

Imperial College London
Department of Computing

Quantitative Modelling and Learning in Financial Markets: An Agent-based Approach

Kang Gao

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Computing of Imperial College London and
the Diploma of Imperial College, October 2024

Statement of Originality

I, Kang Gao of Imperial College London, hereby confirm that this thesis is a presentation of my independent research. Every effort is made to explicitly acknowledge contributions from others in the references. A full list of references is given in the bibliography. Generative AI tools (such as ChatGPT) are only used for checking and improving English grammar.

Copyright

The copyright of this thesis rests with the author. Unless otherwise indicated, its contents are licensed under a Creative Commons Attribution-Non Commercial 4.0 International Licence (CC BY-NC).

Under this licence, you may copy and redistribute the material in any medium or format. You may also create and distribute modified versions of the work. This is on the condition that: you credit the author and do not use it, or any derivative works, for a commercial purpose.

When reusing or sharing this work, ensure you make the licence terms clear to others by naming the licence and linking to the licence text. Where a work has been adapted, you should indicate that the work has been changed and describe those changes.

Please seek permission from the copyright holder for uses of this work that are not included in this licence or permitted under UK Copyright Law.

Abstract

This thesis proposes innovative methodologies for modelling financial markets using agent-based models (ABMs). It addresses key challenges including price formation analysis, optimal hedging strategy, and flash crash analysis.

The first contribution is the XGB-Chiarella method, which integrates the Extended Chiarella model with an XGBoost-based calibration technique for realistic financial market simulations. This method, tested across different stock exchanges, effectively replicates market dynamics. The analysis of the price formation process highlights agent-based models' capability to capture universal phenomena in financial markets. The XGB-Chiarella method establishes a new standard for financial market simulations and provides a robust workflow for calibrating ABMs.

The second contribution is the creation of the Chiarella-Heston model. This model combines the Extended Chiarella model with the Heston stochastic volatility model. It is calibrated to reproduce empirical stylised facts and is utilised to train a deep hedging agent. Tested against empirical data, the trained agent demonstrates superior performance in optimising hedging strategies compared to traditional models. The Chiarella-Heston model provides financial institutions with a robust tool to enhance risk management strategies.

The third contribution is a simulator for simulating high-frequency trading environments and risk scenarios, such as flash crashes. Building on the XGB-Chiarella method, this simulator operates at sub-second intervals with full exchange protocols. It offers a comprehensive tool for analysing the dynamics and conditions leading to flash crashes. The simulator also provides insights into factors that influence the severity of such risk scenarios, including the market maker inventory limits and trading frequency for different types of traders. The simulator also examines mini-flash crash events, enhancing our understanding of high-frequency trading risks.

Overall, this thesis demonstrates the effectiveness of an agent-based approach in modelling financial markets, bridging the gap between theory and practice. It provides a foundation for future applications in risk management and policy evaluation.

Acknowledgements

I am deeply thankful to my supervisors, Prof. Wayne Luk and Prof. Stephen Weston, for their invaluable guidance, support, and motivation during my research. Prof. Luk has played a crucial role in my personal and professional development, providing critical insights into research methodologies, teamwork, and scholarly writing, which have significantly contributed to my growth. His belief in my potential has profoundly shaped my journey. Prof. Weston's visionary leadership made this project feasible, uniting individuals from diverse backgrounds to work towards a shared objective. His vast expertise in agent-based modelling, machine learning, and quantitative finance has enriched my learning immensely. Both Prof. Luk and Prof. Weston have shown unwavering support, encouragement, and patience throughout my PhD studies.

I would also like to thank Dr. Ce Guo, whose invaluable support significantly enhanced my research experience. Whenever I encountered an issue, I could always rely on Dr. Guo for insightful suggestions. His assistance greatly facilitated my research journey.

I was fortunate to have the possibility to collaborate with Simudyne, where I got invaluable assistance. In particular, I extend my sincere gratitude to Dr. Perukrishnen Vytelingum. Krishnen consistently offered help and support, and he created a unique opportunity for me to integrate my research at Imperial College with my work at Simudyne. He was instrumental in navigating challenges, giving advice, providing feedback on my papers, and providing invaluable assistance over the years. I also extend my thanks to Dr. Namid Stillman for his support in my research on reinforcement learning. Additionally, I also thank Justin Lyon and Jiyan Babaie-Harmon for their help during my time at Simudyne. Furthermore, I am grateful to Kaveh Asaraai and Javier A. Varela for their excellent suggestions, support and encouragement. I also wish to express my gratitude to Francois Buet-Golfouse for generously sharing his profound insights on reinforcement learning in Finance. My conversations with Francois have been invaluable and significantly enriched my journey.

I am thankful for the opportunity to have studied in the Custom Computing Group at the Department of Computing, Imperial College London. I am also deeply appreciative of all my colleagues at Imperial, particularly the members of the Custom Computing group, with whom I had the privilege of working closely. Their invaluable discussions and collaboration greatly enriched my experience.

I would like to thank all the people I met at Imperial College, including my office family and friends within the department. Their presence significantly enriched my time spent in the Huxley Building, making it truly unforgettable. The camaraderie, support, and shared experiences with them not only enhanced my academic journey but also provided joy and comfort during challenging times. Each

moment spent together has left a lasting impression, contributing to a vibrant and supportive community that I was fortunate to be a part of.

Finally, I would like to extend my warmest thanks to my parents and my girlfriend Lijing, for their unwavering support throughout the years. Their love, encouragement, and belief in me have been my constant source of strength and motivation. Their presence in my life has been an invaluable gift for which I am deeply grateful.

‘If you no longer go for a gap that exists, you are no longer a racing driver’

Ayrton Senna

Contents

Statement of Originality	1
Copyright	3
Abstract	5
Acknowledgements	7
1 Introduction	25
1.1 Motivation	25
1.2 Challenges and Contributions	26
1.2.1 First Challenge and Contribution: Price Formation Process Investigation	27
1.2.2 Second Challenge and Contribution: Optimal Hedging Strategy Development	28
1.2.3 Third Challenge and Contribution: High-frequency Financial Market Simulation and Flash Crash Scenario Analysis	30
1.3 Connections Between the Three Main Contributions	32
1.4 Publications	33
1.5 Thesis Organisation	34
1.6 About this journey	34

2	Background and Related Work	36
2.1	Introduction	36
2.2	Agent-based Financial Market Simulation	36
2.2.1	Agent-based Models	36
2.2.2	XGBoost Algorithm for Agent-based Model Calibration	37
2.2.3	Agent-based Financial Market Simulation	38
2.2.4	Agent-based Models and Simulation in This Thesis	41
2.3	Price Formation Process	41
2.3.1	Financial Markets	41
2.3.2	Existing Price Formation Process Research	43
2.3.3	Existing Gaps and Our Contributions	45
2.4	Deep Reinforcement Learning and Deep Hedging	45
2.4.1	Reinforcement Learning	45
2.4.2	Hedging	47
2.4.3	Combining Reinforcement Learning and Hedging with Deep Learning Techniques	47
2.4.4	Existing Gaps and Our Contributions	48
2.5	Flash Crash Analysis	49
2.5.1	Review of Existing Flash Crash Research	49
2.5.2	Existing Gaps and Our Contributions	51
2.6	Summary	52
3	Price formation process investigation with agent-based financial market simulation	53
3.1	Introduction	53
3.1.1	Motivation	53
3.1.2	Our Contributions	56

3.2	Model Architecture	57
3.2.1	Model Set-up	57
3.2.2	Fundamental Trader	57
3.2.3	Momentum Trader	58
3.2.4	Noise Trader	59
3.2.5	Model Dynamics	59
3.3	Model Calibration	60
3.3.1	Data	61
3.3.2	Stylised Facts and Loss Function	61
3.3.3	Surrogate Modelling Calibration Method	67
3.4	Results and Evaluation	72
3.4.1	Stylised Facts Distance	72
3.4.2	Surrogate Model Performance	73
3.4.3	Autocorrelations	76
3.5	Summary	77
4	Chiarella-Heston model for optimal deep hedging	80
4.1	Introduction	80
4.1.1	Motivation	80
4.1.2	Our Contributions	82
4.2	Proposed Chiarella-Heston Model	83
4.2.1	Model Setup	83
4.2.2	Relationship with Existing Models	88
4.2.3	Model Calibration	89
4.2.4	Model Validation	92

4.3	Deep Hedging Experiments	93
4.3.1	Problem Setting	94
4.3.2	Reinforcement Learning Algorithm	95
4.3.3	Deep Hedging Experimental Results	96
4.4	Summary	97
5	High-frequency financial market simulation and flash crash analysis	99
5.1	Introduction	99
5.1.1	Motivation	99
5.1.2	Our Contributions	102
5.2	Model Structure	104
5.2.1	Model Set-up	104
5.2.2	Price Formation Process	104
5.2.3	Common Trader Behaviours	105
5.2.4	Fundamental Trader (FT)	107
5.2.5	Momentum Trader (MT)	108
5.2.6	Noise Trader (NT)	111
5.2.7	Market Maker (MM)	112
5.2.8	Experimental Setup for Simulation	114
5.3	Model Calibration and Validation	116
5.3.1	Calibration Target: Data and Stylised Facts for Realistic Simulation	116
5.3.2	Calibration Workflow and Results	122
5.3.3	Model Validation	123
5.4	2010 Flash Crash Scenario	127
5.4.1	Simulating Historical Flash Crash	128

5.4.2	Flash Crash Under Different Conditions	133
5.4.3	Simulating Other Flash Crash Events	139
5.4.4	Alternative Explanations for Flash Crash Events	139
5.5	Mini Flash Crash Scenarios	140
5.5.1	Introduction of Spiking Trader (ST)	140
5.5.2	Mini Flash Crash Analysis	141
5.5.3	Conditions for Mini Flash Crash Scenarios	142
5.5.4	Comparison Between Large and Mini Flash Crash Events	145
5.6	Summary	146
6	Conclusion	148
6.1	Summary of Thesis Achievements	148
6.2	Future Work	150
6.2.1	Robust calibration and validation techniques	150
6.2.2	More effective deep hedging strategy	151
6.2.3	Prediction of flash crash events	151
6.2.4	Multi-agent reinforcement learning and multi-agent simulation	152
6.3	Final Thoughts	153
	Bibliography	154
	Appendices	165

List of Tables

3.1	Specific stocks from three exchanges in the dataset.	61
3.2	Average excess kurtosis for return distributions of five stocks.	62
3.3	Average stylised facts distance comparison between XGB-Chiarella calibration and Expectation-Maximisation method. The stylised facts distance is the average across all trading days for each stock; values in parentheses are corresponding standard deviations. There are several rare cases where distances are much larger, showing algorithms fail to obtain a reasonable set of parameters during the parameter estimation process.	73
3.4	Percentage of trading days when XGB-Chiarella calibration outperforms Expectation-Maximisation method.	74
3.5	Comparison between surrogate model predicted and actual stylised facts distance at the surrogate predicted optimal point. The results shown here are for one trading day (2nd of November, 2021), while results for other trading days are very similar.	75
4.1	Comparison of stylised facts distance among different models after calibration	92
4.2	Statistical test between two sets of GSL-div values	93
4.3	Hedging performance - P&L (\$)	97
4.4	Hedging performance - Expected Shortfall (%)	97
5.1	Caption for LOF	102
5.2	Order types submitted by market makers under different trading conditions	112

5.3	Hill Estimator values for all trading days	117
5.4	Calibrated values for the seven model parameters and the yielded stylised facts distances	123
5.5	P-values for the calibrated model for all trading days	125
5.6	Moment coverage ratios of the calibrated model	127
6.1	Descriptions for all parameters involved in the proposed agent-based model in Chapter 5.	165
6.2	Values for fixed model parameters in the calibration process in Chapter 5.	166
6.3	Values for model parameters in simulating the 2010 Flash Crash in Chapter 5.	167
6.4	Values for model parameters in simulating the example Mini Flash Crash in Chapter 5.	168

List of Figures

2.1	Adapted from [GPW ⁺ 13], an illustration of a limit order book.	42
2.2	Components and flows of the RL framework. Adapted from [PAD22].	46
3.1	Cumulative distribution function of historical returns in two stocks (FB, RIO.L) for one trading day, in comparison with a normal distribution reference.	62
3.2	Autocorrelation patterns for returns of three stocks from the three exchanges.	63
3.3	Autocorrelation patterns for squared returns of three stocks from the three exchanges.	64
3.4	Sensitivity analysis around the optimal point predicted by the XGBoost surrogate model. This surrogate model corresponds to the trading day on the 29th of September, 2021 for stock FB. For each sub-graph, model parameters are fixed to the optimal value except for the parameter in x-axis. Gray lines represent the actual stylised facts distances while blue lines represent the XGBoost surrogate model prediction for the stylised facts distances.	75
3.5	(a): Autocorrelation of squared returns generated by a GBM for CMCSA on Sept 28th, 2021. (b): Autocorrelation of squared returns generated by XGB-Chiarella method for CMCSA on Sept 28th, 2021. (c): Historical autocorrelation of squared returns for CMCSA on Sept 28th, 2021.	76
3.6	Simulated price scenarios generated by GBM and XGB-Chiarella, respectively.	77
3.7	Average autocorrelation of squared returns associated with the simulated price scenarios in Figure 3.6. Blue line represents the GBM simulated scenarios while red dashed line represents the XGB-Chiarella simulated scenarios.	78
4.1	Historical stylised facts	90

4.2	Chiarella-Heston model simulated stylised facts after calibration	92
4.3	P&L distribution of three hedging strategies	97
5.1	Autocorrelation function of second-level returns for E-mini S&P 500 futures on two days	118
5.2	Autocorrelation function of squared second-level returns for E-mini S&P 500 futures on two days	118
5.3	Comparison between empirical mid-price time series on May 5th and simulated mid-price time series on May 5th	124
5.4	One single simulation of the 2010 Flash Crash event. Red line is the historical price; green line is the simulated price; blue line is the fundamental value extracted by Kalman Smoother	130
5.5	Comparison between simulated volume and historical volume on May 6th, 2010	130
5.6	(a): Total inventory level for each type of trader, around the simulated flash crash event (14:00 - 15:15). (b): The volume change for the selling orders of the institutional trader, during the time when the institutional trader is active	133
5.7	Detailed simulated mid-price against the inventories of each category of trader from 14:30 to 14:50. (a): Simulated and historical price from 14:30 to 14:50. (b): Total inventory level for each type of trader during period of time from 14:30 to 14:50	134
5.9	(a): Relationship between the amplitude of simulated flash crashes and the inventory limit of market makers. (b): Relationship between the amplitude of simulated flash crashes and the percentage of volume r for the Sell Algorithm (Institutional Trader). (c): Relationship between the amplitude of simulated flash crashes and the trading frequency of fundamental traders.	136
5.10	One typical simulated mini flash crash event. Red line is the historical price; green line is the simulated price; blue line is the fundamental value	143
5.11	Detailed simulated price against the inventories of each type of trader during the simulated mini flash crash event. (a): Simulated and historical price from 12:08:30 to 12:09:00. (b): Total inventory level for each type of trader during period of time from 12:08:30 to 12:09:00	143

5.12	(a): Relationship between the average number of mini flash crashes per simulation and the inventory limit of market maker. (b): Relationship between the average number of mini flash crashes per simulation and the trading frequency of fundamental traders.	145
5.13	(a): Relationship between the average amplitude of simulated mini flash crashes and the inventory limit of market maker. (b): Relationship between the average amplitude of simulated mini flash crashes and the trading frequency of fundamental traders.	146
6.1	The VPIN measure before the simulated 2010 Flash Crash	152

Acronyms

A2C Advantage Actor Critic.

ABM Agent-based Model.

CFTC Commodity Futures Trading Commission.

DDPG Deep deterministic policy Gradient.

DL Deep Learning.

DQN Deep Q-Network.

DRL Deep Reinforcement Learning.

EM Expectation–Maximization.

GAN Generative Adversarial Network.

GSL-div Generalised Subtracted L-divergence.

HKEX Hong Kong Exchanges and Clearing Limited.

K-S Kolmogorov-Smirnov.

LOB Limit Order Book.

LSE London Stock Exchange.

LSTM Long Short-Term Memory.

MARL Multi-agent Reinforcement Learning.

MDP Markov Decision Process.

ML Machine Learning.

MSM Method of Simulated Moments.

PPO Proximal Policy Optimization.

RDPG Recurrent Deterministic Policy Gradient.

RL Reinforcement Learning.

RNN Recurrent Neural Network.

SEC U.S. Securities and Exchange Commission.

VPIN Volume-Synchronized Probability of Informed Trading.

Chapter 1

Introduction

1.1 Motivation

Modelling is widely used across various fields to understand system behaviours, focusing on essential factors and eliminating unnecessary details. These models can range from simple thought experiments to complex mathematical simulations, particularly in quantitative disciplines. They help researchers predict outcomes in hypothetical scenarios, especially when real-world data is limited or experiments are impractical. Effective models must accurately replicate real-world observations, be user-friendly, and provide simple explanations.

Most financial market models aim to identify the fundamental causes behind market behaviours, ignoring irrelevant details and concentrating on the factors crucial to the hypothesis, question, or policy [SH97]. While some models are basic thought experiments, those in finance often involve sophisticated mathematical techniques and simulations. Creating a simplified and artificial representation of financial markets enables researchers and policymakers to explore potential outcomes under different scenarios. In financial markets, data can be limited, and real-world experiments are rarely feasible. This characteristic makes financial market models especially valuable. Different financial market models are designed to address specific questions. All those models should be able to replicate real-world phenomena as accurately as possible. It is also beneficial if models are easily interpretable and capable of explaining the underlying causes of these phenomena.

