models have been built to model the arrivals of orders in LOBs, the order flow. Amongst others, point processes have extensively been used as outlined in Section 5.2.

Most models, however, are stationary once they are estimated, and several days of data are used for parameter estimation.

This work proposes to build a probabilistic model on top of existing LOB models to learn the distribution of calibrations of smaller time periods conditioned on external variables. The aim of this hierarchical model is to introduce new dynamics and non-stationarity to the baseline model, while preserving its (theoretical) properties. In particular, we train a conditional GAN similar to the design presented in (Mirza & Osindero, 2014) to learn the conditional distribution of the calibrations for the Poisson order flow. The results show GANs can capture much of the structure in the data. Furthermore, the approach also learns the conditions on external variables, such as time of day and volatility index VIX, without extensive parameter tuning.

We then use the model to generate entire days of LOB data by iterative generation of calibrations and simulation of the order stream with the synthetic calibrations. Most importantly, the resulting simulations exhibit macro properties which are not observed with using only one global calibration, such as U-shaped intraday pattern, decreasing volatility over the day, or temporary drifts caused by imbalances in the calibrations.

Despite the promising results, there are several directions for improvement. First, the fit of the GAN and its training stability can be further improved as this study did not focus on extensive tuning. Second, one can study how long markets may be assumed to be stationary, and what is a suitable time interval that renders these intervals stochastic. Third, our approach adds non-stationary features in the macro perspective. It does not improve the model basis and certain weaknesses remain. E.g. the Poisson order flow with dynamic calibrations still does not show excitation effects. Furthermore, the Poisson order flow seems to produce less price changes than the real data shows. One reason for this is the mean reverting behavior of queue dynamics. Lastly, the estimation of the Poisson order flow is fairly simple. The calibration of Hawkes processes with a GAN may circumvent the computationally expensive estimation of its parameters. A possible extension would thus be to use other baseline LOB models, in particular Hawkes processes.

Chapter 6

Conclusion

6.1 Summary

This dissertation contributes to the rich literature on limit order books. In particular, we focus on the application of data-driven methods for the statistical modeling and simulation of limit order markets. This has the two main purposes of better understanding markets and their heterogeneity as well as the simulation of such. Both fields have a potentially large impact in both academia as well as industry.

In Chapter 3, we analyze a particular view on the limit order book, i.e. the *broker view*, using data from a major broker. Using unsupervised learning, we extract for heterogeneous agent types, which we interpret as *Quantitative*, *Day VWAP*, *Signal* and *Res* (residual). The agent types show heterogeneous behavior in many dimensions both on an individual as well as an aggregated level. We furthermore show stability of agent types both across instruments, as well as across the temporal dimension, altogether indicating that the findings are stable observations that can be persistently observed. The findings allow for the development of parent order flow models and agent-based models with actual evidence for certain agent types.

In Chapter 4, we turn towards the direct simulation of limit order books. We consider the simplified problem of probabilistic modeling of LOB snapshots. Generative adversarial networks are leveraged to learn the conditional probability distribution of future order book snapshots. Learning the probability distribution accurately enough allows to generate time series of LOB snapshots whose price paths show similar properties to the ones observed in real data. Despite losing some degree of granularity, results show the generative model automatically learns realistic patterns of market impact. This allows the use of such a method for backtesting execution strategies and many other applications.

In Chapter 5, we pursue a hybrid approach. Instead of generating synthetic data directly in a data-driven fashion, we learn the distribution of parameters of conventional order flow models. This allows to preserve the theoretical results of such a model while enhancing it with a data-driven model on top. Precisely, we train a generative model to learn the calibrations of Poisson order flow on shorter time frames for which markets may assumed to be stationary. The model easily learns the distribution of the parameters as well as conditional patterns without imposing any structure on these. The synthetic calibrations can then be used for the order book model to generate data which exhibits non-stationary patterns such as intraday effects, volatility dependency, or temporal trends. Alternatively, the synthetic parameters can be used for other downstream tasks for which they may be of interest. The approach can easily be extended to other order flow models.

The field of machine learning in finance as well as research activity of limit order markets is far from complete and rapidly evolving. While offering opportunities, it is important to properly research applications. This thesis straddles this boundary and makes the next step on this avenue. While many results were found, this work also shows that there are aspects that can, and potentially must, be explored in order to, for instance, use certain methods in an industry setting.

6.2 Future work

6.2.1 Limit order book simulation with GANs

Chapter 4 introduces a GAN-based model to learn the conditional distribution of future order book snapshots. Despite very good results, there are many avenues that can be pursued in this direction to improve or extend the model as mentioned in the conclusion (Section 4.7) which are worth investigating.

Temporal dependency. The results presented in Chapter 4 are based on a condition of $S_t = X_t$ for the generator. While this simple representation already shows a lot of expected properties, it raises the question of how the inclusion of further history improves the model. In particular, Figure 4.11 does not show a decaying market impact which is commonly observed in the literature. Preliminary results show that market impact decay can be learned by our GAN approach when including more history.

Multi-asset simulation In Chapter 4, we learn to generate the dynamics of one asset and investigate how interaction with the order book state impacts the market. A natural extension is to use two or more assets in a similar framework to simulate co-movements or potentially entire markets. Some key questions may be considered here. Can cross-impact be learned? In particular, training a similar model which learns the joint order book dynamics for, i.e. two stocks. Furthermore, to which extent can cross-asset impact be learned from data by the model? Lastly, how can a multi-asset model be scaled to entire universes?

Trading in a generated LOB with reinforcement learning Synthetic data/ market generators are often built with the aim of using them as an environment in reinforcement learning settings because of a lack of data and because the system has to be interactive. The latter means an environment/simulator should respond to the actions of an agent. The next step is to deploy the simulator from Chapter 4 as an environment for an agent that performs e.g. a round trip trade or an execution of a certain trading quantity. Questions of interest would be to study the general behavior of an agent, i.e. at what time it trades and what quantity it chooses depending on the available liquidity. Furthermore, such a setting could be used to investigate whether the agent is able to discover some form of arbitrage in the environment.

Sparse order books. For stocks with an average spread larger than δ or frequent empty levels in general, the distribution of X_t is zero-inflated. With linear activation in the output layer, the generator has to map many noise seeds exactly or very close to 0 to represent this zero inflation. This is generally harder to learn. Instead, one may "support" the model to generate zero-inflated distribution. Generally, the ReLU activation function is suitable to do so, mapping $\mathbb{R} \to \mathbb{R}_0^+$

$$\text{ReLU}(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0, \end{cases} \tag{6.1}$$

mapping all negative values to 0. For the present problem, mapping to $\mathbb{R}_0^+$ is undesirable as the sign indicates whether $Q_p(t)$ is a bid or ask quantity to indicate price changes.

One alternative would be to map X_t into a two-dimensional array, one for bid and one for ask quantities, e.g.

$$X_t = \begin{bmatrix} Q_p^a(t) & Q_{p+\delta}^a(t) & Q_{p+2\cdot\delta}^a(t) & Q_{p+3\cdot\delta}^a(t) & Q_{p+4\cdot\delta}^a(t) & Q_{p+5\cdot\delta}^a(t) \\ Q_p^b(t) & Q_{p+\delta}^b(t) & Q_{p+2\cdot\delta}^b(t) & Q_{p+3\cdot\delta}^b(t) & Q_{p+4\cdot\delta}^b(t) & Q_{p+5\cdot\delta}^b(t), \end{bmatrix} \tag{6.2}$$

such that $Q_p^a(t) \cdot Q_p^b(t) = \mathbf{0} \; \forall \; p$ and $Q_p^{a/b}(t) \geq 0$. The upper (lower) row indicates ask (bid) quantities. $Q_p^{a/b}(t) \geq 0$ can be enforced with a ReLU activation to facilitate zero inflation. Furthermore, $Q_p^a(t) \cdot Q_p^b(t) = \mathbf{0}$ must hold for any p to ensure a price level which either corresponds to a bid or ask quantity. In this representation, the generator can more easily map frequent values to zero. However, this representation comes with constraints the model needs to learn.

Order stream and fill dynamics. Currently, the presented model in Chapter 4 learns the conditional probability distribution of LOB snapshots. However, the model does not reflect any dynamics of the flow and thus does not give insights into fill dynamics. For example, the snapshot at t is the same as $t + \Delta t$, i.e. $X_t = X_{t+\Delta t}$. However, this does not mean that no order has been submitted from t to Δt. In the future, the state representation may be enhanced by quantities such as the cumulative order flow between t and Δt.