In recent years, agent-based models (ABMs) have gained significant popularity in modelling financial markets. These models simulate the actions of individual traders as well as interactions between different

kinds of traders, aiming to explain system-level market behaviours. The strength of ABMs lies in their ability to demonstrate how simple, individual behaviours can combine to produce the complex behaviours observed in real-world systems [Jan05]. ABMs offer a more granular and dynamic understanding of market behaviours than traditional equilibrium-based models, capturing the complexities and emergent phenomena that arise from the diverse strategies and decisions of market participants. Unlike traditional models that often rely on aggregate equations and assumptions of rational behaviour, agent-based models can incorporate heterogeneous behaviours, adaptive learning, and influence between different traders. This makes them particularly useful for exploring financial market dynamics, assessing the impact of regulations, and predicting potential market crises. The increasing use of ABMs reflects a shift towards more realistic and detailed representations of financial systems.

Various agent-based models have been developed to simulate financial markets, including the Zero-Intelligence Model [GS93], the Extended Chiarella Model [MCB20], and the Low- and High-frequency Trading Model [JL NRF16]. Despite these advances, creating highly realistic agent-based simulations of financial markets that can address practical issues remains challenging. First, there is a significant gap between the current models in the literature and the ideal high-fidelity agent-based models for financial market simulation. Second, effectively calibrating and validating these models continues to be a focus of active research. Third, addressing practical challenges in financial markets with an agent-based modelling approach is a relatively new research area. This thesis contributes to the above fields by utilising agent-based models to address three practical problems in financial markets. The challenges and corresponding contributions in this thesis are presented below.

1.2 Challenges and Contributions

Building on the motivations outlined above, this thesis aims to model financial markets using agent-based models and address the practical problems in an agent-based modelling framework. This process involves developing, calibrating, and validating agent-based models. These models are utilised to address three specific challenges. First, we examine the intraday price formation process within agent-based models to better understand market dynamics at intraday time scales. Second, we employ deep reinforcement learning to tackle the optimal hedging problem, utilising agent-based models to optimise hedging strategies. Third, we construct a realistic high-frequency financial market simulation using agent-based models to analyze risk scenarios such as flash crashes. The following three subsections elaborate on the

challenges faced and the contributions made in each topic.

1.2.1 First Challenge and Contribution: Price Formation Process Investigation

The first topic in this thesis involves the price formation process. Over the past decade, algorithmic trading has significantly expanded across the world, becoming the predominant method for trading in financial markets [NMTY11]. This growth has been driven by continuous advancements in computing technology and the evolution of market micro-structures, which have automated many aspects of trading such as price quotation and order execution. Consequently, intraday price formation, which represents the dynamics of how prices are set through the interactions between buyers and sellers, has attracted increased attention. Exploring these dynamics has become a popular research area aimed at enhancing trading strategies and improving overall market efficiency.

Research on intraday price formation is increasingly conducted through agent-based simulations, which provide a dynamic representation of financial markets by incorporating heterogeneous traders and realistic trading mechanisms. Unlike traditional economic models, these simulations do not assume equilibrium or rational behaviour, offering a more detailed view of financial market behaviours [Jan05]. High-frequency data is essential for capturing the rapid fluctuations of intraday trading, and the models must be carefully calibrated to closely mimic real-world data. The realism is reflected by reproducing universally observed market behaviours known as stylised facts, such as volatility clustering and fat tails in return distributions [Con01]. Achieving the ideal simulation that effectively reproduces these complex market dynamics remains challenging, especially due to the vast parameter space and the complexity involved in designing an effective calibration strategy. Specifically, there are two **challenges** involved:

- C1.1: How to implement an agent-based financial market simulator which allows for the investigation of the intraday price formation process in financial markets.
- C1.2: How to calibrate the agent-based financial market simulator to ensure realism and reproduce common stylised facts.

To address these challenges, the XGB-Chiarella method has been developed. This method integrates the Extended Chiarella model [MCB20] with a novel calibration approach that utilises the XGBoost algorithm and surrogate modelling approach. This innovative method enables the generation of

realistic intraday financial market data. Our extensive testing across a broad spectrum of stocks from major global exchanges—including the Nasdaq, London Stock Exchange (LSE), and Hong Kong Stock Exchange (HKEX) has demonstrated its effectiveness. In addition, we demonstrate the existence of a universal process of intraday price formation, characterized by trends and value effects. The main **contributions** are:

- Addressing challenge C1.1: Originally validated with monthly data, the Extended Chiarella model has now been extended and rigorously tested using minute-by-minute intraday data to analyze the intraday price formation process. This extensive testing covered more than 75 stocks from three of the world’s major stock exchanges: Nasdaq, LSE, and HKEX. The consistent results across different markets indicate that trends and value effects in price formation are universal phenomena, irrespective of the exchange. Details of this contribution can be found in Section 3.2 and Section 3.4.
- Addressing challenge C1.2: Moving beyond the traditional Expectation-Maximization algorithm, we have adopted a novel machine learning surrogate modelling method to calibrate the Extended Chiarella model, addressing challenge C1.2. This approach uses an XGBoost regressor, which innovatively predicts the difference between the simulated stylised facts and their historical counterparts. This prediction process not only considers the distribution of returns but also the autocorrelations between returns and squared returns. Additionally, an exploration-exploitation mechanism is employed during the iterative parameter space selection, significantly enhancing the calibration performance. Compared to the Expectation-Maximisation baseline, the proposed approach reduces stylised facts distance by 10%, demonstrating superior performance in replicating empirical stylised facts. Details of this contribution are presented in Section 3.3.

The **novelty** of our approach lies in several aspects. First, we adapt the Extended Chiarella model to generate intraday financial time series. Second, the model is extensively tested on different stocks across three major stock exchanges around the world. Third, the exploration-exploitation mechanism is innovatively introduced to the model calibration process, enhancing the calibration performance.

1.2.2 Second Challenge and Contribution: Optimal Hedging Strategy Development

Addressing the first challenge yields an agent-based model for financial market simulation, as well as a calibration methodology. A natural next step is how to utilise and extend this methodology to solve

real-world problems. The second part of this thesis proposes an extension of the previous model and applies the extension to the optimal hedging problem.

In recent years, the quantitative finance industry has significantly embraced data science, utilising vast data volumes and increased computing power to deploy complex algorithms for practical applications like hedging derivatives [HLJB19]. The Black-Scholes model is widely recognised as the standard framework for pricing and hedging derivatives. However, it relies on several strict assumptions that are inconsistent with real-world financial markets, such as the absence of transaction costs and market slippage [Jan18]. Consequently, financial institutions are increasingly turning to deep learning techniques for hedging derivative portfolios instead of traditional methods based on the Black-Scholes model. Deep Hedging [HLJB19] has been a popular research topic for the past few years. This approach uses deep reinforcement learning methods for optimising hedging strategies. Specific risk measures are integrated into the reward function for the reinforcement learning agent. However, the deep hedging approach relies heavily on synthetic data from models like the Heston stochastic volatility model and Geometric Brownian Motion, which often fail to reflect real-world market dynamics. Agent-based models can be utilised to address these shortcomings and improve the efficacy of deep hedging in practical environments. Specifically, the main **challenge** to be addressed is:

- C2: How to develop and calibrate an agent-based model to generate realistic financial time series data, specifically tailored for optimising the performance of a deep hedging agent in real-world market conditions.

We tackle this challenge by proposing an innovative agent-based model called the Chiarella-Heston model. This model is calibrated to reproduce stylised facts exhibited by empirical data. It generates realistic financial time series data that are utilised to train a deep hedging agent. To demonstrate the practical utility of our approach, we conduct tests on the trained deep hedging agent using empirical data. The main **contributions** of the second part of this thesis are as follows:

- We propose a novel agent-based model named the Chiarella-Heston model, which includes momentum traders, fundamental traders, and volatility traders. This model is an extension of both the Extended Chiarella model in [MCB20] and the Heston stochastic volatility model in [Hes93]. It is calibrated to reflect empirical stylised facts such as the fat tails of returns and the autocorrelation patterns of returns. Compared to baseline models such as the extended Chiarella

model and the Heston stochastic volatility model, the Chiarella-Heston model reduces the stylised facts distance by more than 50%, demonstrating its ability to generate more realistic financial time series. This conclusion is further supported by established validation metrics. Details of this contribution are presented in Section 4.2.

- Utilising the calibrated agent-based financial market simulator, a training dataset is generated to train a deep hedging agent for optimal hedging strategies under different transaction cost levels. The deep hedging agent employs the Deep Deterministic Policy Gradient algorithm [LHP⁺15] and is trained to maximize profits and minimize risks. At realistic levels of transaction costs, the deep hedging agent, trained with data from the Chiarella-Heston model, outperforms the baseline, where the agent is trained with data generated by Geometric Brownian Motion. In terms of risk management, the proposed approach reduces the average expected shortfall from -3.93% to -2.74%, representing a 30% improvement. Additionally, our testing process is carried out using empirical data, showing that the trained deep hedging agent is able to perform well in realistic trading environments. Details of this contribution are presented in Section 4.3.

The **novelty** of this methodology lies in the following aspects. First, the Chiarella-Heston model is a novel extension of two classic models: the Extended Chiarella model and the Heston stochastic volatility model. Second, for the first time in literature, an agent-based model is employed for enhancing the performance of deep hedging practice. Third, unlike most works in literature that test the hedging performance with synthetic data, real-world data are used in our testing experiments. We have made the code for the Chiarella-Heston model and the associated hedging methodology publicly available as open-source.¹

1.2.3 Third Challenge and Contribution: High-frequency Financial Market Simulation and Flash Crash Scenario Analysis

The first and second parts of this thesis introduce methodologies for agent-based financial market simulation and explore the optimal hedging problem. A common feature of these contributions is their focus on the low- to mid-frequency spectra of financial markets. The third part of this thesis advances the study of high-frequency market dynamics, addressing issues specific to high-frequency trading.

The rapid evolution of electronic financial markets and the standardisation of the centralised limit

¹https://github.com/Ferrari1996gk/Deeper_Hedging

order book have greatly enhanced liquidity, narrowed bid-ask spreads, and accelerated transaction times [TD20]. Alongside these advancements, algorithmic trading, particularly high-frequency trading (HFT), has seen substantial growth. This shift has underscored the importance of speed in trading, as evidenced by events like the 2010 Flash Crash [KKST17]. The 2010 Flash crash caused dramatic disruption in global financial markets, highlighting the systemic vulnerabilities associated with HFT. Additionally, the occurrence of smaller, yet frequent, "mini flash crashes" further emphasises the need for deeper understanding and improved monitoring of these rapid market movements [GKP12].

In light of these developments, agent-based models have become an essential tool for analysing such swift market dynamics, especially the interactions between traders during the short duration of these events. ABMs enable the simulation of complex trader interactions and provide valuable insights into the conditions that lead to risk scenarios such as flash crashes. Nevertheless, existing simulations lack the frequency and realism necessary to accurately represent high-frequency trading environments. Given the above context, there are two challenges to be addressed:

- C3.1: How to develop and refine a high-frequency agent-based financial market simulator that fully integrates exchange protocols and includes a rigorous calibration and validation process to realistically replicate high-frequency financial market dynamics.
- C3.2: Using the proposed agent-based financial market simulator, investigate the market dynamics during flash crashes (including both large flash crashes and mini flash crashes) and identify the conditions that influence the characteristics of flash crashes.

To address the above challenges, we developed a novel high-frequency financial market simulator to bridge the existing gaps. The simulator is employed to explore the dynamics during flash crashes and the conditions that affect flash crashes. We have the following **contributions**:

- A high-frequency simulation capable of sub-second (100 milliseconds) time steps is developed, incorporating various types of traders and full exchange protocols to mimic real-world order matching processes. The step size of 100 milliseconds is selected to sufficiently capture the dynamics of empirical price time series without significantly compromising computational efficiency². This setup is validated through statistical tests and proves effective in replicating realistic stylised

²Detailed explanation of this choice is presented in Section 5.1.

facts in financial markets, highlighting the simulator's capacity to capture high-frequency market dynamics accurately. Details of this contribution are presented in Section 5.2 and Section 5.3.

- Using this simulator, the 2010 Flash Crash is realistically reproduced by simulating an institutional trader based on the real-world Sell Algorithm used during the actual event. Detailed analysis of this simulation revealed how different factors, such as the percentage of volume of the Sell Algorithm, market maker inventory limits, and the trading frequency of different traders, influence the amplitude of flash crashes. The analysis provides insights into the complex dynamics of such market events. Details of this contribution are presented in Section 5.4.
- The simulator is also adapted to study mini flash crash events by introducing a "Spiking Trader" to trigger these crashes, allowing for the analysis of conditions affecting their characteristics. Findings indicate that the market maker inventory limit significantly impacts the frequency and amplitude of mini flash crash events. In contrast, the trading frequency of certain types of traders does not have notable effects on the outcome. These findings highlight specific factors that drive market instability during mini flash crashes. We also present a comparison between large and mini flash crash events, specifying the difference between these two types of risk scenarios. Details of this contribution are presented in Section 5.5.

Three key aspects highlight the **novelty** of the proposed high-frequency simulator and flash crash analysis. First, this research is among the very few that address the challenge of simulating high-frequency financial markets, achieving simulations at the sub-second level. Second, to the best of our knowledge, this work is the only one that simulates the 2010 Flash Crash event with sub-second granularity, providing a detailed phase-by-phase analysis of market dynamics. Third, we simulate and analyse mini flash crashes, a phenomenon that is rarely explored in the literature.

1.3 Connections Between the Three Main Contributions

The three contributions of this thesis are closely related.

In Chapter 3, the first challenge is addressed through the introduction of the XGB-Chiarella method, which is a versatile agent-based model for generating synthetic financial time series data, coupled with a robust calibration technique. In Chapter 4, the Chiarella-Heston model is proposed, which is based on

the same underlying model of the XGB-Chiarella method in Chapter 3. Furthermore, the calibration methodology from Chapter 3 is applied to the calibration of the Chiarella-Heston model in Chapter 4. Both agent-based models in Chapters 3 and 4 are designed to realistically simulate financial markets, addressing challenges related to price formation processes and optimal hedging strategies. The model in Chapter 3 operates on a minute-level timescale, while the model in Chapter 4 simulates daily price time series. However, modern financial markets now operate at sub-second intervals, with high-frequency trading occurring in microseconds or even nanoseconds. To encompass the entire spectrum of trading activities in financial markets, Chapter 5 explores the domain of high-frequency trading. Building on the model from Chapter 3, an extended model is proposed that operates at sub-second intervals and incorporates comprehensive exchange protocols, including limit order books for transaction matching. The agent-based model in Chapter 5 is again calibrated using the calibration method proposed in Chapter 3.

1.4 Publications

The following publication contributes to Chapter 3:

- **Kang Gao**, Perukrishnen Vytelingum, Stephen Weston, Wayne Luk and Ce Guo. 2022. Understanding intraday price formation process by agent-based financial market simulation: calibrating the extended chiarella model. *Wilmott*, vol. 2022, iss. 119, p. 22–38, 2022. <http://dx.doi.org/10.54946/wilm.11014>

The following publication contributes to Chapter 4:

- **Kang Gao**, Stephen Weston, Perukrishnen Vytelingum, Namid Stillman, Wayne Luk, and Ce Guo. 2023. Deeper Hedging: A New Agent-based Model for Effective Deep Hedging. In *Proceedings of the Fourth ACM International Conference on AI in Finance (ICAIF '23)*. Association for Computing Machinery, New York, NY, USA, 270–278. <https://doi.org/10.1145/3604237.3626913>

The following publication contributes to Chapter 5:

- **Kang Gao**, Perukrishnen Vytelingum, Stephen Weston, Wayne Luk and Ce Guo. 2024. High-Frequency Financial Market Simulation and Flash Crash Scenarios Analysis: An Agent-

Based Modelling Approach. Journal of Artificial Societies and Social Simulation 27 (2) 8
<https://www.jasss.org/27/2/8.html>

1.5 Thesis Organisation

The remaining chapters of this thesis are organised as follows:

- Chapter 2 provides the basic background knowledge of the mechanisms of the modern financial market, agent-based modelling and reinforcement learning techniques. A comprehensive review of existing literature relevant to this thesis is also presented in this chapter.
- Chapter 3 details the development and implementation of the XGB-Chiarella methodology, including the model architecture, calibration techniques, and evaluation metrics. The intraday price formation process is investigated in this chapter.
- Chapter 4 proposes the Chiarella-Heston model for more realistic financial market simulation, and explores the integration of the Chiarella-Heston model with deep reinforcement learning algorithms to optimise hedging strategies.
- Chapter 5 develops a high-frequency agent-based financial market simulator and presents in-depth simulations of flash crash scenarios, revealing key factors that significantly impact flash crash scenarios and high-frequency market dynamics.
- Chapter 6 summarises the key contributions of the research and outlines directions for future work.

1.6 About this journey

In this subsection, I briefly describe my research journey and how my PhD was shaped. My interest in financial markets was inspired by my father. He works in a bank, and also invests in the stock market. He often listened to financial news at home. Over time, I became increasingly fascinated by financial markets and dreamed of studying them one day.

During my undergraduate studies, I was introduced to quantitative finance, which uses computational methods to find patterns in financial data. It reminded me of my younger days when I recorded the

historical prices of a stock with pen and paper, hoping to predict future movements. I quickly became convinced that quantitative methods were the key to understanding financial markets.

When I started this PhD journey, my supervisors suggested exploring agent-based models. After an extensive review, I believe agent-based models offer a more suitable framework than traditional models for capturing dynamics in financial markets. This is because modern financial markets are complex systems, with millions of heterogeneous participants. It is impossible to model such a complex system with a set of closed-form equations. However, agent-based models offer a promising alternative in modelling financial markets by explicitly modelling these diverse agents. This led to the work presented in Chapter 3.

Before starting my PhD, I spent a year working in a market-making team, where we provided liquidity in derivative markets. At that time, one of our core challenges was hedging. Although the Black-Scholes model is a foundational tool, it has many practical limitations. Motivated by these experiences, I began exploring more adaptive approaches to hedging after building the agent-based simulator. Deep hedging caught my attention. I believe that by utilising deep learning techniques, we could train an intelligent agent to achieve more efficient hedging. The research resulting from this direction is presented in Chapter 4.

I still recall my second supervisor, Stephen, remarking, "It is easy to make money for a while, but the real challenge is keeping your profits in the long run." Risk management is very important for success in financial markets. There are many examples of once-successful financial institutions that collapsed overnight. One typical risk scenario is flash crashes, which are extreme events that expose the vulnerabilities of high-frequency markets. With my developed agent-based financial market simulator, I started working on simulating flash crash events, aiming to understand their causes and investigate how to mitigate the effects of such risk scenarios. This work is presented in Chapter 5.

Chapter 2

Background and Related Work

2.1 Introduction

In this chapter, we introduce the foundational concepts and existing research related to this thesis. We begin by exploring agent-based models, which provide a powerful tool for simulating and analysing complex financial systems. Then we introduce the related works that align with the three main chapters in this thesis. Essential concepts such as limit order books, hedging and reinforcement learning are introduced along the way.

2.2 Agent-based Financial Market Simulation

2.2.1 Agent-based Models

An agent-based model (ABM) is a computational simulation that operates based on the individual decisions made by programmed agents [TBSY16]. Each agent in an ABM operates based on a set of predefined rules and can represent individuals, groups, or entities with distinct behaviours and decision-making processes [CH11]. These models are particularly useful for studying complex systems where the interactions between agents lead to emergent phenomena that are not easily predictable from the properties of individual agents alone.

ABMs are widely used across various disciplines, including ecology, sociology, economics, and finance. They offer a bottom-up approach to modelling, where the behaviour of the entire system emerges from the

interactions of its components [CH11]. This contrasts with traditional top-down modelling approaches, which often rely on aggregate equations and assumptions. ABMs can incorporate heterogeneous agents, adaptive behaviours, and localised interactions, making them well-suited for exploring dynamic and non-linear systems.

In financial markets, agent-based models are valuable for understanding market dynamics, price formation, and the impact of various trading strategies [LeB06]. They can simulate market microstructure by modelling interactions between different types of traders, such as liquidity providers and liquidity takers, to explain phenomena like market impact, price volatility, and the formation of bubbles and crashes. ABMs also study price formation and discovery, analysing how information dissemination, order execution, and trading strategies affect price movements, particularly in the context of high-frequency and algorithmic trading.

2.2.2 XGBoost Algorithm for Agent-based Model Calibration

Agent-based models require calibration to replicate real-world dynamics. Various agent-based model calibration methods are developed, such as the Methods of Moments [FW12] and Machine Learning Surrogate Modelling Method [LRS18]. Throughout the thesis, we calibrate the agent-based models with a surrogate modelling method based on XGBoost algorithm [CG16]. Here we introduce the XGBoost algorithm, which is the core component of the model calibration process.

Extreme Gradient Boosting (XGBoost) is a scalable and efficient implementation of the gradient boosting framework introduced by [CG16]. It has become one of the most widely used machine learning algorithms for structured data because of its accuracy, efficiency, and flexibility [FF12]. At its core, XGBoost builds a strong predictive model by combining multiple weak learners, usually decision trees. The algorithm proceeds sequentially, and each new tree is trained to correct the residual errors of the previous trees [CG16]. The guidance of the XGBoost algorithm is to minimise a regularised objective function that balances model complexity with training loss. For more details about the algorithm, refer to [CG16].

One distinguishing feature of the XGBoost algorithm is the use of a second-order Taylor expansion as the loss function, enabling more precise gradient-based updates during training [FF12]. In addition, the algorithm also includes both L1 and L2 regularisation to further enhance generalisation performance. The XGBoost algorithm is also designed to handle sparse data efficiently by learning default directions

for missing values. Furthermore, XGBoost is highly efficient for large-scale datasets because of its support for parallelised tree construction.

In this thesis, XGBoost is employed as a surrogate model to approximate the mapping between ABM parameters and observed market dynamics. As ABM simulations typically involves repeatedly running stochastic simulations and comparing their outputs to real market dynamics, ABM calibration process is often computationally intensive. By training an XGBoost surrogate model, we obtain a fast and flexible approximation of this parameter-to-feature relationship. This surrogate can then be used to efficiently explore the ABM parameter space. In this way, the efficiency of the calibration process is significantly improved. The ability of XGBoost to handle nonlinear dependencies, high-dimensional inputs, and noisy outputs makes it particularly suitable for this role.

2.2.3 Agent-based Financial Market Simulation

ABMs find widespread application in simulating financial markets. In agent-based simulated financial markets, each agent aims to process the vast amounts of time series information generated during the market simulation and translate it into trading decisions [LeB01]. The model aggregates these trading decisions from all agents to build market snapshots (e.g. limit order book states) for each step and generate transactions. With the advantage of capturing the heterogeneity of agents and diversity of the underlying economic system, ABMs provide a promising alternative to traditional equilibrium-based economic models.

[GS93] build an agent-based model with only zero-intelligence traders to simulate financial markets. Those zero-intelligence traders are not able to think strategically or do any advanced learning, or statistical modelling of the financial market. Surprisingly, results show that zero-intelligence traders can trade very effectively in the simulated market. The prices tend to converge to the standard equilibrium price and market efficiency tends to reach a very high level. According to their experimental results, they argue that some stylised facts in financial markets may rely more on institutional design rather than actual agent behaviours. Agent-based models are also proposed to model the "Trend" and "Value" effects in financial markets. Chiarella designed an agent-based model composed of two types of traders: fundamentalists and chartists [Chi92]. With only two types of traders, lots of dynamic regimes that are compatible with empirical evidence can be generated in the simulated artificial financial market. An extension of the Chiarella model is proposed in [MCB20]. The extended model adds a new type of trader

called noise trader and allows the fundamental asset value to have a long-term drift. The extended Chiarella model is capable of reproducing more realistic price dynamics. This extended Chiarella model in [MCB20] forms the basis of the proposed agent-based financial market simulation in Chapter 3 of this thesis, while the other two models proposed in this thesis can be regarded as extensions of this basic model. A more complex agent-based model for financial market simulation is proposed in [MBGC19]. Five different types of traders are present in this simulated market: market makers, liquidity consumers, momentum traders, mean reversion traders, and noise traders. Their model is capable of replicating most of the existing stylised facts of limit order books, such as autocorrelation of returns, volatility clustering, concave price impact, long memory in order flow, and the presence of extreme price events. Those stylised facts have been observed across different asset classes and exchanges in real financial markets. The successful replication of these stylised facts indicates the validity of their agent-based simulation model. It is shown that agent-based financial market simulation is capable of generating artificial financial markets with realistic macro behaviours.

The prevalence of electronic order books and automated trading permanently changed the way the market works. It is virtually impossible to infer meaningful relationships between market participants using traditional mathematical methods because of the complexity of electronic financial markets. Instead, agent-based financial market simulation has been gradually gaining popularity in the market microstructure literature. An agent-based simulated financial market offers an experimental environment for examining market features and characteristics. It also provides plenty of artificial financial market data for analysis. [HTC⁺14] develop an agent-based model for use by researchers, which offers the capability of capturing the organisation of exchanges, the heterogeneity of market participants, and the intricacies of the trading process. Agent-based models can also provide regulators with an experimental environment that helps to improve comprehension of complex system outcomes. In other words, it allows for a clearer examination of the relationship between micro-level behaviour and macro outcomes. For example, [DO07] test the regulatory changes that came with decimalisation in the NASDAQ market using agent-based financial market simulation. The agent-based models of the NASDAQ market shed light on how these changes would impact market function.

Given an agent-based model, how to effectively calibrate the model to real data is still an open challenge. Successful calibration of an agent-based model enables the model to generate qualitative or quantitative properties that are observed consistently in empirical data. Such properties are difficult to reproduce using traditional equilibrium based approaches [LeB06]. Many calibration methods for agent-based

models have been proposed in the literature. For some simple models such as the CATS model in [BCGV07], model parameters can be read directly from the data. There are also various analytical calibration methods for agent-based models. For example, [MCB20] apply the Expectation-Maximisation method to get the maximum likelihood estimation of the parameters of the Extended Chiarella model. However, for most complex models, model parameters are not directly observable. In addition, most agent-based models that are of interest are incompatible with calibration methods that require analytical solutions. In those cases, the only choice for model calibration is the simulation-based method. The most common simulation-based calibration method is the simulated minimum distance method and its variations [JM15]. The simulated minimum distance method involves the construction of an objective function that measures the distance between real data and simulated data for a given set of model parameters. Optimisation methods are subsequently applied to minimise the distance to get an optimal set of model parameters. In the context of economic agent-based models, popular distance measures include weighted sums of the squared errors between empirical moments and simulated moments. [Fra09] applies the method of simulated moments to estimate the parameters of an agent-based asset pricing model. Their moment selection emphasises the reflection of certain stylised facts in financial markets, such as the fat tails and autocorrelation patterns of the daily returns in stock price time series. To explore structural stochastic volatility, [FW12] employ the method of simulated moments to estimate parameters of different candidate agent-based asset pricing models. They also take into consideration the proportion of Monte Carlo simulation runs that yield moments within the empirical confidence intervals for the moments involved.

One obstacle in the development of robust and widely applicable agent-based model calibration strategies is computational complexity. Most agent-based models of interest are computationally costly to simulate and the situation is even worse when it comes to large-scale agent-based models. One possible solution to address this challenge is the use of surrogate modelling to help guide parameter space exploration and thus avoid a large amount of intensive agent-based model simulations. Examples of surrogate modelling include the Kriging [Ras03] and the machine learning surrogate approach [LRS18]. [DPV18] apply the Kriging method to enable a global exploration of the parameters space for a multi-firm evolutionary simulation model. Sensitivity analysis is also carried out in this Kriging framework. [LRS18] build machine learning surrogate models to approximate two agent-based models. Experimental results show that their XGBoost-based machine learning surrogate achieves high accuracy in approximating the relationship between model parameters and model outputs.

2.2.4 Agent-based Models and Simulation in This Thesis

As introduced above, agent-based financial market simulation simplifies complex financial system simulation by including a set of individual agents, a topology and an environment. Different agent-based models in the literature focus on different practical problems in financial markets. In this thesis, three challenges are addressed by utilising agent-based financial market simulation. The agent-based financial market simulation presented in this thesis is built upon the Extended Chiarella model in [MCB20]. Different modifications and generalisations of this model are proposed for addressing different challenges. Specifically, the price formation process is investigated in Chapter 3, with Chapter 4 investigating the optimal hedging problem with deep reinforcement learning method in the environment of agent-based financial market simulation. In Chapter 5, we focus on agent-based models applied to high-frequency financial market simulation and flash crash analysis. All three topics are investigated in an agent-based financial market simulation framework. The model calibration process for all agent-based models proposed in this thesis follows the same approach proposed in Chapter 3, which utilises a surrogate modelling approach based on the XGBoost algorithm. In the following section, we provide related work corresponding to the three topics.

2.3 Price Formation Process

With the rapid development of modern financial markets, the price formation process in financial markets has been of great interest to both researchers and practitioners. To investigate the price formation process, we first introduce basic concepts related to financial markets.

2.3.1 Financial Markets

Financial markets refer to marketplaces where the trading of financial instruments, such as stocks, bonds, currencies, and derivatives, takes place. These markets are essential for the functioning of modern economies as they facilitate the allocation of resources, distribution of risk, and creation of liquidity [BEG12]. The concept of financial markets dates back to ancient times when basic forms of trade and barter systems were invented. Over time, financial markets have evolved significantly, driven by advancements in information technology, regulatory changes, and globalisation [GS97]. There are two key components of modern financial markets that are relevant to this work: limit order books and

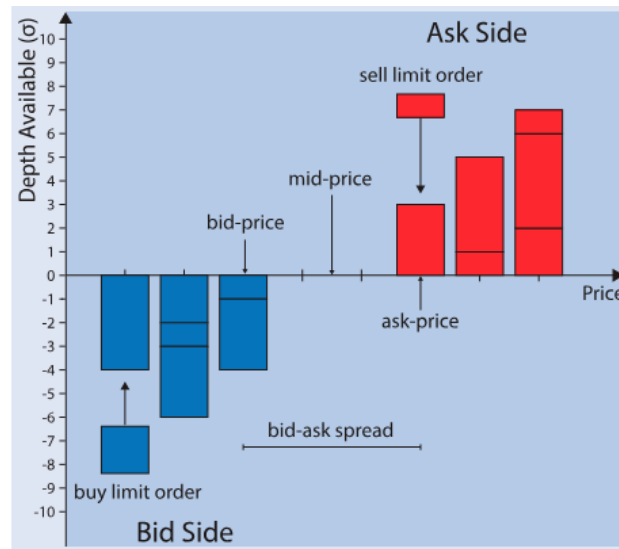


Figure 2.1 Adapted from [GPW⁺13], an illustration of a limit order book.

algorithmic trading.

Limit Order Book

Limit order books (LOBs) are among the most widely used mechanisms in financial markets globally [GPW⁺13]. In these systems, market participants place buy or sell limit orders, specifying the highest price they are willing to pay or the lowest price they are willing to accept, as well as the size of the order. The LOB tracks and aggregates unfilled limit orders on both the buy side (bids) and the sell side (asks). If a new order can be immediately matched with existing orders on the opposing side at acceptable prices, it is executed right away, thus removing liquidity from the book. The most common matching prioritisation method is price-time priority. In the price-time priority mechanism, orders are matched first based on the best price for the incoming order, and then by arrival time, with the oldest limit orders at each price level being matched first.

Figure 2.1 provides a snapshot of an imaginary limit order book status. The figure displays the top 3 price levels on both the bid and ask sides, along with the aggregated available volume at each price. In this instance, the highest bid is \$3 for 4 shares, which is the highest price a seller could instantly trade at using a market order. Conversely, a buyer could acquire up to 3 shares at a price of \$6. Trading larger quantities on either side would deplete the volume at the best price and require additional liquidity from deeper levels. Placing a buy limit order below the best ask price would not result in an immediate transaction but would instead add new volume to the bid side of the LOB. For a more detailed review of limit order book dynamics and related models, we refer readers to [GPW⁺13].

Algorithmic Trading

Algorithmic trading, often referred to as algo trading or automated trading, involves using computer algorithms to execute trades based on predetermined rules and instructions [NMTY11]. These algorithms follow a set of predetermined instructions or rules to trade financial instruments, including stocks, bonds, currencies, and derivatives. The primary goals of algorithmic trading are to increase efficiency, reduce costs, and improve the accuracy of trading operations. It has evolved significantly since the late 20th century, driven by advancements in electronic trading platforms and the development of high-frequency trading (HFT). The process typically includes strategy development, coding, backtesting, and execution, allowing trades to be executed at high speeds and frequencies without human intervention.

Common strategies in algorithmic trading include market making, arbitrage, trend following, statistical arbitrage, and mean reversion [TGL13]. These strategies leverage mathematical models and statistical analysis to identify and exploit market opportunities. The primary benefits of algorithmic trading are speed, accuracy, cost efficiency, and consistency, which enhance the overall efficiency and performance of trading operations. By minimising human errors and emotional decision-making, algorithmic systems can maintain consistent performance and execute trades at optimal prices.

However, algorithmic trading also presents challenges and risks such as market volatility, technical failures, regulatory concerns, and data quality issues [TGL13]. High-frequency trading can contribute to market instability, and system bugs can lead to significant financial losses. Regulatory scrutiny is increasing to ensure market fairness and stability, requiring compliance with regulations such as SEC rules. The future of algorithmic trading is likely to be shaped by artificial intelligence and machine learning, which can enhance the predictive power and adaptability of trading algorithms, further transforming the financial markets.

2.3.2 Existing Price Formation Process Research

One group of price formation process literature is based on the equilibrium state of financial markets. [FHBMG11] propose a pure exchange economy with a finite number of types of agents and commodities. They analyse the equilibrium price formation in a differential information market, where traders have incomplete and asymmetric information. [Jac91] shows that it is possible to have an equilibrium with fully revealing prices and costly information acquisition if the price formation process is modelled explicitly and traders are not price-takers. The price formation process is also extensively investigated

from the perspective of the double auction market. [CF96] present 14 laboratory experiments that examine the price formation process in the continuous double auction. It is shown that participants in their double auction market experiments succeed in discovering prices that would achieve most of the exchange surplus. The same is true even with no auctioneer and with traders possessing various private information. [GM94] examine the relationships between market structure and stock price formation. It is shown that trading mechanisms and price formation provide different explanations for the greater volatility at the opening of trading.