In this case, the output of the generator g could take the form

$$X_t = \begin{bmatrix} Q_p(t) & Q_{p+\delta}(t) & Q_{p+2\cdot\delta}(t) & Q_{p+3\cdot\delta}(t) & Q_{p+4\cdot\delta}(t) & Q_{p+5\cdot\delta}(t) \\ OF_p^L((t,t+\Delta t]) & OF_{p+\delta}^L((t,t+\Delta t]) & OF_{p+2\cdot\delta}^L((t,t+\Delta t]) & OF_{p+3\cdot\delta}^L((t,t+\Delta t]) & OF_{p+4\cdot\delta}^L((t,t+\Delta t]) & OF_{p+5\cdot\delta}^L((t,t+\Delta t]) \\ OF_p^M((t,t+\Delta t]) & OF_{p+\delta}^M((t,t+\Delta t]) & OF_{p+2\cdot\delta}^M((t,t+\Delta t]) & OF_{p+3\cdot\delta}^M((t,t+\Delta t]) & OF_{p+4\cdot\delta}^M((t,t+\Delta t]) & OF_{p+5\cdot\delta}^M((t,t+\Delta t]) \end{bmatrix} \tag{6.3}$$

$$X_t = \begin{bmatrix} \cdots & Q_{p+\delta}(t) & Q_{p+2\cdot\delta}(t) & Q_{p+3\cdot\delta}(t) & Q_{p+4\cdot\delta}(t) & \cdots \\ \cdots & OF_{p+\delta}^L((t,t+\Delta t]) & OF_{p+2\cdot\delta}^L((t,t+\Delta t]) & OF_{p+3\cdot\delta}^L((t,t+\Delta t]) & OF_{p+4\cdot\delta}^L((t,t+\Delta t]) & \cdots \\ \cdots & OF_{p+\delta}^M((t,t+\Delta t]) & OF_{p+2\cdot\delta}^M((t,t+\Delta t]) & OF_{p+3\cdot\delta}^M((t,t+\Delta t]) & OF_{p+4\cdot\delta}^M((t,t+\Delta t]) & \cdots \end{bmatrix} \tag{6.4}$$

with $OF_p^L((t,t+\Delta t]) = \Sigma_{x \in \mathcal{L}((t,t+\Delta t])} q_x$, where

$$\mathcal{L}((t,t+\Delta t]) = \{x \mid t_x \in (t,t+\Delta t], p_x = p\},$$

denotes the cumulative quantity of limit orders submitted at price level p between the two consecutive snapshots.

Additionally, $OF_p^M((t,t+\Delta t]) = \Sigma_{x \in \mathcal{L}^M((t,t+\Delta t])} q_x$ where

$$\mathcal{L}^M((t,t+\Delta t]) = \{x \mid t_x \in (t,t+\Delta t], p_x = \{0,\infty\}, p_b(t_x) = p_x \vee p_a(t_x) = p_x\}$$

is the cumulative quantity of market orders sent in $(t,t+\Delta t]$ affecting the queue at price p. For this, during the submission of a particular market order, p must be either $p_a(t)$ or $p_b(t)$. The remaining quantity is then the cumulative cancellation flow, $OF_p^C((t,t+\Delta t])$, computed via $Q_p(\tilde{t}) - Q_p(t) - OF_p^M((t,t+\Delta t]) + OF_p^L((t,t+\Delta t])$.

Instead of directly modeling future state and the order flow statistics in one step, a sequential model on top of the existing approach may help to model realistic flows

based on a given transition:

$$\begin{aligned} g : \Theta^{(g)} \times \mathbb{R}^z \times \mathbb{R}^s \times \mathbb{R}^{2k} &\to \mathbb{R}^{2k \times 2} \\ (\theta^{(g)}, Z, S, X_{t+\Delta t}) &\mapsto g_{\theta^{(g)}}(Z_t, S_t, X_{t+\Delta t}) = OF((t, t+\Delta t]). \end{aligned} \tag{6.5}$$

Including further quantities describing the order flow between X_t and $X_{t+\Delta t}$ brings several advantages. First, one can investigate the dynamics of the order flow between two states. Second, practical applications given certain assumptions can be made when it comes to the interaction of the order flow. In other words, by using the executed quantity and an assumption about the cancellations in the queue, one can approximate a fill probability. Alternatively, another model may be built to use the order flow quantities to approximate whether an order is executed within the next Δt step or not.

Training stability and symmetry The first part of Section 4.5 addresses the training stability with GANs – in general and the present case. Instability is a common problem when training GANs, as also mentioned in many studies (Goodfellow et al., 2014; Arjovsky et al., 2017; Gulrajani et al., 2017; Mescheder et al., 2017, 2018).

While certain methods such as gradient clipping and gradient penalty can improve the training process as shown in Figure 4.6, there is some remaining fluctuation of the model during training. These small fluctuations can lead to small imbalances in the generated distributions which may lead to heavy drifts in the generated price paths.

To this end, it would be desirable to obtain a more stable training process and a model which – at best by design – is symmetric for the bid/ask side. To achieve this symmetry, several approaches may be pursued from sampling techniques, via regularization of imbalances in sign changes, up to the design of a symmetric neural network.

Global conditions At this stage, the model does not distinguish between effects such as time of day, implied volatility, and other potential conditions. Markets, however, tend to be more volatile in the morning. In addition, trading generally becomes less expensive towards the end of the trading day, since many participants want to unload their inventory. In the future, such global conditions may be included in the model in order to account for such non-stationarities.

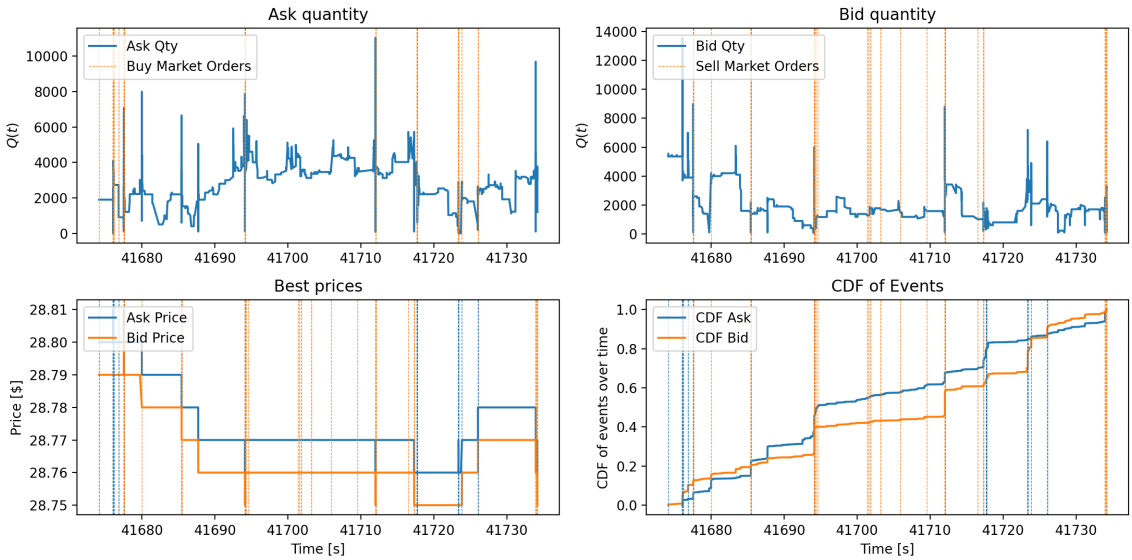


Figure 6.1: Queue process for one minute. Ask queue (upper left), bid queue (upper right), best prices (lower left), and fraction of events during the observed interval (lower right). Vertical dashed lines indicate market orders.