An alternative perspective for analysing the price formation process in financial markets focuses on the trend and value effects. Trend and value effects are indispensable in the price formation process within financial markets. The two interactive effects pervade all financial markets. Trend refers to the price behaviour that positive (negative) returns are more likely to be followed by positive (negative) returns. Value means that the asset price will converge to the intrinsic value of the asset, which indicates that assets with prices higher (lower) than their fundamental value tend to achieve negative (positive) future returns. The trend and value effects correspond to two types of market participants: fundamental traders and momentum traders. Fundamental traders react to the difference between fundamental value and market price, while momentum traders react to price trends. Lots of models investigating trend and value effects are analysed in the literature.

[BG80] build a model which proves that the so-called "speculation on the price trend" plays an important role in the formation of dynamic price behaviours. It is shown that speculative trading can accelerate the convergence to the equilibrium state, but it can also lead to price oscillations and market instability. [FF86] present a model containing fundamentalist and chartist to explain the dollar price in the early 1980s. Not constrained by the assumption of rational behaviours, in this model, each type of trader performs the specific task in a reasonable and realistic way. Their model provides a framework for explaining the price formation process in a variety of asset markets. [Zee74] shows that the unstable behaviours in various financial markets can be credited to the interactions between fundamental traders and momentum traders. In one famous paper, [Chi92] proposes the so-called Chiarella model which consists of fundamentalists and chartists in the artificial financial market. It is shown that the Chiarella model is capable of generating a number of dynamic regimes in financial markets which are consistent with empirical evidence. Based on the Chiarella model, [MCB20] propose the Extended Chiarella model by adding noise traders and changing the demand function of fundamentalists. The Extended Chiarella model investigates the co-existence and interaction between trend and value effects in the framework

of agent-based models. The model parameters are estimated using an Expectation-Maximisation algorithm.

2.3.3 Existing Gaps and Our Contributions

Though numerous agent-based models are proposed in the literature to analyse the price formation process, many of these models exhibit common drawbacks. Firstly, the models in the literature primarily rely on mathematical derivation for estimation. Consequently, these models produce theoretical rather than simulated results. Secondly, existing models primarily explain price time series over longer time frames such as daily or monthly price behaviours. A notable gap remains in effectively explaining the intra-day price formation process concerning trend and value effects. Thirdly, a considerable challenge persists in efficiently calibrating agent-based models.

In Chapter 3, we address these gaps by examining the intra-day price formation process using agent-based models, incorporating trend and value effects through heterogeneous traders. Firstly, the proposed model comprises momentum traders, fundamental traders and noise traders. These traders interact with each other within the simulation model, influencing price dynamics. Secondly, the XGB-Chiarella method is proposed to analyse the intra-day price formation process, offering a significant improvement over daily or monthly simulations. Thirdly, we propose an enhanced version of the calibration method based on machine learning surrogate modelling. This calibration method is employed throughout this thesis.

2.4 Deep Reinforcement Learning and Deep Hedging

2.4.1 Reinforcement Learning

Reinforcement Learning (RL) is a machine learning framework aimed at optimising a numerical quantity called a reward through iterative decision-making [SB⁺99]. An agent, the algorithm responsible for learning the decision rules, receives the current state S_t from the environment and maps states to actions using a policy function. When the agent performs an action a_t , the environment updates the reward r_t and transitions to a new state S_{t+1} . Unlike supervised learning, RL involves the agent's actions influencing future states, creating interaction within the system [KLM96]. Figure 2.2 provides an illustration of the RL framework.

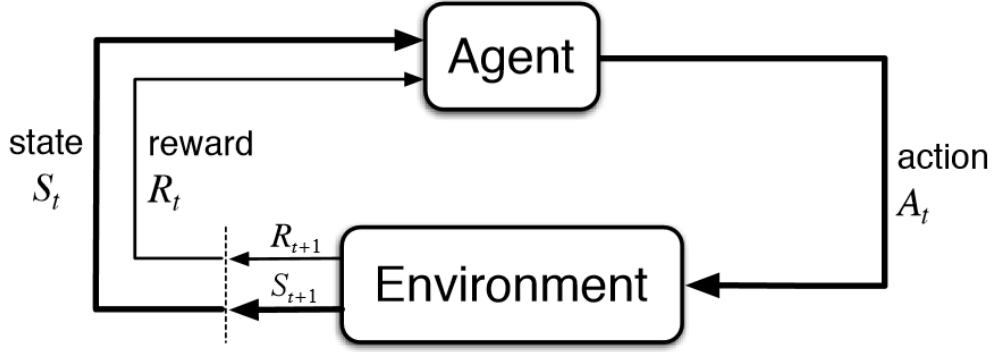


Figure 2.2 Components and flows of the RL framework. Adapted from [PAD22].

Formally, RL is modelled as a Markov Decision Process (MDP), characterised by the agent-environment interaction through actions (a_t), states (S_t), and rewards (r_t) [PAD22]. An MDP is defined by discrete time steps $t_0, t_1, \dots, T$ and a tuple $\{\mathcal{S}, \mathcal{A}(S), \mathbb{P}(S_{t+1}|S_t, a_t), R, \gamma\}$, where:

- $\mathcal{S}$ represents the set of states such that $S_t \in \mathcal{S}$.
- $\mathcal{A}(S)$ is the set of possible actions $a_t \in \mathcal{A}(S)$ that can be taken when in state $S_t = S$.
- $\mathbb{P}(S_{t+1}|S_t, a_t)$ describes the probabilities of transitioning to state $S_{t+1} \in \mathcal{S}$ given that action a_t is taken in state S_t .
- R and γ represent the reward function and the discount factor, which prioritises immediate rewards over future rewards.

The transition probabilities ($\mathbb{P}$) is supposed to fulfill the Markov Property:

$$\mathbb{P}(S_{t+1}|S_{0:t}) = \mathbb{P}(S_{t+1}|S_t) \quad (2.1)$$

which asserts that the future state S_{t+1} depends only on the current state S_t , not on the sequence of states that preceded it. As for the agent, the objective is to discover a policy π that maximises the expected cumulative reward, discounted over time:

$$\operatorname{argmax}_{\pi} \mathbb{E}^{\pi} \left[\sum_{t=0}^T \gamma^t R_t(S_t, a_t) \right] \quad (2.2)$$

Various reinforcement learning algorithms are proposed to solve this optimisation problem: DQN [MKS⁺13], DDPG [LHP⁺15], A2C [MBM⁺16], PPO [SWD⁺17], etc.

2.4.2 Hedging

Hedging is a risk management strategy used by investors and companies to protect themselves against potential losses from adverse price movements in financial markets [Bar19]. By taking an offsetting position in a related security, derivative, or asset, hedgers can mitigate the risk of price fluctuations that might negatively impact their holdings. The primary goal of hedging is not to eliminate risk entirely but to reduce it to a manageable level, thereby stabilising returns and preserving capital [RS11].

The concept of hedging dates back to the origins of trade and finance, where merchants and farmers sought to protect themselves against the risks associated with price volatility in goods and commodities. In modern financial markets, hedging is widely used across various asset classes, including equities, bonds, currencies, commodities, and interest rates. Common instruments used for hedging include futures contracts, options, swaps, and forwards, each providing different mechanisms and degrees of protection [HB16].

Financial hedges utilise these financial instruments to offset potential losses from market movements. For example, a company that relies heavily on oil might hedge against rising oil prices by purchasing futures contracts. Investors might use options to hedge against potential declines in stock prices. While hedging can effectively manage risk, it often involves costs such as premiums for options or margin requirements for futures [RS11]. Additionally, hedging strategies require careful planning and execution to ensure they align with the overall financial goals and risk tolerance of the entity. Properly implemented, financial hedging can help achieve greater financial stability and predictability in investment portfolios and business operations.

2.4.3 Combining Reinforcement Learning and Hedging with Deep Learning Techniques

Deep Reinforcement Learning (DRL) is a type of machine learning that combines deep neural networks and reinforcement learning techniques to learn complex decision-making processes. Compared with traditional Reinforcement Learning (RL), one of the significant advantages of DRL is its ability to learn directly from high-dimensional input data, such as images or speech signals. DRL has been widely applied in various fields such as robotics, gaming, natural language processing, and computer vision. [MKS⁺13] introduces Deep Q-Network, which demonstrates how deep neural networks could be used to learn control policies for playing Atari games directly from the game screen. DQN is one of the first successful applications of DRL to high-dimensional visual inputs. Later in [MKS⁺15], they further

improved the algorithm and achieved human-level performance on various Atari games, showcasing the potential of DRL for complex control tasks. [LHP⁺15] introduces the Deep Deterministic Policy Gradient (DDPG) algorithm, extending DRL to continuous action spaces. DDPG has been widely used in robotics and control tasks. [HHS15] modify the DDPG algorithm to use recurrent networks trained with backpropagation through time. They demonstrate that the resulting algorithms, Recurrent DPG (RDPG) can be applied to a number of partially observed physical control problems with diverse memory requirements.

The idea of using deep reinforcement learning in finance is not new, but the application of these techniques in hedging has gained attention recently due to the need to hedge complex financial instruments, such as options on baskets of stocks or bonds, where traditional linear or non-linear models are not effective. Deep hedging [HLJB19] is a recent development that aims to create hedging strategies for complex financial instruments using deep reinforcement learning techniques. [HLJB19] introduces the concept of deep hedging and demonstrates how deep learning techniques, such as neural networks, can be used to optimise hedging strategies for financial derivatives. That work provides a theoretical framework for deep hedging and explores its applications in various financial settings. Later in [BGT⁺20], the deep hedging framework is extended to address the risk management problem of illiquid derivatives in a non-frictionless market, using a reinforcement learning approach. [HTZ21] train the deep hedging agent with synthetic data generated from a rough volatility model, which provides a more accurate representation of market volatility dynamics compared to traditional models. The authors analyse the hedging performance of the original architecture under rough volatility models. [CCHP21] propose a revised DDPG algorithm with two different Q-functions to derive optimal hedging strategies for derivatives when transaction costs exist. Their results showed that deep hedging can outperform traditional methods, such as delta hedging, in terms of hedging performance and hedging accuracy.

2.4.4 Existing Gaps and Our Contributions

The most significant gap identified in the literature on deep hedging concerns the availability of data. Deep hedging inherently requires large volumes of data to train the reinforcement learning agent. Since historical data alone is insufficient, synthetic data becomes essential. However, as shown in the above review, most research focuses on reinforcement learning algorithms, with little attention given to the data challenges.

To our knowledge, no existing work integrates agent-based models within the deep hedging framework. In Chapter 4, we address this gap by combining an agent-based model with the deep hedging framework. For the deep reinforcement learning method, we adopt the revised DDPG algorithm from [CCHP21]. Two separate Q-functions are used to track the expected value of both the hedging cost and the square of the hedging cost. The deep hedging agent's learning objective is to minimise a function that combines the mean hedging cost and a constant multiplied by the standard deviation of the hedging cost. Though there is no inherent distributional assumption, this setup allows the standard deviation of the hedging cost to be incorporated into the objective function.

2.5 Flash Crash Analysis

2.5.1 Review of Existing Flash Crash Research

During the 2010 Flash Crash, over a trillion dollars were wiped off the value of US equity markets in an event that has been largely attributed to the rapid rise of algorithmic trading and high-frequency trading [KKST17]. The base indices in both the futures and securities market experienced a rapid price fall of more than 5% in just several minutes, after which the bulk of the price drop was recovered nearly as fast as it fell. The staff from CFTC and SEC present a thorough report on what happened during the 2010 Flash Crash event [SC10]. They identified an automated execution algorithm that sold a large number of contracts as the main catalyst for the flash crash. The Sell Algorithm, which was activated on the E-mini S&P 500 futures market, kept pace with the market aiming at selling around 9% of the previous minute's trading volume [SC10]. Even though no negative impact was known previously, this process triggered a cascade of panic selling by market participants who employ high-speed automated trading systems. The consequent "hot-potato" effect, where those market participants rapidly acquired and then liquidated positions among themselves, resulted in rapid and extreme price decline.

Flash crash episodes have attracted attention after the 2010 Flash Crash event. Several months after the crash, the staff from regulatory authorities released a report that highlighted the important role of a large seller in initiating the flash crash event [SC10]. This seller employed a sell algorithm that focused solely on market volume to determine the order size. It is reported that although high-frequency traders appear to have exacerbated the magnitude of the crash, they did not actually trigger the flash crash. Though high-frequency traders played a role in creating the so-called "hot-potato" effect, the flash crash would very likely have been avoided without the overly simplistic sell algorithm based on

volume alone. There is also a lot of academic research on flash crash episodes. For example, [KKST17] apply purely empirical approaches to understanding the causes of the 2010 Flash Crash. They use regression analysis on a unique dataset that is labelled with the identities of all market participants. It is demonstrated that in response to the activity of the Sell Algorithm, high-frequency traders caused the "hot-potato" effect that exacerbated the price drop. This is consistent with the official report [SC10]. [BCP13] apply a simulation approach to studying the consequences of different regulatory interventions that aim to stabilise the limit order book market after a flash crash event. They trigger a flash crash by simulating the submission of an extremely large order. In this way, they explore the conditions under which the influence of a flash crash could be substantial and the mechanisms that could mitigate the influence. Their simulation results indicate that three mechanisms could potentially mitigate the detrimental effects of flash crashes: introducing minimum testing times, switching to a call auction market mechanism and shutting off trading for a period of time.

An important stream of flash crash research is simulating flash crashes in agent-based models, which usually involves investigating how high-frequency traders contribute to the emergence of flash crash events. [PHT⁺12] develop an agent-based model of the E-mini S&P futures applied to flash crash analysis. A general flash crash in price is replicated in their model. However, they only reproduce a rough shape of the flash crash price behaviour, and detailed analyses of trader behaviours and market depth are absent. [KNWB18] develop an agent-based model to analyse the flash crash episodes in the sterling-dollar Forex market. They emphasise the important role of high-frequency traders in the emergence of flash crash episodes. The proposed approach in Chapter 5 is partly inspired by their work. [JL NRF16] develop an agent-based model to study how the interplay between low-frequency traders and high-frequency traders impacts asset price dynamics. Their experimental results indicate that high-frequency trading exacerbates market volatility and plays a fundamental role in the emergence of flash crashes. They also show that flash crashes are associated with two salient characteristics of high-frequency traders, namely their capability of generating high bid-ask spreads and synchronising on the sell side of the limit order book. The most similar model to our proposed one in literature is the agent-based model proposed in [VW14]. They build an agent-based simulation framework that reproduces key characteristics of the 2010 Flash Crash. Consistent with [SC10], they argue that high-frequency traders play an important role in creating a vicious feedback loop system, which is triggered by a large institutional sell. Their results imply that policymakers should pay more attention to the number of high-frequency traders in the market, the inventory sizes of these traders, and the regulated tick size. [LN19] test the effects of a set of regulatory policies with regard to high-frequency

trading by developing an agent-based model that is able to generate flash crashes. It is shown that there is a trade-off between market stability and resilience when determining policies directed towards high-frequency trading. Their experimental results indicate that possible policies to prevent flash crash events include the imposition of minimum order resting times, circuit breakers, cancellation fees and transaction taxes.

There are also different angles of view for flash crash episodes in literature. [PCW19] design and implement a hybrid microscopic and macroscopic agent-based approach to investigate the conditions that give rise to the "electronic contagion"¹ of flash crash events. Their results demonstrate that the flash crash contagion between different assets is dependent on portfolio diversification, behaviours of algorithmic traders, and network topology. It is also emphasised that regulatory interventions are important during the propagation of flash crash distress. [MY19] look at the flash crash event from the perspective of cross-arbitrage. They find that the breakdown of cross-arbitrage activities between related markets plays an important role in exacerbating the flash crash event. [KO20] analyse price impact during flash crash events in their market microstructure invariance model. It is shown that the actual price declines in flash crash events are larger than the predicted price impact. [Mad12] argues that the flash crash episodes are linked directly to the current market structure, mostly the pattern of volume and market fragmentation. It is suggested that a lack of liquidity is the critical issue that requires the greatest policy attention to prevent future flash crash events. Similarly, [BDSY10] explicitly owe the flash crash in ETFs to an extreme deterioration in liquidity. Their results are consistent with the liquidity provision behaviour in financial markets. [PHSB17] explore how the levels of information can be used to predict the occurrence of flash crash events. Their findings suggest that some stability indicators derived from limit order book information are capable of signalling a high likelihood of an imminent flash crash event. [GKP12] analyse mini flash crashes, which are the scaled-down versions of the 2010 Flash Crash. It is shown that mini flash crashes also have an adverse impact on market liquidity and are associated with the fleeting liquidity phenomenon [CT11].

2.5.2 Existing Gaps and Our Contributions

The review highlights a significant gap in the literature regarding the analysis of high-frequency dynamics during flash crashes in agent-based financial market simulations. Most existing studies focus solely on

¹Electronic contagion refers to the contagion phenomenon in financial markets that results from interactions between trading algorithms, instead of human traders.

analysing historical flash crash events, neglecting the potential variations between different flash crashes. A simulator capable of generating diverse flash crash scenarios would be highly beneficial. While some flash crash simulators exist, they typically do not operate at high-frequency levels. For instance, the simulation in [PHT⁺12] reproduces only a rough approximation of flash crash price behaviour, lacking detailed analysis of trader interactions.

In Chapter 5, we address this gap by proposing a high-frequency financial market simulator capable of operating at the millisecond level to simulate flash crash events. This simulator is used to reproduce the 2010 Flash Crash, providing detailed insights into the dynamics and interactions of traders during the event. The analysis reveals key factors influencing the characteristics of flash crashes. Additionally, the simulator is applied to study mini flash crashes, uncovering critical drivers of market instability. These contributions offer valuable insights into the mechanisms underlying flash crashes.