6.2.2 Order bursts

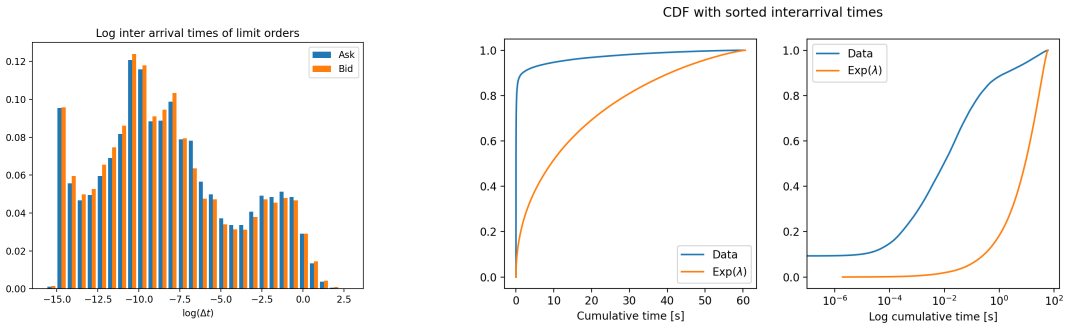
It is well known that orders arrive in clusters at exchanges. Hawkes processes have been particularly successful modeling the event stream of LOBs due to their ability to capture excitement effects of orders (both self and cross excitation), especially in (Abergel & Jedidi, 2015; Morariu-Patrichi & Pakkanen, 2022).

However, when looking at empirical properties of event streams, the question arises whether Hawkes processes are able to capture the sudden increase in the arrival of orders and the sudden decay. One reason for this is that Hawkes processes – in particular using exponential kernels – are subject to stability constraints.

Figure 6.1 shows an exemplary queue process for a stock in one minute. The upper plots show the ask (left) and bid (right) queues. The lower left shows the best ask (bid) price and the lower right plot shows the fraction of events that have happened within the displayed time horizon – around one minute with 4000 events. Additionally, market orders are indicated using horizontal lines.

It can be seen that the queue sizes of both queues show strong non-stationary behavior. In particular, there seem to be bursts of orders during which the cdf of events shown on the lower right exhibits jumps. Furthermore, it seems that market orders are mostly submitted during these periods and also price changes occur predominantly during such bursts. However, market orders and price changes do not necessarily

appear to be leading factors, but often also occur within these bursts. These abrupt changes in frequency suggest that LOBs may be looked at in two regimes, a slow one and a fast one.



(a) Log interarrival times (b) Cumulative sorted interarrival times

Figure 6.2: Distribution of inter-arrival times. Both the logarithm as well as the cumulative sorted interarrival times vs. the fraction of the number of orders are shown.

Figure 6.2a shows the histogram of inter-arrival times on a logarithmic scale for both limit orders on the best bid and best ask queue. Clearly, this distribution appears to be bimodal. The right mode incorporates the rather slow regime with larger inter-arrival times and the left mode incorporates the fast regime. Modeling inter-arrival times with state-independent Hawkes processes (as for instance in (Abergel & Jedidi, 2015)) does not lead to two modes. (Morariu-Patrichi & Pakkanen, 2022) use state-dependent Hawkes processes to model the event stream. However, the state refers to spread and order imbalance.

Figure 6.2b displays the cumulative sum of sorted inter-arrival times versus the fraction of events that have happened until that point. This is shown for the data as well as a Poisson process using the intensity estimated using the data. This is both shown on linear (left) and logarithmic (right) scales. The two different curves differ substantially. In fact, 80% of the orders arrive only in a cumulative arrival time of ~ 0.18 seconds. In comparison to the Poisson process simulation, this is around 30 seconds. The right-hand plot in Figure 6.2b showing the logarithm of the left hand further supports this striking discrepancy.

These observations suggest that markets may be modeled via different regimes rather than self-exciting processes which then smoothly decay. Instead, one can assume a "calm" regime and a "burst" regime. The literature around market microstructure has not yet explored these burst phenomena in detail. Hence, a number of directions could be investigated:

1. Classification of regimes. When is a period to be classified as a burst? In particular, methods from (online) change-point detection may be used here, both from the univariate and multivariate settings.

2. When and how often do these burst regimes occur, are they correlated with other assets and what information do they contain towards, for example, future price moves?

3. How does a model perform which has two regimes in comparison to Hawkes and Poisson order flow?

Many of the suggested research directions are direct extensions to the work in this thesis and surely worth to be further investigated.

References

Abergel, F., Anane, M., Chakraborti, A., Jedidi, A., & Toke, I. M. (2016). *Limit Order Books*. Cambridge University Press.

Abergel, F., & Jedidi, A. (2013). A mathematical approach to order book modeling. *International Journal of Theoretical and Applied Finance*, *16*(05), 1350025.

Abergel, F., & Jedidi, A. (2015). Long-time behavior of a Hawkes process–based limit order book. *SIAM Journal on Financial Mathematics*, *6*(1), 1026–1043.

Alfonsi, A., & Blanc, P. (2016). Dynamic optimal execution in a mixed-market-impact Hawkes price model. *Finance and Stochastics*, *20*(1), 183–218.

Alfonsi, A., Fruth, A., & Schied, A. (2010). Optimal execution strategies in limit order books with general shape functions. *Quantitative Finance*, *10*(2), 143–157.

Almgren, R., & Chriss, N. (2001). Optimal execution of portfolio transactions. *Journal of Risk*, *3*, 5–40.

Almgren, R. F. (2003). Optimal execution with nonlinear impact functions and trading-enhanced risk. *Applied Mathematical Finance*, *10*(1), 1–18.

Ang, A., & Bekaert, G. (2007). Stock return predictability: Is it there? *The Review of Financial Studies*, *20*(3), 651–707.

Ariyo, A. A., Adewumi, A. O., & Ayo, C. K. (2014). Stock price prediction using the ARIMA model. In *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation* (pp. 106–112).

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein Generative Adversarial Networks. In *International Conference on Machine Learning* (pp. 214–223).

Assefa, S. A., Dervovic, D., Mahfouz, M., Tillman, R. E., Reddy, P., & Veloso, M. (2020). Generating synthetic data in finance: opportunities, challenges and pitfalls. In *Proceedings of the first acm international conference on ai in finance* (pp. 1–8).

Bacry, E., Delattre, S., Hoffmann, M., & Muzy, J.-F. (2013). Modelling microstructure noise with mutually exciting point processes. *Quantitative Finance*, *13*(1), 65–77.

Bacry, E., Iuga, A., Lasnier, M., & Lehalle, C.-A. (2015). Market impacts and the life cycle of investors orders. *Market Microstructure and Liquidity*, *1*(02), 1550009.

Bacry, E., Mastromatteo, I., & Muzy, J.-F. (2015). Hawkes processes in finance. *Market Microstructure and Liquidity*, *1*(01), 1550005.

Bacry, E., & Muzy, J.-F. (2014). Hawkes model for price and trades high-frequency dynamics. *Quantitative Finance*, *14*(7), 1147–1166.

Barbon, A., Di Maggio, M., Franzoni, F., & Landier, A. (2019). Brokers and order

flow leakage: Evidence from fire sales. *The Journal of Finance*, *74*(6), 2707–2749.

Belkin, M., & Niyogi, P. (2002). Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems* (pp. 585–591).

Bennett, S., Cucuringu, M., & Reinert, G. (2022). Detection and clustering of lead-lag networks for multivariate time series with an application to financial markets.

Bouchaud, J.-P., Farmer, J. D., & Lillo, F. (2009). How markets slowly digest changes in supply and demand. In *Handbook of Financial Markets: Dynamics and Evolution* (pp. 57–160). Elsevier.

Bouchaud, J.-P., Mézard, M., Potters, M., et al. (2002). Statistical properties of stock order books: empirical results and models. *Quantitative Finance*, *2*(4), 251–256.

Brogaard, J., Hendershott, T., & Riordan, R. (2014). High-frequency trading and price discovery. *The Review of Financial Studies*, *27*(8), 2267–2306.

Brogaard, J., et al. (2010). High frequency trading and its impact on market quality. *Northwestern University Kellogg School of Management Working Paper*, *66*.

Buehler, H., Horvath, B., Lyons, T., Perez Arribas, I., & Wood, B. (2020). Generating financial markets with signatures. *Available at SSRN*.

Buehler, H., Murray, P., Pakkanen, M. S., & Wood, B. (2021). Deep hedging: learning to remove the drift under trading frictions with minimal equivalent near-martingale measures. *arXiv preprint arXiv:2111.07844*.

Byrd, D., Hybinette, M., & Balch, T. H. (2019). ABIDES: Towards high-fidelity market simulation for AI research. *arXiv preprint arXiv:1904.12066*.

Cartea, Á., Cohen, S. N., Graumans, R., Labyad, S., Sánchez-Betancourt, L., & van Veldhuijzen, L. (2023). Statistical predictions of trading strategies in electronic markets. *Available at SSRN 4442770*.

Cartea, Á., Cohen, S. N., & Labyad, S. (2021). Gradient-based estimation of linear hawkes processes with general kernels. *arXiv preprint arXiv:2111.10637*.

Cartea, A., & Jaimungal, S. (2015). Optimal execution with limit and market orders. *Quantitative Finance*, *15*(8), 1279–1291.

Cartea, Á., Jaimungal, S., & Penalva, J. (2015). *Algorithmic and high-frequency trading*. Cambridge University Press.

Cartea, Á., Jaimungal, S., & Wang, Y. (2020). Spoofing and price manipulation in order-driven markets. *Applied Mathematical Finance*, *27*(1-2), 67–98.

Cartea, A., Payne, R., Penalva, J., & Tapia, M. (2019). Ultra-fast activity and intraday market quality. *Journal of Banking & Finance*, *99*, 157–181.

Cirulli, A., Kobak, M., & Ulrych, U. (2022). Portfolio construction with hierarchical momentum. *Available at SSRN 4125072*.

Cohen, S. N., Snow, D., & Szpruch, L. (2021). Black-box model risk in finance. *arXiv preprint arXiv:2102.04757*, *10*.

Coletta, A., Moulin, A., Vyetrenko, S., & Balch, T. (2022). Learning to simulate realistic limit order book markets from data as a world agent. In *Proceedings of the Third ACM International Conference on AI in Finance* (pp. 428–436).

Cont, R. (2001). Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, *1*(2), 223.

Cont, R. (2011). Statistical modeling of high-frequency financial data. *IEEE Signal Processing Magazine*, *28*(5), 16–25.

Cont, R., Cucuringu, M., Glukhov, V., & Prenzel, F. (2023). Analysis and modeling of client order flow in limit order markets. *Quantitative Finance*, 1–19.

Cont, R., Cucuringu, M., Kochems, J., & Prenzel, F. (2023). Limit order book simulation with generative adversarial networks. *Available at SSRN 4512356*.

Cont, R., Degond, P., & Xuan, L. (2023). A mathematical framework for modelling order book dynamics. *arXiv preprint arXiv:2302.01169*.

Cont, R., & De Larrard, A. (2013). Price dynamics in a Markovian limit order market. *SIAM Journal on Financial Mathematics*, *4*(1), 1–25.

Cont, R., Kukanov, A., & Stoikov, S. (2014). The price impact of order book events. *Journal of Financial Econometrics*, *12*(1), 47–88.

Cont, R., & Müller, M. S. (2021). A stochastic partial differential equation model for limit order book dynamics. *SIAM Journal on Financial Mathematics*, *12*(2), 744–787.

Cont, R., Stoikov, S., & Talreja, R. (2010). A stochastic model for order book dynamics. *Operations research*, *58*(3), 549–563.

Da Silva, B., & Shi, S. S. (2019). Style transfer with time series: Generating synthetic financial data. *arXiv preprint arXiv:1906.03232*.

Di Maggio, M., Franzoni, F., Kermani, A., & Sommavilla, C. (2019). The relevance of broker networks for information diffusion in the stock market. *Journal of Financial Economics*, *134*(2), 419–446.

Di Maggio, M., Kermani, A., & Song, Z. (2017). The value of trading relations in turbulent times. *Journal of Financial Economics*, *124*(2), 266–284.

Donnelly, R. (2022). Optimal execution: A review. *Applied Mathematical Finance*, 1–32.

Dutta, C., Karpman, K., Basu, S., & Ravishanker, N. (2023). Review of statistical approaches for modeling high-frequency trading data. *Sankhya B*, *85*(Suppl 1), 1–48.

Eckerli, F., & Osterrieder, J. (2021). Generative adversarial networks in finance: an overview. *arXiv preprint arXiv:2106.06364*.

Eisler, Z., Bouchaud, J.-P., & Kockelkoren, J. (2012). The price impact of order book events: Market orders, limit orders and cancellations. *Quantitative Finance*, *12*(9), 1395–1419.

Farmer, J. D., & Foley, D. (2009). The economy needs agent-based modelling. *Nature*, *460*(7256), 685–686.

Farmer, J. D., Patelli, P., & Zovko, I. I. (2005). The predictive power of zero intelligence in financial markets. *Proceedings of the National Academy of Sciences*, *102*(6), 2254–2259.

Gatheral, J. (2010). No-dynamic-arbitrage and market impact. *Quantitative finance*, *10*(7), 749–759.

Gatheral, J., & Schied, A. (2013). Dynamical models of market impact and algorithms for order execution. *Handbook on Systemic Risk, Jean-Pierre Fouque, Joseph*

A. Langsam, eds, 579–599.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems* (pp. 2672–2680).

Gould, M. D., Porter, M. A., Williams, S., McDonald, M., Fenn, D. J., & Howison, S. D. (2013). Limit order books. *Quantitative Finance*, *13*(11), 1709–1742.

Guéant, O. (2016). *The financial mathematics of market liquidity: From optimal execution to market making* (Vol. 33). CRC Press.

Guéant, O. (2017). Optimal market making. *Applied Mathematical Finance*, *24*(2), 112–154.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems* (pp. 5767–5777).

Hagströmer, B., & Nordén, L. (2013). The diversity of high-frequency traders. *Journal of Financial Markets*, *16*(4), 741–770.

Hamilton, J. D. (2020). *Time series analysis*. Princeton university press.

Hasbrouck, J., & Saar, G. (2013). Low-latency trading. *Journal of Financial Markets*, *16*(4), 646–679.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Hautsch, N., & Huang, R. (2012). The market impact of a limit order. *Journal of Economic Dynamics and Control*, *36*(4), 501–522.

Hawkes, A. G. (1971). Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, *58*(1), 83–90.

Hendershott, T., Jones, C. M., & Menkveld, A. J. (2011). Does algorithmic trading improve liquidity? *The Journal of finance*, *66*(1), 1–33.

Hendershott, T., Li, D., Livdan, D., & Schürhoff, N. (2020). Relationship trading in over-the-counter markets. *The Journal of Finance*, *75*(2), 683–734.

Hendricks, D., & Wilcox, D. (2014). A reinforcement learning extension to the almgren-chriss framework for optimal trade execution. In *2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)* (pp. 457–464).

Hewlett, P. (2006). Clustering of order arrivals, price impact and trade path optimisation. In *Workshop on Financial Modeling with Jump processes, Ecole Polytechnique* (pp. 6–8).

Horvath, B., Issa, Z., & Muguruza, A. (2021). Clustering market regimes using the wasserstein distance. *arXiv preprint arXiv:2110.11848*.

Huang, W., Lehalle, C.-A., & Rosenbaum, M. (2015). Simulating and analyzing order book data: The queue-reactive model. *Journal of the American Statistical Association*, *110*(509), 107–122.

Issa, Z., & Horvath, B. (2023). Non-parametric online market regime detection and regime clustering for multidimensional and path-dependent data structures. *arXiv preprint arXiv:2306.15835*.

Jordon, J., Szpruch, L., Houssiau, F., Bottarelli, M., Cherubin, G., Maple, C., ... Weller, A. (2022). Synthetic data–what, why and how? *arXiv preprint*

arXiv:2205.03257.

Kercheval, A. N., & Zhang, Y. (2015). Modelling high-frequency limit order book dynamics with support vector machines. *Quantitative Finance*, *15*(8), 1315–1329.

Kirilenko, A., Kyle, A. S., Samadi, M., & Tuzun, T. (2017). The flash crash: High-frequency trading in an electronic market. *The Journal of Finance*, *72*(3), 967–998.

Lehalle, C.-A., Guéant, O., & Razafinimanana, J. (2011). High-frequency simulations of an order book: A two-scale approach. In *Econophysics of Order-driven Markets* (pp. 73–92). Springer.

Lewis, P. W., & Shedler, G. S. (1979). Simulation of nonhomogeneous poisson processes by thinning. *Naval research logistics quarterly*, *26*(3), 403–413.

Li, C.-L., Chang, W.-C., Cheng, Y., Yang, Y., & Póczos, B. (2017). MMD GAN: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems* (pp. 2203–2213).

Li, J., Wang, X., Lin, Y., Sinha, A., & Wellman, M. (2020). Generating realistic stock market order streams. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, pp. 727–734).

Limmer, Y., & Horvath, B. (2023). Robust hedging gans. *Available at SSRN 4489029*.