2.6 Summary

This chapter provides a detailed overview of the background and related work necessary to understand the subsequent chapters of this thesis. We began by discussing financial markets, focusing on key elements such as limit order books, algorithmic trading, and hedging strategies. These discussions highlighted the mechanisms that drive trading activities and risk management in modern financial markets. We then explore agent-based models, emphasising their utility in simulating market dynamics and investigating the interactions between different market participants. The application of reinforcement learning in financial markets is examined, in particular, its role in optimising trading and hedging strategies. Finally, we reviewed the existing literature on agent-based financial market simulations, the price formation process, deep reinforcement learning, and flash crash analysis. These topics are closely related to the main contributions of this thesis. This comprehensive review provides baselines for the contributions presented in the following chapters.

Chapter 3

Price formation process investigation with agent-based financial market simulation

3.1 Introduction

This is the first main chapter of this thesis, where we introduce XGB-Chiarella, a method to generate realistic intra-day financial price data using agent-based models. Despite its simplicity, the model effectively captures the intra-day trend and value effects, providing support for the existence of a universal intra-day price formation mechanism. The proposed XGB-Chiarella method also integrates a novel calibration process for agent-based models, which is based on the XGBoost machine learning algorithm.

3.1.1 Motivation

In the past decade, algorithmic trading has grown rapidly across the world and has become the dominant way securities are traded in financial markets, currently generating more than half of the volume of U.S. equity markets. Constantly improving computer technology and its application by both traders and exchanges, together with the evolution of market micro-structure, automation of price quotation and trade execution have together enabled faster trading. Consequently, intra-day price formation underpinning this trading process has become the focus of intense research attention in recent years as market participants attempt to gain greater insight into how prices are determined and hence improve trading performance.

Price formation determines the price of an asset through interactions between buyers and sellers. It is at the core of the efficient and transparent operation of markets for goods and services. The balance between buyers and sellers provides an effective indicator of demand and supply in a market, where demand and supply are generally significant but not the only driving factors behind price movements. This is because the mechanisms of price discovery indicate what sellers are willing to accept and what buyers are willing to pay, so the price discovery process is concerned with finding an equilibrium (or near equilibrium) price that enables the greatest liquidity for that asset at a given point in time. Beyond supply and demand, attitudes to risk, volatility, available information and market micro-structure all exert varying levels of influence on the price discovery process.

[Lo17] explains the process of price formation in properly functioning markets as generally involving market participants engaging in cause and effect reasoning along the lines of “if my strategy is x, then the market will respond with y, in which case I will respond with z...”. Even this simple process requires that the algorithms of buyers and sellers have some understanding of the other’s motives and incentives. Theoretically, this approach would imply that such chain reasoning could infinitely recurse.

However, [SC19] define a "price formation mechanism" as a high-level map representing the relationship between asset price and variables such as order flow and market price history. Modelling such a mechanism using stochastic differential equation models, machine learning prediction models and market micro-structure, all provide different ways to represent this map. However, an issue central to the price formation mechanism is the degree to which such a high-level map is universal. That is, whether the price formation mechanism is independent of the particular asset being considered. The universal existence of certain empirical stylised facts seems to be evidence supporting this universality traded. In this chapter, we present evidence for the existence of such a universal price formation mechanism by proposing the XGB-Chiarella method, which is able to reproduce realistic synthetic data for various stocks from different exchanges. The fact that the XGB-Chiarella method is based on the same underlying agent-based model backs the existence of a universal price formation mechanism.

Our investigation of the intra-day price formation process is through financial market simulation using agent-based models. Financial markets are obviously one of the most dynamic systems in existence. With huge potential academic and industrial value, financial market simulation in agent-based models is an exciting new field for exploring behaviours of financial markets. In an agent-based artificial financial market, heterogeneous agents (traders) trade a financial instrument through a realistic trading mechanism for price formation. Unlike traditional economic theories, there is no equilibrium assumption

in agent-based financial markets. In addition, traders are no longer assumed to have rational behaviours as in traditional economic theories. The removal of these assumptions makes agent-based financial market simulation more realistic than traditional equilibrium-based economic and financial theories. Various agent-based simulators have been developed in the literature. However, there are still gaps in creating ideal agent-based financial market simulators that are capable of generating realistic synthetic market data and shedding light on the intra-day price formation process. Specifically, most existing agent-based financial markets are of lower frequency such as daily or monthly. To investigate the intra-day price formation process, a higher simulation frequency is needed.

A realistic, intra-day financial time series data generation process requires that the parameters of the agent-based model must be calibrated to be as close to real market data as possible. Realistic simulated market data are supposed to exhibit certain characteristics known as stylised facts, which are universally observed in historical financial market data. Some stylised facts originate from behaviours of market participants, while others could be natural consequences of market structure design. Examples of stylised facts include fat tails of returns, volatility clustering, etc. Parameter calibration with respect to certain stylised facts can be extremely challenging due to the huge parameter space and complexity in designing explicit optimization objective functions.

To sum up, there are still two challenges of great interest in this field:

- C1.1: How to implement an agent-based financial market simulator which allows for the investigation of intra-day price formation process in financial market.
- C1.2: How to calibrate the agent-based financial market simulator to ensure realism and reproduce common stylised facts.

To address the two challenges, we developed XGB-Chiarella, which is a novel approach to developing and calibrating an intra-day financial market simulator to narrow the existing gaps. The XGB-Chiarella methodology has two essential components: the underlying mathematical model for simulation and the calibration workflow with surrogate modelling. For the underlying model, the simple but powerful Extended Chiarella model [MCB20], which consists of fundamental traders, momentum traders and noise traders, is used as the underlying model for the XGB-Chiarella method. For the calibration workflow, the method utilises XGBoost¹ algorithm to build a machine learning surrogate for the

¹XGBoost [CG16] is a highly flexible and efficient machine learning algorithm based on an ensemble of decision trees.

purpose of model calibration. We will show that even with only one agent for each type of trader, the XGB-Chiarella method is able to generate realistic price series simulations after a proper model parameter calibration process. The setup of one agent representing each type of trader is consistent with the modelling choice commonly found in the literature, particularly from a mean-field theory perspective [LLLL16].

3.1.2 Our Contributions

In this chapter, we propose XGB-Chiarella - a method for developing and calibrating an agent-based market simulator to generate realistic intra-day financial market data. The underlying model is the Extended Chiarella model [MCB20]. We adapt the machine learning surrogate modelling approach in [LRS18] to calibrate the agent-based financial market simulator. By reproducing realistic synthetic data for various stocks, we show that there is a universal intra-day price formation process capable of reproducing observed trends and value effects. The main contributions are:

- Addressing challenge C1.1: The Extended Chiarella model is originally tested on monthly data. To address Challenge C1, we further tested the Extended Chiarella model in intra-day minute data to explain the intra-day price formation process. The model is tested extensively in more than 75 stocks from three major exchanges in the world: Nasdaq, London Stock Exchange, and Hong Kong Stock Exchange. For all stocks of different exchanges, the experimental results are similar and the simulator can all produce realistic simulated financial time series. This shows that trend and value effects exist universally in the stock market price formation process, regardless of the exchanges.
- Addressing challenge C1.2: Instead of the Expectation-Maximisation algorithm, we propose a novel application of a recent machine learning surrogate modelling approach [LRS18] to calibrate the Extended Chiarella model, which addresses challenge C2. The foundation for this calibration method is the surrogate modelling approach in [LRS18]. Instead of building an XGBoost classifier to predict positive calibration and negative calibration, we train an XGBoost regressor to innovatively predict the actual distance between simulated stylised facts and historical stylised facts. The distance involves not only the return distribution but also the autocorrelations between returns and squared returns. In addition, exploration-exploitation mechanism is introduced in the iterative process of selecting new points in parameter space. In terms of stylised facts distance,

results show that the proposed method performs much better than the baseline Expectation-Maximisation estimation algorithm.

3.2 Model Architecture

This section presents the set-up and components of the agent-based financial market simulator.

3.2.1 Model Set-up

We denote the price of a stock at time t as P_t . The total signed volume traded on the market from t to $t + \Delta$ constitutes the cumulative demand imbalance in the same period. This quantity is denoted as $D(t, t + \Delta)$. This aggregated demand depends on the trading strategies of various types of market participants. Following [MCB20] and [Kyl85], the price dynamics is assumed to be governed by a linear price impact mechanism:

$$P_{t+\Delta} - P_t = \lambda D(t, t + \Delta) \quad (3.1)$$

where λ is called "Kyle's lambda", which is related to the liquidity of the market and is a first-order approximation of market price sensitivity to market demand and supply. The market participants are assumed to be heterogeneous in their trading decisions. Following the Extended Chiarella model [MCB20], we populate our model with fundamental traders, momentum traders, and noise traders. In alignment with the mean-field theory, and because traders of the same type exhibit the same behaviours, we only use one agent for each type of trader. This single agent represents the corresponding type of traders in the market. With only three agents in the model, the simulation process is significantly accelerated. Each trader is associated with some parameters that control the trading behaviours and the amount of demand generated by the corresponding trader group. We will show the calibration of these parameters in later sections.

3.2.2 Fundamental Trader

Fundamental traders make their trading decisions based on the perceived fundamental value of the stock. The fundamental value is denoted as V_t . A fundamentalist will tend to buy a stock if the stock is under-priced ($V_t - P_t > 0$), otherwise it will tend to sell the stock. Following the convention in [Chi92],

we assume the aggregated demand of fundamental traders is proportional to the level of mispricing. In other words, the aggregated demand of fundamentalists is $\kappa(V_t - P_t)$, where κ controls the overall demand generated by fundamental traders. The fundamental value V_t is an exogenous signal that is input to the model.

3.2.3 Momentum Trader

Momentum traders are also called "Chartists". This group of traders buy and sell financial assets after being influenced by recent price trends. The assumption is to take advantage of upward or downward trend of the stock prices until the trend starts to fade. Instead of looking at the fundamental value of the stock, momentum traders focus more on recent price action and price movement. If the stock price has been recently rising, a long position is established; otherwise momentum traders will enter a short position.

There are lots of methods to estimate the momentum of stock prices. A common trend signal is the exponentially weighted moving average of past returns with decay rate α . This trend signal is denoted by M_t :

$$M_t = (1 - \alpha)M_{t-1} + \alpha(p_t - p_{t-1}) \quad (3.2)$$

where α is the decay rate. Given the trend signal M_t , the demand function of momentum traders is denoted as $f(M_t)$. The demand function $f(M_t)$ must satisfy two conditions:

- $f(M_t)$ is increasing.
- $f''(M_t) * M_t < 0$

where the first condition is consistent with the nature of momentum trading and the second condition imposes the risk-averse assumption to momentum traders. Consistent with [Chi92], here we choose $f(M_t) = \beta \tanh(\gamma M_t)$ with the requirement that $\gamma > 0$. γ represents the saturation of momentum traders' demand when momentum signals are very large. This phenomenon is partly due to for example budget constraints and risk aversion, which is prevalent in real chartists. β controls the overall demand generated by momentum traders. β is also assumed to be positive, i.e. the demand of momentum traders is positive when the momentum signal (M_t) is positive, otherwise the demand is negative. The choice of this demand function for momentum traders strictly satisfies the two requirements.

3.2.4 Noise Trader

Another group of market participants is noise traders. They are designed so as to capture other market activities that are not reflected by trend-following and value investing. As a result, the cumulative demand of noise traders can be described by a random walk. The random walk is also multiplied by a parameter σ_N , which controls the overall demand level from noise traders. Mathematically, for each step the demand from noise traders is sampled from a normal distribution with zero mean and standard deviation σ_N .

3.2.5 Model Dynamics

Simulation Process

Using mathematical formulas to approximate the supply and demand generated by all the participating traders, the resulting model dynamics for $\Delta t \rightarrow 0$ can be described by the following dynamical system. Note that here the fundamental value V_t is an exogenous signal that is input to the model, which is a major adaption from the original Extended Chiarella model. In addition, there are only three agents in our simulation, namely the fundamental trader, momentum trader, and noise trader. Each trader creates a demand corresponding to one term in Equation 3.3. Our simulation results will show that this model is able to generate very realistic artificial financial price time series, even though there are only three agents included in the model.

$$D(t, t + \Delta t) = \underbrace{\kappa'(V_t - P_t)\Delta t}_{\text{Fundamental}} + \underbrace{\beta' \tanh(\gamma M_t)\Delta t}_{\text{Momentum}} + \underbrace{\sigma'_N \epsilon_t \sqrt{\Delta t}}_{\text{Noise}} \quad (3.3)$$

$$dM_t = -\alpha M_t \Delta t + \alpha dP_t$$

where ϵ_t follows standard normal distribution. Substitute Equation 3.1 into the above equations, we can get:

$$dP_t = \kappa(V_t - P_t)\Delta t + \beta \tanh(\gamma M_t)\Delta t + \sigma_N \epsilon_t \sqrt{\Delta t} \quad (3.4)$$

$$dM_t = -\alpha M_t \Delta t + \alpha dP_t$$

where κ, β, σ_N equal to $\lambda\kappa', \lambda\beta', \lambda\sigma'_N$, respectively. Furthermore, the simulator runs according to a

discrete-time version of model 3.4, in which Δt is 1, corresponding to 1 unit of the smallest simulation time interval:

$$\begin{aligned} P_t - P_{t-1} &= \kappa(V_t - P_t) + \beta \tanh(\gamma M_t) + \sigma_N \epsilon_t \\ M_t &= (1 - \alpha)M_{t-1} + \alpha(P_t - P_{t-1}) \end{aligned} \tag{3.5}$$

The whole simulation runs according to Equation 3.5. For each step, each trader collects and processes market information. Internal variables associated with each trader are calculated. According to agent types and values of internal variables, demands are generated by the traders. The price evolves according to Equation 3.1.

Fundamental Value from Kalman Smoother

The only remaining unknown variable is the fundamental value of the stock. According to Equation 3.5, the simulation can proceed only if fundamental value is known and is exogenously input to the model. One difficulty is the non-observability of the fundamental value. According to [MM61], the fundamental value of a stock equals to the expected value of discounted dividends that the company will pay to the shareholders in the future. However, this methodology requires extremely strong assumptions on the future dynamics of the stock dividends. Furthermore, this approach can never reflect the intra-day change of fundamental value, while the consensus fundamental value can indeed vary during the trading day due to the continuous feed of events and news.

In this chapter, we propose a novel method which is to apply Kalman Smoother [RdB05] directly to the stock price time series to get the hidden fundamental value. Note that Equation 3.5 is a linear dynamical system in V_t , which is treated as a hidden variable of the system. Here the observations are the actual prices traded in the real market. The specific Kalman Smoothing algorithm used here can be found in [BSS04]. The algorithm is already implemented in the Python package "pykalman".

3.3 Model Calibration

In this section, we present the methodology for calibrating the agent-based financial market simulator. Calibration means finding an optimal set of model parameters to make the model generate most realistic simulated financial market. First, we describe the real data and the associated stylised facts in financial

Exchange	Stocks
Nasdaq	AAPL, HUT, AMD, AAL, MSFT, INTC, UBER, NVDA, SOFI, DKNG, WISH, HON, FB, TSLA, MRNA, AFRM, LCID, CMCSA, HBAN, TLRY, MU, CSX, CSCO, JD, UAL
LSE	LLOY.L, VOD.L, RR.L, GLEN.L, IAG.L, HSBA.L, BP.L, PRU.L, DGE.L, LGEN.L, AAL.L, AV.L, RDSA.L, STAN.L, ULVR.L, BHP.L, BATS.L, RIO.L, GSK.L, NG.L, REL.L, EZJ.L, JET.L, SMT.L, BRBY.L
HKEX	0700.HK, 3690.HK, 9988.HK, 1299.HK, 1211.HK, 2020.HK, 2269.HK, 0175.HK, 2331.HK, 1024.HK, 0916.HK, 2318.HK, 1918.HK, 9618.HK, 9999.HK, 2333.HK, 9888.HK, 0836.HK, 1919.HK, 0388.HK, 3968.HK, 1772.HK, 1171.HK, 0005.HK, 0027.HK

Table 3.1 Specific stocks from three exchanges in the dataset.

markets. Second, we define the distance between historical and simulated stylised facts, which acts as the loss function in the calibration process. Finally, the machine learning surrogate modelling for parameter space exploration is presented in detail.

3.3.1 Data

In the model calibration process, real financial market data is essential to set up the calibration target. We collected stock price data of 75 stocks from three major exchanges in the world - Nasdaq, London Stock Exchange, and Hong Kong Stock Exchange. The stocks are selected from three major market indices corresponding to the three exchanges: Nasdaq-100 index, FTSE 100 index, and Hang Seng index. Our dataset comprises intra-day minute price data for those 75 stocks, spanning the entire trading period from September 20th, 2021 to November 30th, 2021. Detailed stocks from each exchange are shown in Table 3.1.

3.3.2 Stylised Facts and Loss Function

Financial price time series data display some interesting statistical characteristics that are commonly called stylised facts. According to [Sew11], stylised facts refer to empirical findings that are so consistent (for example, across a wide range of financial instruments and different time periods) that they are accepted as truth. A stylised fact is a simplified presentation of an empirical finding in financial markets. A successful and realistic financial market simulation is capable of reproducing various stylised facts. These stylised facts include fat-tailed distribution of returns, autocorrelation of returns, and volatility clustering. The loss function used in the calibration process is constructed by measuring the distance between historical and simulated stylised facts.

Stock	FB	AAPL	VOD.L	JET.L	9999.HK	0700.HK
Excess Kurtosis	5.51	5.64	16.20	9.60	9.49	4.66

Table 3.2 Average excess kurtosis for return distributions of five stocks.