Lin, S.-Y., Liu, D.-R., & Huang, H.-P. (2022). Credit default swap prediction based on generative adversarial networks. *Data Technologies and Applications*, *56*(5), 720–740.

Lu, X., & Abergel, F. (2018). High-dimensional Hawkes processes for limit order books: modelling, empirical analysis and numerical calibration. *Quantitative Finance*, *18*(2), 249–264.

Lucchese, L., Pakkanen, M., & Veraart, A. (2022). The short-term predictability of returns in order book markets: a deep learning perspective. *arXiv preprint arXiv:2211.13777*.

MacQueen, J., et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (Vol. 1, pp. 281–297).

Malkiel, B. G. (1989). Efficient market hypothesis. *Finance*, 127–134.

Malkiel, B. G. (2003). The efficient market hypothesis and its critics. *Journal of Economic Perspectives*, *17*(1), 59–82.

Mankad, S., Michailidis, G., & Kirilenko, A. (2013). Discovering the ecosystem of an electronic financial market with a dynamic machine-learning method. *Algorithmic Finance*, *2*(2), 151–165.

Marti, G. (2020). Corrgan: Sampling realistic financial correlation matrices using generative adversarial networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 8459–8463).

Meila, M., & Shi, J. (2001). A random walks view of spectral segmentation.

Mescheder, L., Geiger, A., & Nowozin, S. (2018). Which training methods for GANs do actually converge? In *International Conference on Machine Learning* (pp. 3481–3490).

Mescheder, L., Nowozin, S., & Geiger, A. (2017). The numerics of GANs. In *Advances in Neural Information Processing Systems* (pp. 1825–1835).

Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Moallemi, C. C., & Yuan, K. (2016). A model for queue position valuation in a limit order book. *Columbia Business School Research Paper No. 17-70*.

Morariu-Patrichi, M., & Pakkanen, M. S. (2022). State-dependent Hawkes processes and their application to limit order book modelling. *Quantitative Finance*, *22*(3), 563–583.

Muni Toke, I., & Yoshida, N. (2017). Modelling intensities of order flows in a limit order book. *Quantitative Finance*, *17*(5), 683–701.

Nevmyvaka, Y., Feng, Y., & Kearns, M. (2006). Reinforcement learning for optimized trade execution. In *Proceedings of the 23rd International Conference on Machine Learning* (pp. 673–680).

Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems* (pp. 849–856).

Ni, H., Szpruch, L., Wiese, M., Liao, S., & Xiao, B. (2020). Conditional sig-wasserstein GANs for time series generation. *arXiv preprint arXiv:2006.05421*.

Ning, B., Lin, F. H. T., & Jaimungal, S. (2021). Double deep q-learning for optimal execution. *Applied Mathematical Finance*, *28*(4), 361–380.

Ogata, Y. (1981). On lewis' simulation method for point processes. *IEEE transactions on information theory*, *27*(1), 23–31.

Ogata, Y. (1998). Space-time point-process models for earthquake occurrences. *Annals of the Institute of Statistical Mathematics*, *50*, 379–402.

Paddrik, M., Hayes, R., Todd, A., Yang, S., Beling, P., & Scherer, W. (2012). An agent based model of the e-mini s&p 500 applied to flash crash analysis. In *2012 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)* (pp. 1–8).

Papamakarios, G., Nalisnick, E. T., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B. (2021). Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, *22*(57), 1–64.

Potters, M., & Bouchaud, J.-P. (2003). More statistical properties of order books and price impact. *Physica A: Statistical Mechanics and its Applications*, *324*(1-2), 133–140.

Prenzel, F., Cont, R., Cucuringu, M., & Kochems, J. (2022). Dynamic calibration of order flow models with generative adversarial networks. In *Proceedings of the Third ACM International Conference on AI in Finance* (pp. 446–453).

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 888–905.

Sirignano, J., & Cont, R. (2019). Universal features of price formation in financial markets: Perspectives from deep learning. *Quantitative Finance*, *19*(9), 1449–1459.

Sirignano, J. A. (2019). Deep learning for limit order books. *Quantitative Finance*, *19*(4), 549–570.

Smith, E., Farmer, J. D., Gillemot, L. s., Krishnamurthy, S., et al. (2003). Statistical theory of the continuous double auction. *Quantitative Finance*, *3*(6), 481–514.

Smith, K. E., & Smith, A. O. (2020). Conditional GAN for timeseries generation. *arXiv preprint arXiv:2006.16477*.

Storchan, V., Vyetrenko, S., & Balch, T. (2020). Mas-gan: Adversarial calibration of multi-agent market simulators.

Takahashi, S., Chen, Y., & Tanaka-Ishii, K. (2019). Modeling financial time-series with generative adversarial networks. *Physica A: Statistical Mechanics and its Applications*, *527*, 121261.

Toth, B., Eisler, Z., Lillo, F., Kockelkoren, J., Bouchaud, J.-P., & Farmer, J. D. (2012). How does the market react to your order flow? *Quantitative Finance*, *12*(7), 1015–1024.

Van Kervel, V., & Menkveld, A. J. (2019). High-frequency trading around large institutional orders. *The Journal of Finance*, *74*(3), 1091–1137.

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, *17*(4), 395–416.

Vuletić, M., Prenzel, F., & Cucuringu, M. (2023). Fin-gan: Forecasting and classifying financial time series via generative adversarial networks. *Available at SSRN 4328302*.

Vyetrenko, S., Byrd, D., Petosa, N., Mahfouz, M., Dervovic, D., Veloso, M., & Balch, T. H. (2019). Get real: Realism metrics for robust limit order book market simulations. *arXiv preprint arXiv:1912.04941*.

Wang, X., Hoang, C., Vorobeychik, Y., & Wellman, M. P. (2021). Spoofing the limit order book: A strategic agent-based analysis. *Games*, *12*(2), 46.

Wiese, M., Bai, L., Wood, B., & Buehler, H. (2019). Deep hedging: Learning to simulate equity option markets. *Available at SSRN 3470756*.

Wiese, M., Knobloch, R., Korn, R., & Kretschmer, P. (2020). Quant GANs: Deep generation of financial time series. *Quantitative Finance*, *20*(9), 1419–1440.

Wiese, M., & Murray, P. (2022). Risk-neutral market simulation. *arXiv preprint arXiv:2202.13996*.

Wiese, M., Wood, B., Pachoud, A., Korn, R., Buehler, H., Murray, P., & Bai, L. (2021). Multi-asset spot and option market simulation. *arXiv preprint arXiv:2112.06823*.

Wright, I. D., Reimherr, M., & Liechty, J. (2022). A machine learning approach to classification for traders in financial markets. *Stat*, *11*(1), e465.

Wu, P., Rambaldi, M., Muzy, J.-F., & Bacry, E. (2019). Queue-reactive hawkes models for the order flow. *arXiv preprint arXiv:1901.08938*.

Wu, Y., Mahfouz, M., Magazzeni, D., & Veloso, M. (2021). Towards robust representation of limit orders books for deep learning models. *arXiv preprint arXiv:2110.05479*.

Zhang, Z., Lim, B., & Zohren, S. (2021). Deep learning for market by order data. *Applied Mathematical Finance*, *28*(1), 79–95.

Zhang, Z., & Zohren, S. (2021). Multi-horizon forecasting for limit order books: Novel deep learning approaches and hardware acceleration using intelligent processing units. *arXiv preprint arXiv:2105.10430*.

Zhang, Z., Zohren, S., & Roberts, S. (2019). Deeplob: Deep convolutional neural networks for limit order books. *IEEE Transactions on Signal Processing*, *67*(11), 3001–3012.

Zhou, X., Pan, Z., Hu, G., Tang, S., & Zhao, C. (2018). Stock market prediction on high-frequency data using generative adversarial nets. *Mathematical Problems in Engineering*.