Fat-tailed distribution of returns

The distributions of price returns have been found to be fat-tailed across all timescales. In other words, the return distributions exhibit positive excess kurtosis. Understanding positively kurtotic return distributions is important for risk management since large price movements are much more likely to occur than in commonly assumed normal distributions. In this chapter, we focus on intra-day minute price returns. Excess kurtosis of the distribution of minute-level returns is calculated for each stock on each trading day. Table 3.2 presents the average excess kurtosis for the intra-day minute-level return distributions of six randomly chosen stocks. For each stock, the return distribution has significantly positive excess kurtosis, proving the fat-tail characteristic of return distributions. Figure 3.1 also shows a comparison between the cumulative distribution function of historical returns in two stocks for one trading day and that of a normal distribution with identical mean and standard deviation. Similar results are found in all other stocks from the three exchanges.

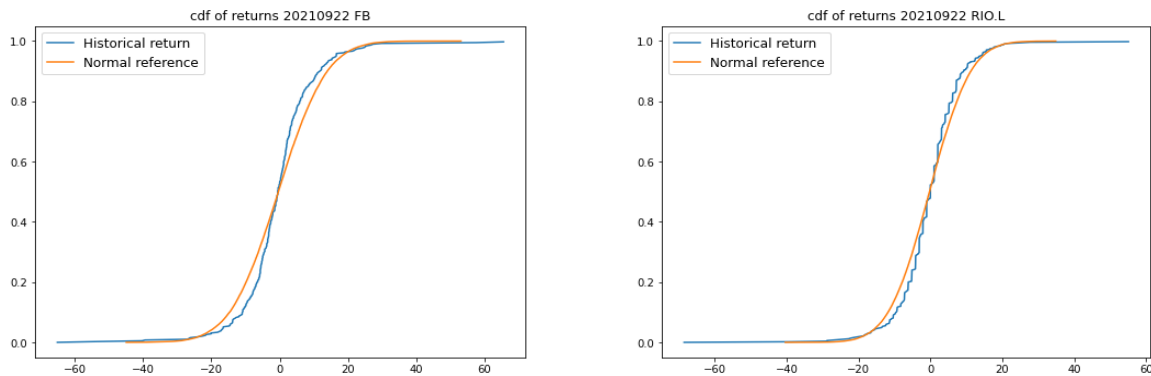


Figure 3.1 Cumulative distribution function of historical returns in two stocks (FB, RIO.L) for one trading day, in comparison with a normal distribution reference.

Autocorrelation of returns

Autocorrelation is defined to be a mathematical representation of the degree of similarity between a time series and a lagged version of the same time series. It measures the relationship between a variable's past values and its current value. Take first-order autocorrelation for example. A positive first-order autocorrelation of returns indicates that a positive (negative) return in one period is prone to be followed

by a positive (negative) return in the subsequent period. Instead, if the first-order autocorrelation of returns is negative, a positive (negative) return will usually be followed by a negative (positive) return in the next period. It is observed that the return series lacks significant autocorrelation, except for weak, negative autocorrelation on very short timescales. [MBGC19] show that the negative autocorrelation of returns is significantly stronger at a smaller time horizon and disappears at a longer time horizon. Examination of our data supports this stylised fact. Figure 3.2 shows the autocorrelation function of minute-level return time series for several stocks up to lag 30. We can see that the autocorrelation is significantly negative for small lags, and the negative autocorrelation gradually disappears for larger lags.

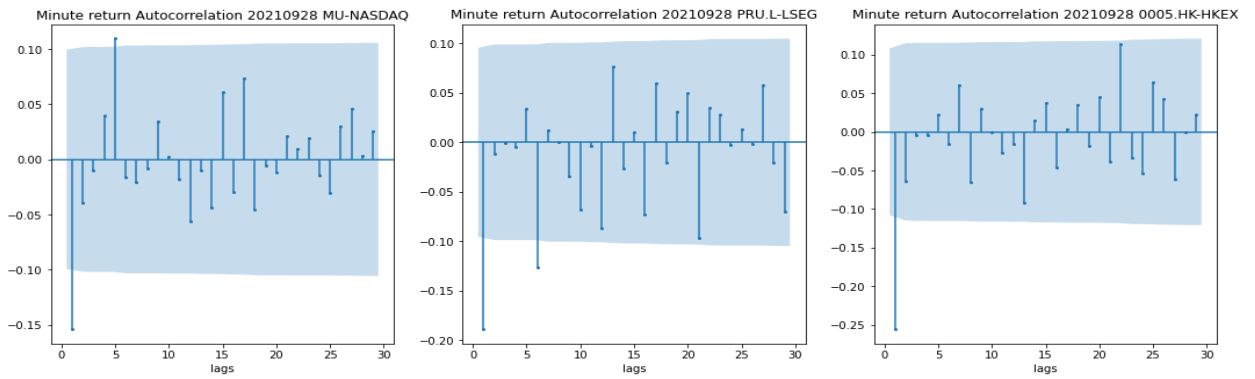


Figure 3.2 Autocorrelation patterns for returns of three stocks from the three exchanges.

Volatility clustering

Financial price returns often exhibit the volatility clustering property: large changes in prices tend to be followed by large changes, while small changes in prices tend to be followed by small changes. This property results in the persistence of the amplitudes of price changes [Con07]. It is found that the volatility clustering property exists on timescales varying from minutes to weeks and months. Volatility clustering also refers to the long memory of square price returns [MBGC19]. Consequently, volatility clustering can be manifested by the slow decaying pattern in the autocorrelations of squared returns. Specifically, for short lags the autocorrelation function of squared returns is significantly positive, and the autocorrelation slowly decays with the lags increasing. Figure 3.3 shows the autocorrelation patterns for squared returns of several randomly chosen stocks from the three exchanges. Autocorrelation for squared returns of other stocks also exhibit similar patterns. It is shown that the volatility clustering stylised fact exists universally in financial markets.

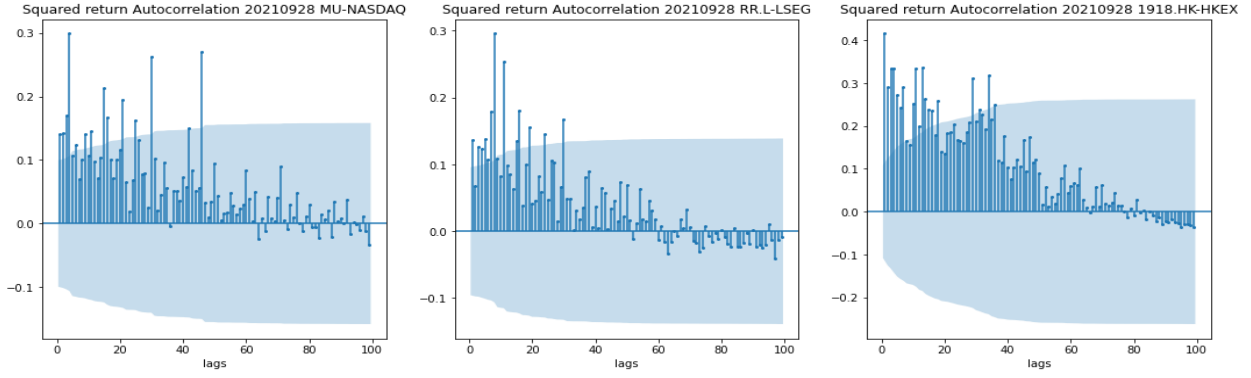


Figure 3.3 Autocorrelation patterns for squared returns of three stocks from the three exchanges.

Stylised Facts Distance as Loss Function

The target for agent-based model calibration is to find an optimal set of model parameters to make the model generate realistic simulated financial market. To solve this optimization problem, it is essential to have a metric that is able to quantify the "realism" of a simulated financial market. First of all, a realistic simulated financial market must exhibit similar characteristics to real financial markets, such as the return distribution and volatility level. In addition, realistic simulated financial data are also required to reproduce other stylised facts such as the autocorrelation patterns in returns and squared returns. Here we design a stylised facts distance to quantify the similarities between simulated and historical financial data. The stylised facts distance is the weighted sum of four quantities: K-S statistic of return distributions, volatility difference, autocorrelation difference of returns and autocorrelation difference of squared returns:

$$D = w_1 * KS + w_2 * \Delta_V + w_3 * \Delta_{ACF^1} + w_4 * \Delta_{ACF^2} \quad (3.6)$$

Detailed calculations of the four quantities in the stylised facts distance are introduced below.

Kolmogorov-Smirnov (K-S) statistic is a quantity from K-S test in statistics. K-S test is a non-parametric test of the equality of probability distributions that can be used to compare two samples. Here the two samples are simulated returns and historical returns, respectively. The K-S statistic quantifies the distance between the distribution functions of simulated returns and historical returns. Recall that the empirical distribution function is an estimate of the cumulative distribution function that generated the points in the sample. Let $F_s(x)$ and $F_h(x)$ denote the empirical distribution functions of simulated returns and historical returns, respectively. Then the K-S statistic is calculated as follows:

$$KS = \sup_x |F_s(x) - F_h(x)| \quad (3.7)$$

where $\sup_x$ is the supremum of the set of differences. Intuitively, the statistic represents the largest absolute difference between the two distribution functions across all x values. The smaller the statistic is, the more similar the simulated and historical returns are. The inclusion of K-S statistic in the stylised facts distance addresses the requirements that the simulated data have a similar return distribution to historical data and exhibit fat tails of returns.

The second part of the stylised facts distance is the absolute volatility difference between simulated returns and historical returns:

$$\Delta_V = |V_s - V_h| \quad (3.8)$$

where V_s and V_h denote simulated volatility and historical volatility, respectively. This part addresses the requirement that simulated financial markets should be similar to real markets in terms of volatility.

The third part of the stylised facts distance is the difference between simulated and historical autocorrelations of returns. This part of the stylised facts distance measures the model's ability to reproduce autocorrelation patterns commonly found in historical returns. It is shown that financial price return time series lack significant autocorrelation, except for short time scales, where significantly negative autocorrelations exist. This phenomenon is backed by our historical minute-level returns data. For very small lags the autocorrelations are negative, while for larger lags the autocorrelations become insignificant. To measure the distance in autocorrelation patterns between simulated data and historical data, we invoke the autocorrelation function of returns and calculate the average absolute difference between autocorrelations of simulated return time series and historical return time series for various lags:

$$\Delta_{ACF^1} = \frac{\sum_{l \text{ in lags}} |ACF_s(l, r) - ACF_h(l, r)|}{|lags|} \quad (3.9)$$

where $ACF_s(l, r)$, $ACF_h(l, r)$ are the autocorrelation function of lag l for simulated returns and historical returns, respectively. $|lags|$ denotes the number of lags used in the calculation. Because the empirical autocorrelations are negative for very small lags and close to zero for larger lags, it is not necessary to consider all of the autocorrelation coefficients. Empirical evidence suggests that the autocorrelation pattern is well represented by the coefficients for three lags: 1, 10, 20. Also, to reduce the effects of accidental outliers, the autocorrelation function is smoothed by calculating the three-lag average. That

is, the lag 1 autocorrelation is calculated as the average autocorrelation of lag 1, 2, 3, and so as the calculation for lag 10 and lag 20. In total, autocorrelations of 9 lags (1, 2, 3, 10, 11, 12, 20, 21, 22) are considered and included in the calculation.

The last part of the stylised facts distance is the difference between simulated and historical autocorrelations of squared returns. The replication of autocorrelation patterns in squared returns indicates the model's capability to reproduce the volatility clustering stylised fact. It is shown empirically that large price changes tend to be followed by other large price changes, known as the volatility clustering phenomenon. Consequently, though there is generally no significant patterns in autocorrelations of returns, the autocorrelations of squared returns are significantly positive, especially for small time lags. Also, as time lag increases, the autocorrelation of squared returns displays a slowly decaying pattern, as shown in Figure 3.3. Similar to the difference between autocorrelations of returns Δ_{ACF^1} , the difference between autocorrelations of squared returns is calculated as follows:

$$\Delta_{ACF^2} = \frac{\sum_{l \text{ in lags}} |ACF_s(l, r^2) - ACF_h(l, r^2)|}{|lags|} \quad (3.10)$$

where $ACF_s(l, r^2)$, $ACF_h(l, r^2)$ are the autocorrelation function of lag l for simulated squared returns and historical squared returns, respectively. $|lags|$ denotes the number of lags used in the calculation. Unlike the case for autocorrelations of returns calculation, here we use a different $lags$. Because empirical autocorrelations of squared returns are significantly positive and slowly decaying, we consider the autocorrelations of squared returns from a minimal lag length of 1 minute up to a maximal lag length of 20 minutes. In total, the autocorrelations of 20 lags (1, 2, 3, ..., 18, 19, 20) are considered and included in the calculation.

The above four parts, along with the corresponding weights, constitute the stylised facts distance in Equation 3.6. In our experiments, there is no preference for any one of the four stylised facts. Thus all the weights are equal to 1. Since it happens that the four quantities are of the same orders of magnitude, there is no need to adjust weights either. Also note that the stylised facts distance is a function of model parameters. In other words, given a set of model parameters, there is a unique stylised facts distance calculated from the simulated time series, which corresponds to that particular set of model parameters. Let θ denote the vector of model parameters to be estimated, Equation 3.6

can be rewritten as:

$$D(\theta) = w_1 * KS(\theta) + w_2 * \Delta_V(\theta) + w_3 * \Delta_{ACF1}(\theta) + w_4 * \Delta_{ACF2}(\theta) \quad (3.11)$$

The smaller $D(\theta)$ is, the more realistic the simulation is. Thus $D(\theta)$ serves as the loss function that the calibration method aims to minimise by finding an optimal set of model parameters. Let Θ denote the admissible set for model parameter vector θ , the calibration target is to find the optimal model parameter vector $\hat{\theta}$ that minimises the stylised facts distance:

$$\hat{\theta} = \arg \min_{\theta \in \Theta} D(\theta) \quad (3.12)$$

The calibration method is presented in detail in subsequent sections.

3.3.3 Surrogate Modelling Calibration Method

Specification

We propose a novel surrogate modelling approach to calibrate the model parameters. That is, to find an optimal set of model parameters to minimise the loss function - stylised facts distance. The approach is adapted from the machine learning surrogate modelling methodology in [LRS18]. The original methodology mainly trains an XGBoost classifier to classify a positive calibration or negative calibration [LRS18], while in our approach we train an XGBoost regressor to directly approximate the mapping from model parameters to stylised facts distance. Another novelty in our approach is the introduction of an exploration-exploitation mechanism in selecting new points in parameter space. The advantage of the surrogate modelling approach is that it significantly reduces the computational cost of calibrating an ABM with several model parameters. Using only a limited budget (N) of ABM evaluations, the XGBoost surrogate model is proved to be a fairly good approximation of the mapping from model parameters to the target stylised facts distance. The surrogate provides a costless way to predict the model's response and allows for efficiently finding the optimal point in the model parameter space that minimises the loss function.

With regard to the Extended Chiarella simulation model, during calibration the model can be represented as a mapping $M: \theta \rightarrow D(\theta)$ from a vector of model parameters θ into the stylised facts distance $D(\theta)$. Generally the number of parameters ranges from several to dozens. In our case, we have three

parameters to calibrate: κ , β , and σ_N . The values for parameters γ and α are fixed in advance. Parameter α indicates the typical horizon of trend computation for momentum traders. Following the original Extended Chiarella model [MCB20], α is relatively small to represent a relatively low frequency momentum trader. Here we take the value 0.1 for α . Other values of α are also tested, and similar results are obtained as long as α is smaller than 0.4. For larger α value, the low frequency momentum traders will become high frequency momentum traders, which would be an extension of the Extended Chiarella model. For γ , the value is fixed to be 10. Since γ appears in the same term as β in Equation 3.5, the model calibration will have difficulties pinning down the value of parameters γ and β if both parameters are to be estimated. As a result, fixing γ significantly improves the robustness of the model calibration process. We have also tested other values of γ and the results are similar.

Instead of classifying a positive calibration or negative calibration as in the original method [LRS18], here our calibration objective is obvious: finding an optimal set of values for κ , β , and σ_N to minimise the stylised facts distance. Our calibration measure is a real-valued number providing a quantitative assessment of the realism of the simulated market. An XGBoost machine learning surrogate model is trained to approximate the mapping from model parameters to stylised facts distance, and help to find the optimal parameters. Detailed implementations are presented below.

Implementation

The whole calibration methodology proceeds with the following steps.

Step 1. Initialisation

The process starts with drawing a relatively large pool of parameter combinations. Each combination is a vector containing a value for each parameter: κ , β , and σ_N . The requirement is that the pool should be a good approximation of the whole parameter space. In our experiments, we use Sobol sampling [MC94] to implement the sampling routine. Sobol sampling is capable of guaranteeing uniformity of distribution even though the sampled set has a small number of points. It is shown that in terms of uniformity properties, Sobol sequences outperform the sequences generated by other sampling techniques such as Latin Hypercube sampling [KAS15]. Other advantages of Sobol sampling include efficient implementation and faster sampling speed. The sampling number and quality of the pool of parameter combinations dominate the ability of the whole process to learn a good surrogate model. As a result, a

faster sampler is preferred so that more samples can be obtained in limited computational time. In our experiments, we use Sobol sampling to sample 16384 (2^{14}) points as the pool of parameter combinations.