Appendix A

Analysis and Modeling of Client Order Flow in Limit Order Markets

A.1 Feature table

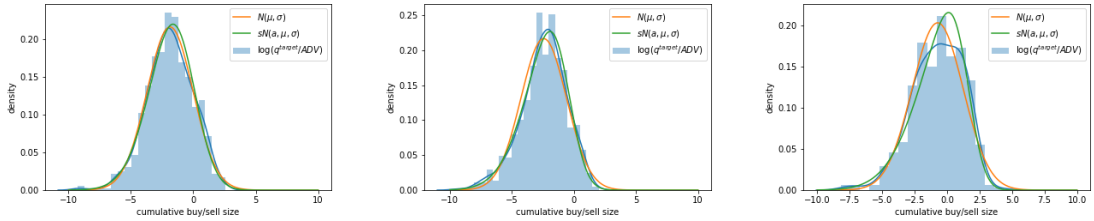
Feature Name	Explanation
Buy ratio	Pct. of client orders which is a buy order
Cancellation ratio	Pct. of client orders which are cancelled before full execution
# Trades per month	Number of trades per month.
Inventory	Mean inventory accumulation of a client on a given day
NO Limit price ratio	Percentage of the client's orders for which no limit price has been specified (maximum/minimum price for execution of child orders)
Maximum order creation time	Latest time at which a client submits an order
Mean order creation time	Average time at which a client submits an order
Mean order size (ratio of ADV)	Mean order size measured on the average daily volume of the last 20 days, execution may be less.
Mean momentum (bps)	Mean momentum of entire trading day measured in basis points (bps)
Mean percentage of volume	Mean percentage of traded volume during trade horizon (visible fills + dark fills) / (visible market volume), exceeding 100 is indicator for larger placements in dark venues
Mean volatility	Mean volatility during which a client trades measured on the last 20 days
Minimum order creation time	Earliest time a client creates an order
# Active days per month	Number of days a client trades per month
Mean # orders per active day	Number of orders a client trades if it trades during a day
Standard deviation # orders per active day	Standard deviation of the number of orders of days during which a client places at least one order
Standard deviation of order creation time	Standard deviation of a client's creation time
Standard deviation order size	Standard deviation of a clients order size
Total order size	Cumulative trade size measured on the average daily volume of the last 20 days. Indication of how much a client in average trades at all

Table A.1: Feature list used for the clustering. All features are computed on the base of a one month data set. For instance, #Trades indicates the number of trades for a particular client per month. The creation time, measured from the time passed since midnight is set to a minimum of 7 as some clients, sending their orders on the evening before disrupt the feature distribution. 7am in this case involves all orders sent before market opening.

A.2 Plots – parent order model

Additional fits for suggested distributions for a high market cap stock (left), high volume stock (middle) and low volume stock (right).

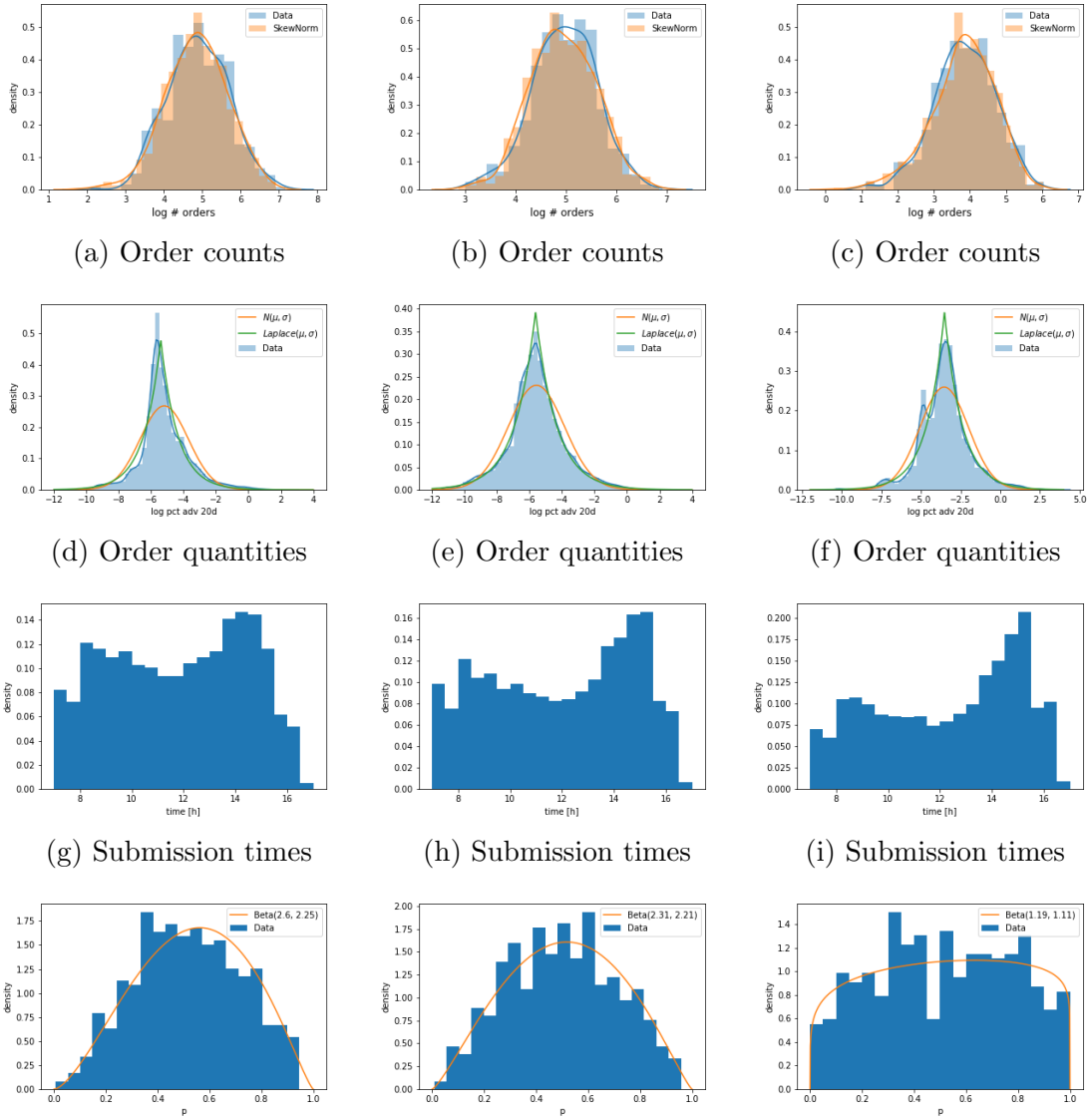
A.2.1 Day VWAP



(a) Order quantities (b) Order quantities (c) Order quantities

Figure A.1: Order quantities of the Day VWAP trader type for a high market cap stock (left), high volume stock (middle) and low volume stock (right).

A.2.2 Quant



(a) Order counts (b) Order counts (c) Order counts

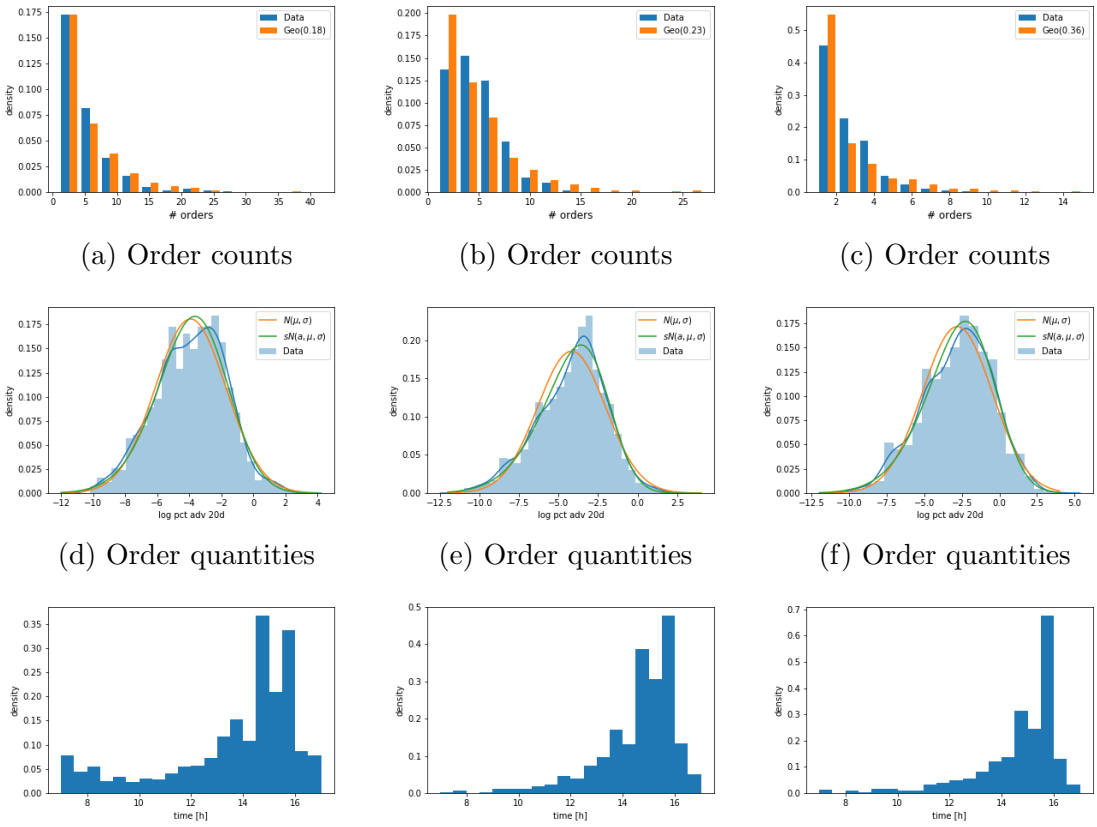
(d) Order quantities (e) Order quantities (f) Order quantities