After the pool of parameter combinations is obtained, an initial set of samples is chosen randomly from the pool as the initial training set. Each point in the set of initialization samples is evaluated by actually running the agent-based model with the corresponding parameter combination. The corresponding stylised facts distance is calculated, which will act as the true label associated with that point in the training set. After this step we obtain an initial training set of parameter combinations with corresponding stylised facts distances as labels. The size of the initial training set in our experiments is 2000.

Step 2. Surrogate Model Training

Given a training set of evaluated parameter combinations and corresponding stylised facts distances, an XGBoost regressor is learned over the training set in order to build the surrogate model. The input is the vector of model parameters to be calibrated, which in our experiments has dimension 3. The output is the stylised facts distance, which is a scalar. The XGBoost regressor is trained to fit the mapping from model parameters to stylised facts distance. Implemented under the Gradient Boosting framework, XGBoost is a machine learning algorithm designed to be highly flexible, efficient and portable [CG16]. The XGBoost algorithm builds an ensemble of simple decision trees, which are subsequently aggregated to improve the prediction performance. Details about the XGBoost algorithm can be found in [CG16].

Remark 1. One difficulty in training the XGBoost regressor is how to tune the hyper-parameters of the XGBoost algorithm. Here we employ the Bayesian Optimisation method based on Gaussian Process [SLA12] to fine-tune the hyper-parameters of XGBoost. In the framework of Bayesian Optimization, the performance of the XGBoost regressor is modelled as a sample from a Gaussian Process. The Gaussian Process then guides the exploration of the hyper-parameter space and helps to find an optimal set of hyper-parameters of XGBoost. Note that here the focus is the exploration of the hyper-parameter space of the XGBoost machine learning algorithm, which is different from the parameter space of the agent-based model. Details about Bayesian Optimisation with Gaussian Process can be found in [SLA12].

Remark 2. Another technique we have applied during the training process is to clip the values of training labels into a relatively small range. Specifically, for any training point, if the calculated stylised facts distance is too large, the distance value will be replaced with a relatively smaller value. The logic for this operation is that our calibration focus is the area of the model parameter space where stylised facts distances are small. In other words, parameter combinations with large stylised facts distances are of no interest and it is not important to have precise surrogate model approximations of stylised facts distances in those areas. If the range of distance value is not restricted, some large input labels would utterly bias the learning process of the XGBoost surrogate model. In this circumstance, the XGBoost surrogate model would wrongly try to fit those large outlier values. Consequently, the model is no longer a good approximation of the mapping from agent-based model parameters to stylised facts distance. By restricting the values of training labels to a relatively smaller range, the bias is successfully corrected and experimental results show that the surrogate model predicts the stylised facts distances quite precisely. Since optimal stylised facts distances in our experiments are generally smaller than 0.8, we restrict the training labels to the range of $(0, 1]$.

Step 3. Surrogate Model Prediction

Once the surrogate XGBoost model is trained, it is used to predict the stylised facts distances over the set of remaining unlabelled parameter combinations. That is, the stylised facts distances that would be generated if those unlabelled parameter combinations were to be used in agent-based model simulation. The surrogate model predictions are used to guide further exploration of the model parameter space, as specified in the next step.

Step 4. Supplement Training Set

Given the XGBoost surrogate model predictions, a subset of the unlabelled parameter combinations is drawn from the pool sampled in the first step. These unlabelled parameter combinations are evaluated in the agent-based model simulation, and the true stylised facts distances are calculated and subsequently assigned as the true labels of these samples. This set of newly labelled points is then added to the training set of labelled parameter combinations. There are two critical issues in this process: how many new points are to be drawn into the subset of the unlabelled parameter combinations and how to select the points. For the first issue, the original method recommends that the number of new points to be drawn in this stage is the logarithm of the computational budget [LRS18]. However, in our experiments, it turns out that this rule would draw too few samples. Consequently, more iterations are required,

which significantly reduces the computational efficiency. After lots of testing, in our method around 2% of total samples are drawn at each iteration, which in our case is 300 points. As for how to choose those unlabelled parameter combinations, we utilise the predictions made by the XGBoost surrogate model. The unlabelled points are sorted according to the predicted stylised facts distances. The points with smaller predicted stylised facts distances are selected, as the optimal parameter combination is more likely to exist among or near those points. However, not all points are selected according to predicted stylised facts distances. We also randomly choose some points from the unlabelled parameter combinations to avoid occasional bias induced by the XGBoost surrogate model. Specifically, 200 points are selected according to the "small predicted stylised facts distance" principle and 100 points are selected randomly.

Remark 3. The way of selecting the subset of unlabelled points implements an exploration-exploitation mechanism, which is a novel aspect of our methodology compared to the original method. Two-thirds of the points are selected according to the predicted stylised facts distances. In this way, we exploit the information given by the XGBoost surrogate model and the model intelligently helps to direct the selection of new samples. The reason is that the optimal parameter combination is more likely to exist among the points with smaller predicted distances. This is true as long as the surrogate is a good approximation of the real mapping from parameter combinations to stylised facts distances. However, exploration is also essential if the aim is to find a global minimum. There might be multiple local minimums inside the model parameter space. Completely exploiting the information given by the surrogate model may get the method stuck in a local minimum. Randomly sampling some unlabelled points helps to avoid this problem and contributes to a better exploration of the whole model parameter space.

Step 5. Iterations

After the training set is supplemented by the newly drawn samples and the corresponding labels, a new XGBoost surrogate model is trained using the new training set. The procedure is identical to previous steps. In other words, the previous "training - predicting - supplement"(step 2 to step 4) process is repeated until the budget of computational time is reached. In our experimental settings, we find that generally less than 5 iterations are required to build a fairly good surrogate model, whose stylised facts distance prediction error is less than 5% at the predicted optimal point.

3.4 Results and Evaluation

The whole methodology is run on 75 stocks from three exchanges: NASDAQ, LSE, and HKEX. The main results are the stylised facts distances of the calibrated model, compared with the baseline stylised facts distances where model parameters are estimated by the Expectation-Maximisation algorithm. We also present the error rate of XGBoost surrogate model prediction for stylised facts distance at the predicted optimal point to evaluate the surrogate model prediction accuracy. Finally, we show the methodology's capability of reproducing autocorrelation patterns in return series, which is an advantage of XGB-Chiarella methodology over other traditional models.

3.4.1 Stylised Facts Distance

Table 3.3 shows the stylised facts distances of the calibrated model for all stocks, averaged by trading days. The corresponding standard deviations are also presented. For stocks in HKEX, the XGB-Chiarella method outperforms the Expectation-Maximisation estimation algorithm for all 25 stocks. As for stocks in NASDAQ, in 22 out of 25 stocks the performance of the XGB-Chiarella method is better than the Expectation-Maximisation estimation method. On average, the XGB-Chiarella method achieves around 10% smaller stylised distances than the Expectation-Maximisation baseline in these two exchanges, showing that the simulated market is calibrated to be more realistic. The standard deviations of the stylised facts distance are also generally a bit smaller for the proposed XGB-Chiarella method in NASDAQ and HKEX. For stocks in LSE, the performance of the XGB-Chiarella method is not as good as the performance in the other two markets, with a smaller advantage over the Expectation-Maximisation baseline estimation algorithm. However, it is still valid to say that the XGB-Chiarella methodology outperforms the baseline since smaller stylised facts distance is achieved in more than half of the stocks. Overall, the results show that the proposed XGB-Chiarella methodology outperforms the Expectation-Maximisation baseline in terms of stylised facts distance and the realism of the market simulation.

Results in Table 3.3 are the mean and standard deviation of stylised facts distances across all trading days in our data span. To scrutinise each trading day for individual stock, Table 3.4 shows the number of trading days when the XGB-Chiarella methodology outperforms the Expectation-Maximisation baseline in terms of stylised facts distance, and the percentage of these trading days out of all trading days. In Table 3.4, column "Better" represents the number of trading days when the performance of

NASDAQ	XGB-Chiarella	EM-algorithm	LSE	XGB-Chiarella	EM-algorithm	HKEX	XGB-Chiarella	EM-algorithm
AAL	0.32 (0.07)	0.36 (0.09)	AAL.L	0.26 (0.17)	0.26 (0.09)	0005.HK	0.4 (0.04)	0.42 (0.05)
AAPL	0.25 (0.07)	0.28 (0.08)	AV.L	0.27 (0.05)	0.28 (0.07)	0027.HK	0.43 (0.07)	0.45 (0.07)
AFRM	0.28 (0.08)	0.34 (0.09)	BATS.L	0.24 (0.04)	0.25 (0.05)	0175.HK	0.43 (0.05)	0.45 (0.06)
AMD	0.29 (0.08)	0.32 (0.08)	BHP.L	0.27 (0.26)	0.26 (0.13)	0388.HK	0.35 (0.07)	0.37 (0.07)
CMCSA	0.32 (0.08)	0.36 (0.09)	BP.L	0.21 (0.06)	0.22 (0.05)	0700.HK	0.32 (0.06)	0.35 (0.06)
CSCO	0.28 (0.07)	0.3 (0.09)	BRBY.L	0.31 (0.07)	0.32 (0.07)	0836.HK	0.39 (0.07)	0.43 (0.09)
CSX	0.32 (0.07)	0.35 (0.08)	DGE.L	0.24 (0.06)	0.24 (0.07)	0916.HK	0.42 (0.08)	0.45 (0.08)
DKNG	0.31 (0.09)	0.36 (0.09)	EZJ.L	25.5 (182.0)	25.2 (179.6)	1024.HK	0.35 (0.08)	0.39 (0.08)
FB	0.28 (0.08)	0.31 (0.08)	GLEN.L	0.24 (0.07)	0.25 (0.1)	1171.HK	0.42 (0.07)	0.44 (0.08)
HBAN	0.39 (0.09)	0.36 (0.08)	GSK.L	0.24 (0.06)	0.25 (0.08)	1211.HK	0.37 (0.08)	0.41 (0.08)
HON	0.28 (0.09)	0.3 (0.1)	HSBA.L	0.23 (0.09)	0.27 (0.36)	1299.HK	0.4 (0.07)	0.43 (0.08)
HUT	0.27 (0.07)	29.5 (206.5)	IAG.L	26.7 (191.0)	25.7 (183.4)	1772.HK	0.39 (0.09)	0.44 (0.09)
INTC	0.28 (0.07)	0.31 (0.08)	JET.L	0.35 (0.27)	0.32 (0.08)	1918.HK	0.44 (0.08)	0.49 (0.1)
JD	0.34 (0.09)	0.38 (0.09)	LGEN.L	0.28 (0.05)	0.29 (0.08)	1919.HK	0.44 (0.07)	0.46 (0.08)
LCID	0.29 (0.07)	6.35 (42.4)	LLOY.L	0.23 (0.06)	0.24 (0.07)	2020.HK	0.35 (0.09)	0.39 (0.08)
MRNA	0.32 (0.09)	0.35 (0.09)	NG.L	0.25 (0.06)	0.25 (0.07)	2269.HK	0.37 (0.08)	0.41 (0.09)
MSFT	0.26 (0.07)	0.29 (0.08)	PRU.L	0.31 (0.07)	0.33 (0.06)	2318.HK	0.37 (0.07)	0.4 (0.07)
MU	0.34 (0.09)	0.39 (0.1)	RDSA.L	0.3 (0.57)	0.27 (0.23)	2331.HK	0.38 (0.07)	0.42 (0.09)
NVDA	0.27 (0.08)	0.3 (0.07)	REL.L	0.35 (0.05)	0.37 (0.05)	2333.HK	0.4 (0.07)	0.44 (0.08)
SOFI	0.3 (0.08)	0.36 (0.1)	RIO.L	0.27 (0.24)	0.26 (0.11)	3690.HK	0.34 (0.08)	0.38 (0.07)
TLRY	0.29 (0.08)	0.34 (0.09)	RR.L	0.25 (0.08)	0.28 (0.09)	3968.HK	0.39 (0.06)	0.42 (0.07)
TSLA	0.36 (0.2)	0.28 (0.08)	SMT.L	0.23 (0.05)	0.26 (0.05)	9618.HK	0.35 (0.07)	0.38 (0.07)
UAL	0.33 (0.07)	0.37 (0.09)	STAN.L	0.27 (0.05)	0.27 (0.06)	9888.HK	0.38 (0.09)	0.4 (0.07)
UBER	0.29 (0.07)	0.33 (0.08)	ULVR.L	0.23 (0.08)	0.23 (0.07)	9988.HK	0.33 (0.06)	0.36 (0.07)
WISH	0.38 (0.14)	0.29 (0.07)	VOD.L	0.24 (0.07)	0.25 (0.12)	9999.HK	0.36 (0.07)	0.4 (0.07)

Table 3.3 Average stylised facts distance comparison between XGB-Chiarella calibration and Expectation-Maximisation method. The stylised facts distance is the average across all trading days for each stock; values in parentheses are corresponding standard deviations. There are several rare cases where distances are much larger, showing algorithms fail to obtain a reasonable set of parameters during the parameter estimation process.

the XGB-Chiarella method is better than baseline Expectation-Maximisation estimation, while column "Total" represents the number of all trading days in our experiments. For the performance in different exchanges, results here are similar to the results in Table 3.3. For most stocks in NASDAQ and HKEX, the XGB-Chiarella method performs better than the Expectation-Maximisation baseline for more than 85% of total trading days. As for stocks listed in LSE, this percentage is slightly lower but is still on average more than 70%. On the whole, for most trading days the XGB-Chiarella method is capable of creating a simulated financial market with smaller stylised facts distance, indicating that the proposed XGB-Chiarella method is able to generate more realistic artificial financial markets.

3.4.2 Surrogate Model Performance

To evaluate the performance of the surrogate model approximation, we present the prediction error rate when the surrogate model is used to predict the real stylised facts distance. Table 3.5 shows a comparison between the surrogate model predicted and actual stylised facts distance at the surrogate predicted optimal point in model parameter space for one trading day. Very similar results are achieved

NASDAQ	Better	Total	Percentage	LSE	Better	Total	Percentage	HKEX	Better	Total	Percentage
AAL	46	50	92.00%	AAL.L	43	52	82.69%	0005.HK	37	47	78.72%
AAPL	41	50	82.00%	AV.L	37	52	71.15%	0027.HK	37	47	78.72%
AFRM	46	50	92.00%	BATS.L	38	52	73.08%	0175.HK	34	47	72.34%
AMD	39	50	78.00%	BHP.L	39	52	75.00%	0388.HK	39	47	82.98%
CMCSA	46	50	92.00%	BP.L	40	52	76.92%	0700.HK	46	47	97.87%
CSCO	42	50	84.00%	BRBY.L	48	51	94.12%	0836.HK	45	47	95.74%
CSX	44	50	88.00%	DGE.L	33	52	63.46%	0916.HK	41	47	87.23%
DKNG	47	50	94.00%	EZJ.L	49	52	94.23%	1024.HK	45	47	95.74%
FB	41	50	82.00%	GLEN.L	34	52	65.38%	1171.HK	38	47	80.85%
HBAN	14	50	28.00%	GSK.L	30	52	57.69%	1211.HK	45	47	95.74%
HON	44	50	88.00%	HSBA.L	31	52	59.62%	1299.HK	42	47	89.36%
HUT	44	50	88.00%	IAG.L	39	52	75.00%	1772.HK	45	47	95.74%
INTC	44	50	88.00%	JET.L	37	52	71.15%	1918.HK	39	47	82.98%
JD	44	50	88.00%	LGEN.L	37	52	71.15%	1919.HK	40	47	85.11%
LCID	47	50	94.00%	LLOY.L	33	52	63.46%	2020.HK	42	47	89.36%
MRNA	47	50	94.00%	NG.L	31	52	59.62%	2269.HK	42	47	89.36%
MSFT	42	50	84.00%	PRU.L	45	52	86.54%	2318.HK	43	47	91.49%
MU	45	50	90.00%	RDSA.L	36	52	69.23%	2331.HK	46	47	97.87%
NVDA	43	50	86.00%	REL.L	44	51	86.27%	2333.HK	42	47	89.36%
SOFI	44	50	88.00%	RIO.L	38	52	73.08%	3690.HK	45	47	95.74%
TLRY	45	50	90.00%	RR.L	40	52	76.92%	3968.HK	43	47	91.49%
TSLA	19	50	38.00%	SMT.L	45	52	86.54%	9618.HK	44	47	93.62%
UAL	46	50	92.00%	STAN.L	38	52	73.08%	9888.HK	43	47	91.49%
UBER	47	50	94.00%	ULVR.L	29	52	55.77%	9988.HK	44	47	93.62%
WISH	14	50	28.00%	VOD.L	29	52	55.77%	9999.HK	44	47	93.62%

Table 3.4 Percentage of trading days when XGB-Chiarella calibration outperforms Expectation-Maximisation method.

for other trading days. The results show that the XGBoost surrogate model is a very accurate proxy of the true model around the predicted optimal point for each stock on each trading day, with predicted stylised facts distance very close to the actual stylised facts distance generated in the agent-based model simulation. For most stocks, errors between predicted distance and actual distance are less than 10%. It is also shown that the accuracy of XGBoost surrogate model prediction is similar across the three exchanges, which indicates the robustness of the XGB-Chiarella methodology.