(g) Submission times (h) Submission times (i) Submission times

(j) Fraction of buy orders (k) Fraction of buy orders (l) Fraction of buy orders

Figure A.2: Order counts, order quantities, submission time and fraction buy orders distribution of the QUANT trader type for a high market cap stock (left), high volume stock (middle) and low volume stock (right).

A.2.3 Signal



(a) Order counts (b) Order counts (c) Order counts

(d) Order quantities (e) Order quantities (f) Order quantities

(g) Submission times (h) Submission times (i) Submission times

Figure A.3: Order counts, order quantities, submission time and buy ratio distribution of the SIGNAL trader type for a high market cap stock (left), high volume stock (middle) and low volume stock (right).

Appendix B

LOB Simulation with GANs

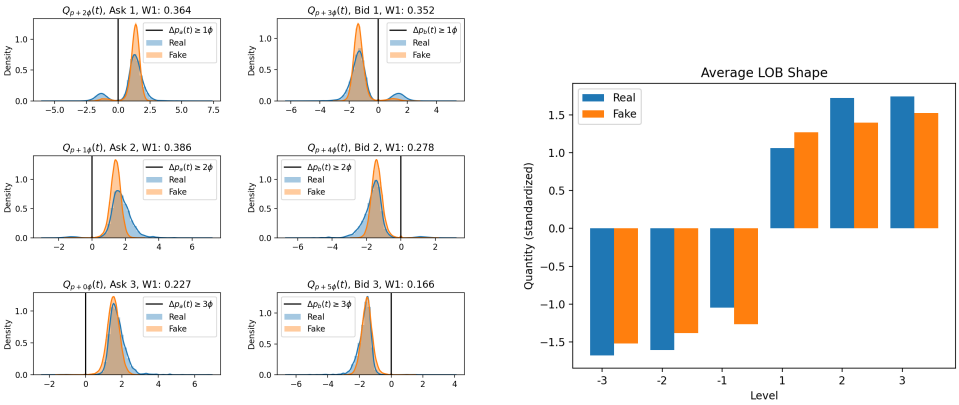
B.1 Simulation procedure

Algorithm 3 Simulator interaction – Market order execution

Require:

Simulation parameters: n_{exec} (number of executions), f (frequency), q (order size)
Initialization: $t \leftarrow 0$, Sample X_0 from $\mathcal{D}$, $N = \frac{1}{f} \cdot n_{exec}$
for $n = 1, \ldots, N$ **do**
  if $n \mod f = 0$ **then** ▷ Execute trade
    $X_t \leftarrow X_t - \Delta X_t$ ▷ ΔX_t as in Equation (4.31) and Equation (4.32)
  end if
  $Z_t \sim \mathcal{N}(0, I)$
  $X_{t+\Delta t} = g(Z_t, X_t)$
  Record price change: Δp_{mid}
  if $\Delta p_{mid} \neq 0$ **then**
    Recenter $X_{t+\Delta t}$ such that $Q_{p+(k-1)\delta}(t+1) = Q_{p_b}(t+1)$
  end if
  $t \leftarrow t + \Delta t$
end for

B.2 Poisson order flow



(a) Marginal distributions $X_{t+\Delta t}$. (b) Average LOB shape.

Figure B.1: Marginal distributions and average order book shape for both real and fake data under Poisson order flow.

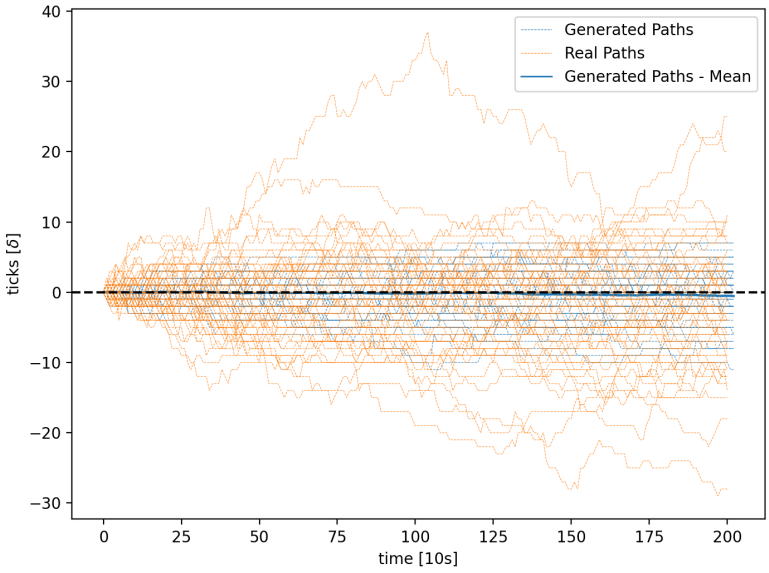


Figure B.2: Price paths for real and fake data under Poisson order flow.

B.3 Hawkes order flow

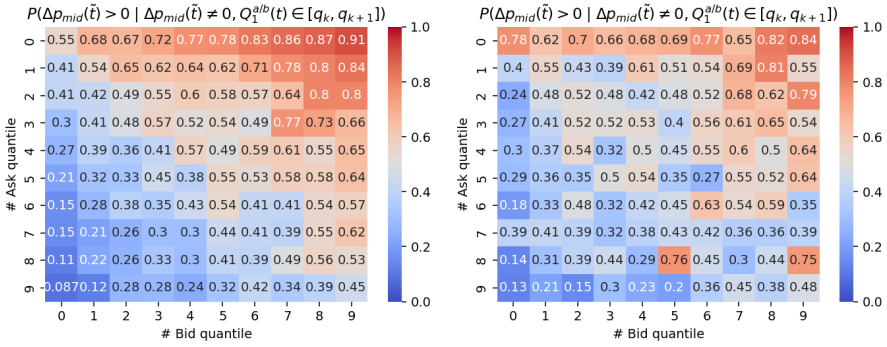


Figure B.3: Matrix indicating the probability of a positive price change given there is a price change depending on the quantiles of the best bid/best ask queue. Left: Real, right: Poisson order flow.

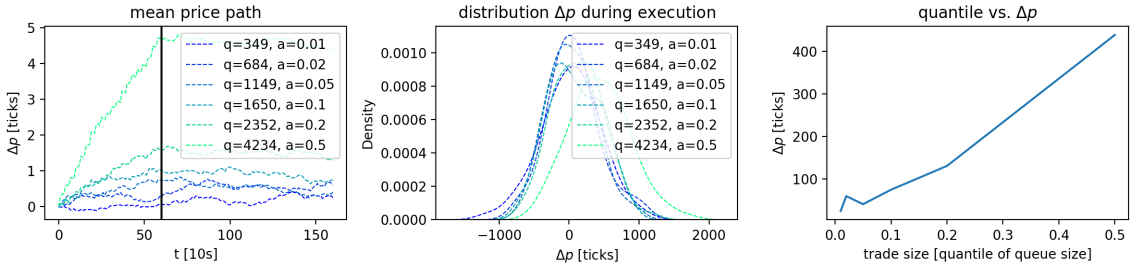
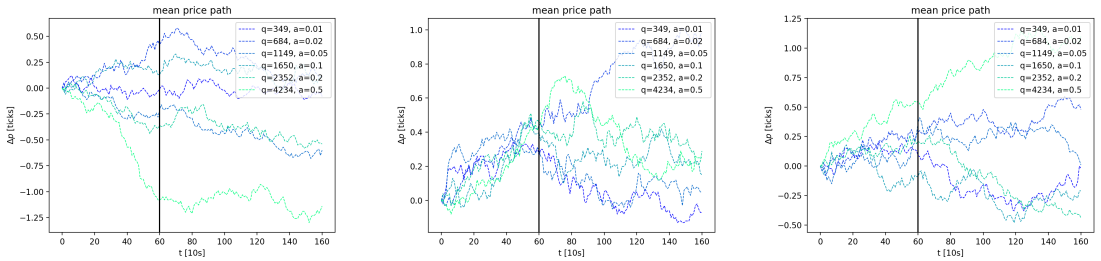
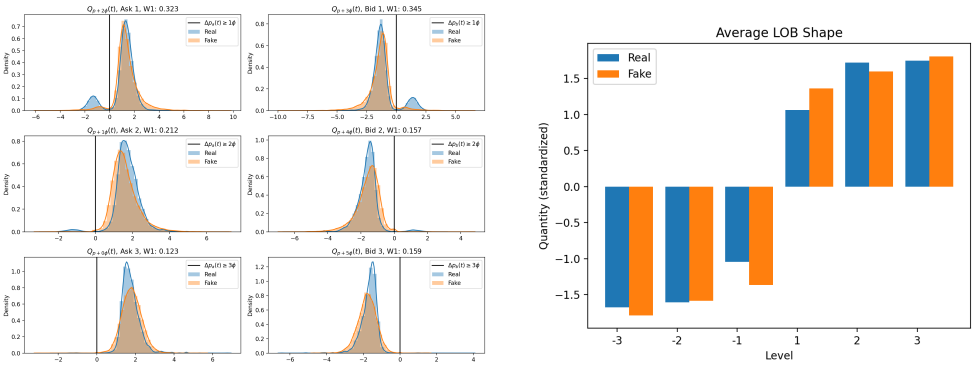


Figure B.4: Execution of a parent order with different sizes based on the empirical distribution of the queue size under Poisson order flow. The black vertical line indicates the stop of the trade.



(a) First Ask (b) First Bid (c) Second Bid

Figure B.5: Market impact of liquidity provision on both ask and bid side of the LOB under Poisson order flow. The black vertical line indicates the stop of the trade.



(a) Marginal distributions $X_{t+\Delta t}$.

(b) Average LOB shape.

Figure B.6: Marginal distributions and average order book shape for both real and fake data under Hawkes order flow.

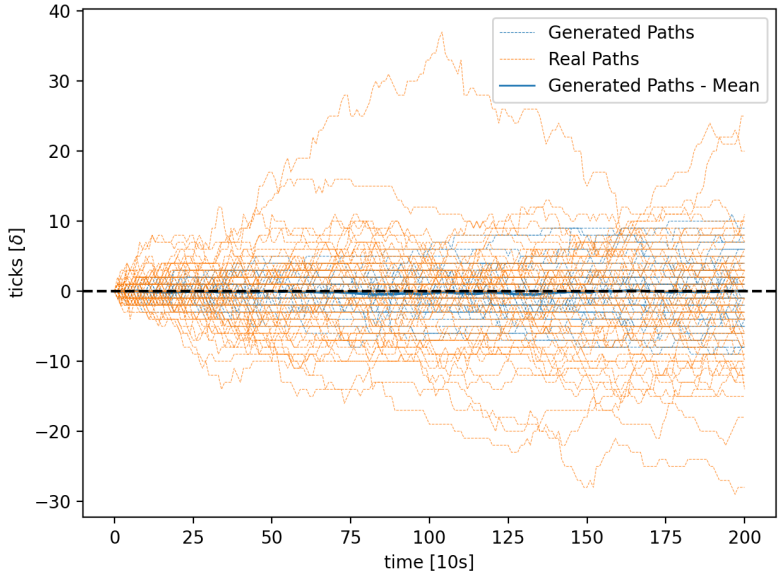


Figure B.7: Price paths and corresponding percentiles of terminal prices for real and fake data under Hawkes order flow.

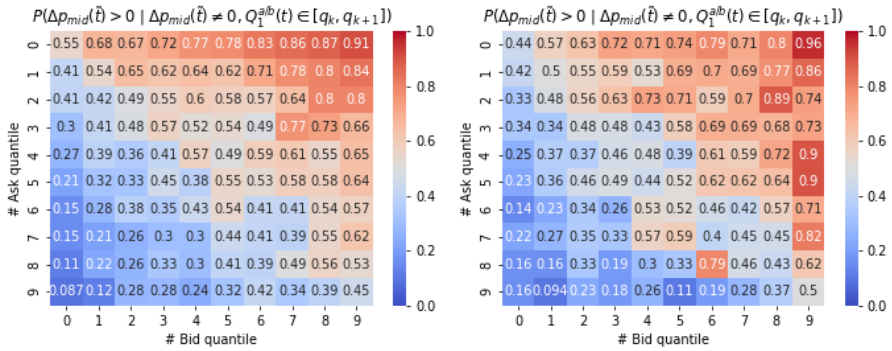


Figure B.8: Matrix indicating the probability of a positive price change given there is a price change depending on the quantiles of the best bid/best ask queue. Left: Real, right: Hawkes order flow.

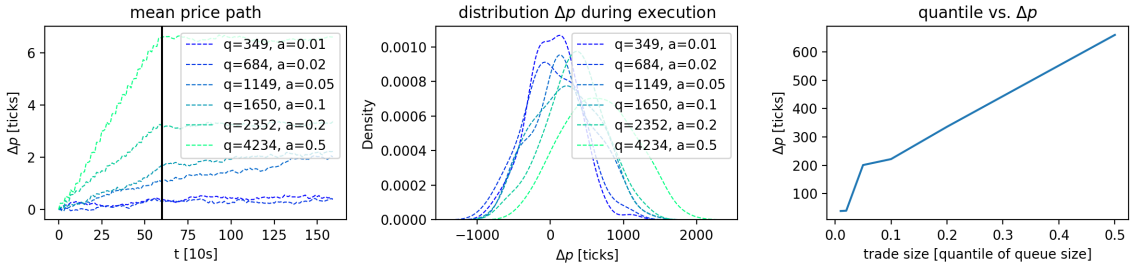
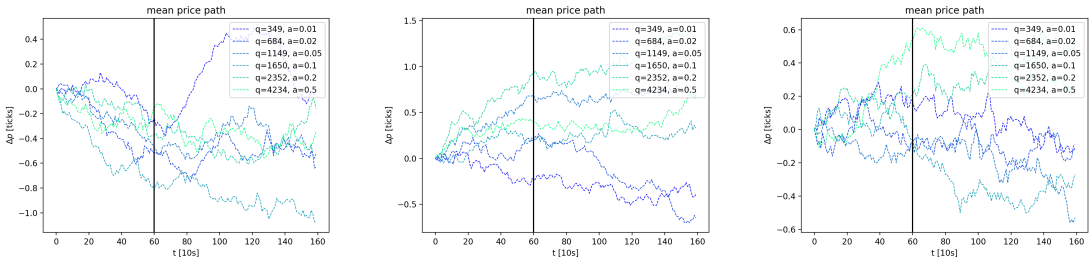


Figure B.9: Execution of a parent order with different sizes based on the empirical distribution of the queue size under Hawkes order flow. The black vertical line indicates the stop of the trade.



(a) First Ask (b) First Bid (c) Second Bid

Figure B.10: Market impact of liquidity provision on both ask and bid side of the LOB under Hawkes order flow. The black vertical line indicates the stop of the trade.

Appendix C

Disclaimer

Opinions and estimates constitute our judgment as of the date of this Material, are for informational purposes only and are subject to change without notice. This Material is not the product of J.P. Morgan's Research Department and therefore, has not been prepared in accordance with legal requirements to promote the independence of research, including but not limited to, the prohibition on the dealing ahead of the dissemination of investment research. This Material is not intended as research, a recommendation, advice, offer or solicitation for the purchase or sale of any financial product or service, or to be used in any way for evaluating the merits of participating in any transaction. It is not a research report and is not intended as such. Past performance is not indicative of future results. Please consult your own advisors regarding legal, tax, accounting or any other aspects including suitability implications for your particular circumstances. J.P. Morgan disclaims any responsibility or liability whatsoever for the quality, accuracy or completeness of the information herein, and for any reliance on, or use of this material in any way.