Apart from the prediction accuracy at the predicted optimal point, we also examine the prediction accuracy around the optimal point as a sensitivity analysis. Figure 3.4 shows the comparison between XGBoost surrogate model prediction and actual simulated stylised facts value. For each sub-graph, one model parameter is assigned a series of values across the given range during calibration. Other model parameters are fixed to the optimal value. In this way, a set of parameter combinations is obtained, with different values for that particular model parameter. Then predicted and actual stylised facts distance are calculated and compared, as shown in Figure 3.4. The gray lines are the actual stylised facts distances while blue lines represent the XGBoost surrogate model prediction for the stylised facts distances. It is shown that the XGBoost prediction line fits the actual simulated stylised facts distance line very well, especially for parameter "sigma_N". For each parameter, there is a unique minimal

NASDAQ	Predict	Actual	Error	LSE	Predict	Actual	Error	HKEX	Predict	Actual	Error
AAL	0.33	0.37	10.81%	AAL.L	0.28	0.27	3.70%	0005.HK	0.39	0.42	7.14%
AAPL	0.25	0.27	7.41%	AV.L	0.35	0.36	2.78%	0027.HK	0.42	0.43	2.33%
AFRM	0.48	0.49	2.04%	BATS.L	0.25	0.24	4.17%	0175.HK	0.37	0.4	7.50%
AMD	0.39	0.4	2.50%	BHP.L	0.26	0.26	0.00%	0388.HK	0.36	0.36	0.00%
CMCSA	0.45	0.46	2.17%	BP.L	0.16	0.16	0.00%	0700.HK	0.27	0.26	3.85%
CSCO	0.39	0.4	2.50%	BRBY.L	0.3	0.29	3.45%	0836.HK	0.29	0.31	6.45%
CSX	0.31	0.31	0.00%	DGE.L	0.17	0.18	5.56%	0916.HK	0.53	0.55	3.64%
DKNG	0.26	0.27	3.70%	EZJ.L	0.22	0.21	4.76%	1024.HK	0.28	0.26	7.69%
FB	0.27	0.27	0.00%	GLEN.L	0.27	0.27	0.00%	1171.HK	0.37	0.36	2.78%
HBAN	0.4	0.38	5.26%	GSK.L	0.4	0.39	2.56%	1211.HK	0.26	0.26	0.00%
HON	0.25	0.25	0.00%	HSBA.L	0.18	0.19	5.26%	1299.HK	0.33	0.35	5.71%
HUT	0.3	0.31	3.23%	IAG.L	0.26	0.27	3.70%	1772.HK	0.35	0.35	0.00%
INTC	0.36	0.35	2.86%	JET.L	0.37	0.37	0.00%	1918.HK	0.3	0.34	11.76%
JD	0.33	0.33	0.00%	LGEN.L	0.25	0.25	0.00%	1919.HK	0.33	0.36	8.33%
LCID	0.28	0.26	7.69%	LLOY.L	0.25	0.27	7.41%	2020.HK	0.35	0.35	0.00%
MRNA	0.32	0.33	3.03%	NG.L	0.2	0.2	0.00%	2269.HK	0.32	0.34	5.88%
MSFT	0.29	0.29	0.00%	PRU.L	0.33	0.35	5.71%	2318.HK	0.31	0.31	0.00%
MU	0.42	0.43	2.33%	RDSA.L	0.23	0.23	0.00%	2331.HK	0.29	0.28	3.57%
NVDA	0.2	0.2	0.00%	REL.L	0.3	0.29	3.45%	2333.HK	0.32	0.35	8.57%
SOFI	0.32	0.32	0.00%	RIO.L	0.25	0.25	0.00%	3690.HK	0.25	0.23	8.70%
TLRY	0.27	0.28	3.57%	RR.L	0.27	0.32	15.63%	3968.HK	0.29	0.31	6.45%
TSLA	0.44	0.44	0.00%	SMT.L	0.21	0.2	5.00%	9618.HK	0.25	0.24	4.17%
UAL	0.24	0.24	0.00%	STAN.L	0.23	0.23	0.00%	9888.HK	0.34	0.33	3.03%
UBER	0.27	0.27	0.00%	ULVR.L	0.21	0.21	0.00%	9988.HK	0.28	0.28	0.00%
WISH	0.52	0.48	8.33%	VOD.L	0.15	0.21	28.57%	9999.HK	0.37	0.37	0.00%

Table 3.5 Comparison between surrogate model predicted and actual stylised facts distance at the surrogate predicted optimal point. The results shown here are for one trading day (2nd of November, 2021), while results for other trading days are very similar.

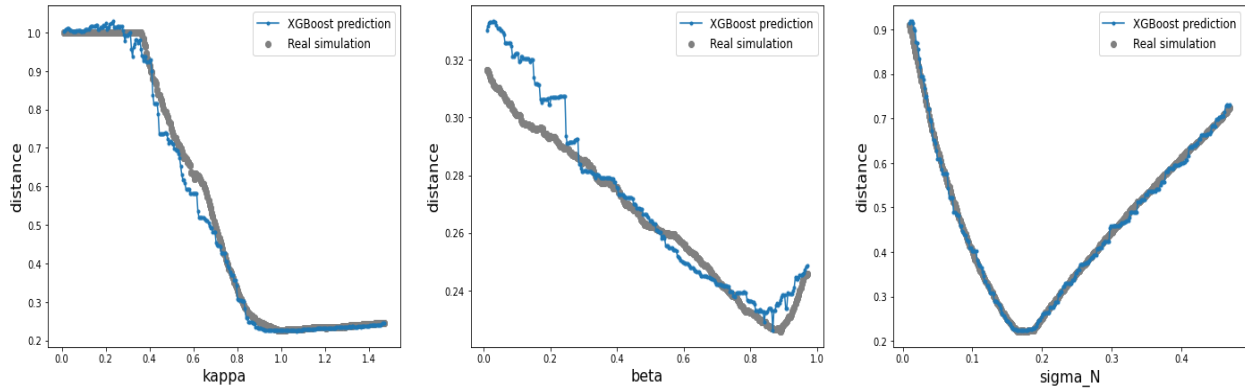


Figure 3.4 Sensitivity analysis around the optimal point predicted by the XGBoost surrogate model. This surrogate model corresponds to the trading day on the 29th of September, 2021 for stock FB. For each sub-graph, model parameters are fixed to the optimal value except for the parameter in x-axis. Gray lines represent the actual stylised facts distances while blue lines represent the XGBoost surrogate model prediction for the stylised facts distances.

value in the graph, which corresponds to the optimal point predicted by the surrogate model. The results here indicate that the surrogate model approximates the real stylised facts distance accurately and is capable of finding the true optimal model parameter combination that is able to generate realistic financial market simulations.

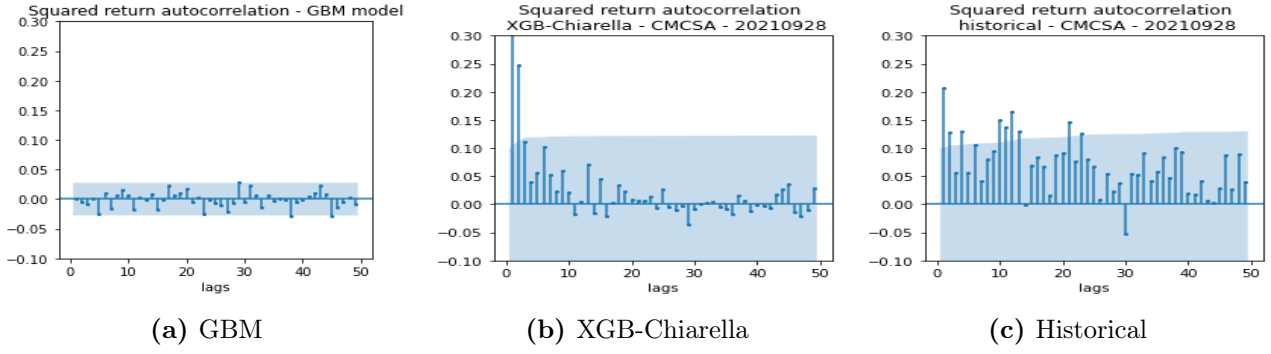


Figure 3.5 (a): Autocorrelation of squared returns generated by a GBM for CMCSA on Sept 28th, 2021. (b): Autocorrelation of squared returns generated by XGB-Chiarella method for CMCSA on Sept 28th, 2021. (c): Historical autocorrelation of squared returns for CMCSA on Sept 28th, 2021.

3.4.3 Autocorrelations

One unique advantage of agent-based financial market simulation is the capability of reproducing autocorrelation patterns of returns. Currently, most literature in financial econometrics model financial price series as Geometric Brownian Motion (GBM). For example, the Black-Scholes model for option pricing explicitly assumes the stock price follows a GBM. However, this case is not true in real financial markets. One obvious anomaly is the autocorrelation patterns in return series, such as the volatility clustering phenomenon shown in Figure 3.3. Our XGB-Chiarella method has an advantage over the GBM model in that it can successfully reproduce the autocorrelation patterns in financial return time series. Figure 3.5 shows autocorrelations of squared returns of a GBM model and that of a simulation under XGB-Chiarella methodology for one stock. The real historical autocorrelation of squared returns on the same day for that stock is also presented. It is obvious that our XGB-Chiarella method outperforms the traditional GBM model in terms of the replication of realistic autocorrelation patterns, especially for small time lags. For small time lags, significant positive autocorrelations of squared returns exist in historical financial market data. This stylised fact is successfully reproduced in the XGB-Chiarella simulation, while the autocorrelations of squared returns generated by a GBM are basically close to zero regardless of time lags. In fact, there are no specific patterns in autocorrelations generated by the GBM model. Our experiments show that simulated results in other stocks for other trading days are similar to what is shown here.

Successful reproduction of autocorrelation patterns gives the XGB-Chiarella method an advantage in the realistic simulation of multiple price series, which can be used in scenario simulations for risk management. When calculating simulation-based risk metrics such as value at risk (VaR), lots of institutions are still using GBM to simulate future scenarios. As shown in Figure 3.5, the return series

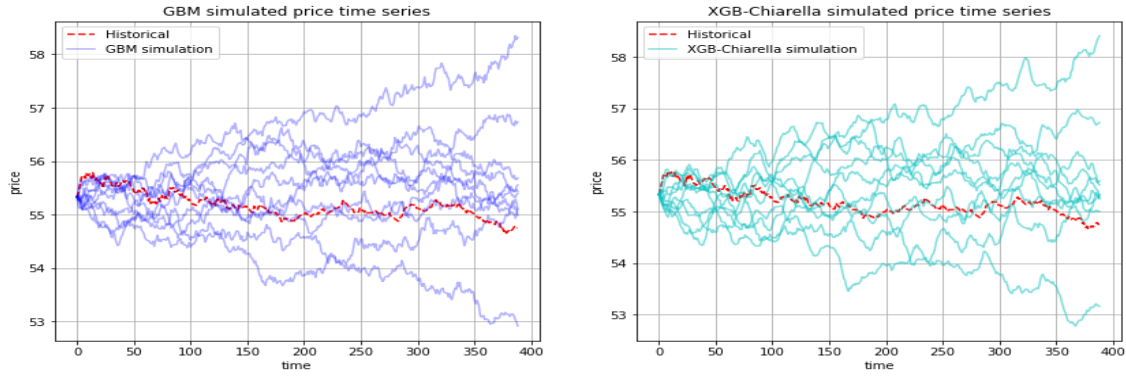


Figure 3.6 Simulated price scenarios generated by GBM and XGB-Chiarella, respectively.

simulated by the GBM model lacks a realistic autocorrelation pattern, which would undermine the credibility of the risk metric calculation. Our proposed method is a better alternative. The proposed XGB-Chiarella method extracts fundamental values and calibrates the model parameters using historical data. With the same set of calibrated model parameters, the agent-based model is capable of generating multiple different scenarios by changing the input fundamental value series. For example, if GBM scenarios are used as fundamental value, the agent-based model is able to reproduce similar price series as the GBM model, but with realistic autocorrelation patterns of returns. Figure 3.6 shows the price scenarios generated by GBM and XGB-Chiarella method, respectively. Here the fundamental values in the XGB-Chiarella model are the price scenarios generated by the GBM. Figure 3.7 shows the corresponding average autocorrelation of squared returns for the two cases. It is shown that the two models can generate similar price scenarios, but the XGB-Chiarella model is able to reproduce much more realistic autocorrelation patterns. Specifically, for small time lags the XGB-Chiarella model reproduces significant positive autocorrelation of squared returns, which is consistent with empirical data. In contrast, there are no significant autocorrelation patterns for GBM scenarios. It is believed that such agent-based price series simulation could provide a richer environment than GBM model for risk management practices such as VaR calculation. This topic is addressed in later chapters in this thesis, where we use agent-based models to simulate flash crashes and explore better practices for risk management.

3.5 Summary

In this chapter, an innovative approach called XGB-Chiarella is proposed to generate realistic intra-day artificial financial price data to provide insight into the intra-day price formation process. To the best

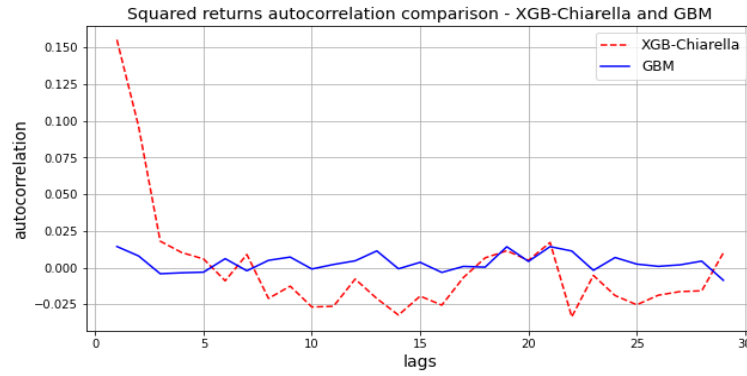


Figure 3.7 Average autocorrelation of squared returns associated with the simulated price scenarios in Figure 3.6. Blue line represents the GBM simulated scenarios while red dashed line represents the XGB-Chiarella simulated scenarios.

of our knowledge, this is the first extension of the Chiarella model to generate minute-level intra-day financial market simulation. The approach utilises agent-based modelling techniques. The underlying simulation model has only three agents: one for the fundamental trader, one for the momentum trader, and one for the noise trader. The model is simulated and model parameters are calibrated by an XGBoost machine learning surrogate. A novel exploration-exploitation mechanism is included in the model calibration process. The proposed methodology is tested on 75 stocks from three exchanges: NASDAQ, LSE, and HKEX. In terms of stylised facts distance, the proposed XGB-Chiarella method is able to generate more realistic financial market simulations than the original Expectation-Maximisation estimation algorithm. This is true in nearly all the stocks from three exchanges. Despite the fact that the methodology is based on a model with only three agents, the XGB-Chiarella methodology successfully generates very realistic financial market simulations. This indicates that one agent per category seems to be sufficient to capture the intra-day price formation process for the time scale (minutes) chosen in this chapter. The very simple model structure not only accelerates the simulation process in terms of computational cost, but also enables us to scrutinise the intra-day price formation process, such as the trend and value effects. The results provide support for the existence of a universal intra-day price formation mechanism. The realistic simulated intra-day financial market indicates that trend and value effects, as well as noise trading, are indispensable to the intra-day price formation process.

We also show that in the process of calibration, the XGBoost surrogate model is an accurate approximation of the true model. At the predicted optimal point for each stock on each trading day, the surrogate model prediction error is mostly smaller than 10%. The machine learning surrogate is capable of intelligently directing the exploration of model parameter space. The exploration-exploitation

mechanism is also introduced in the model calibration process. A practical application of the proposed methodology is also presented.

With the introduction of the XGB-Chiarella method, we have developed a versatile agent-based model for generating synthetic financial time series data, along with a robust calibration technique. The next logical step is to explore how this method can be extended to address related challenges. In this thesis, the optimal hedging problem is chosen as the next challenge to be addressed, utilising the power of agent-based models. Chapter 4 introduces the Chiarella-Heston model, an extension of the Extended Chiarella model that underlies the XGB-Chiarella method discussed in this chapter. Integrated with deep reinforcement learning algorithms, the Chiarella-Heston model proposed in the next chapter is employed to tackle the optimal hedging problem.

Chapter 4

Chiarella-Heston model for optimal deep hedging

4.1 Introduction

This is the second main chapter of this thesis, where we propose the Chiarella-Heston model to address the optimal hedging problem. The proposed Chiarella-Heston model is a generalisation of both the Extended Chiarella model and the Heston stochastic volatility model. In this way, the methodology in this chapter is also an extension of the XGB-Chiarella method in Chapter 3. We validate the model using solid statistical metrics and demonstrate that a deep hedging agent, trained with the DDPG algorithm on data from the proposed model, outperforms baseline models across various transaction cost levels and succeeds in real trading scenarios. Overall, the innovative approach in this chapter offers a promising strategy for improving risk management and hedging in quantitative finance.

4.1.1 Motivation

In recent years, the quantitative finance industry has emerged as a leading field in the application of data science. With a rapidly growing volume of data and increased computing power, algorithms with high time complexity can now be applied to a range of practical applications. In terms of hedging derivatives, financial institutions have begun to explore deep learning algorithms as a replacement for traditional sensitivity-based approaches. In an idealised world with a complete market, perfect hedging

is accessible based on the Black-Scholes model [BS73]. However, the real market always involves various frictions, such as transaction costs and slippage, making hedging optimisation a much more challenging problem. Since the analytic formulas are not available in such a market, it may be necessary to hedge derivatives based on non-closed-form computational procedures or human expertise.

Deep hedging [HLJB19] presents an innovative approach to enhance hedging operations. Within this framework, a neural network is employed to hedge derivatives effectively by minimising a suitable risk measure. Deep hedging utilises deep reinforcement learning agents to derive hedging strategies. Leveraging the powerful representational capabilities of neural networks and advanced optimisation algorithms, training a neural network is likely to achieve an optimal hedge. As a data-driven methodology, deep hedging requires a large amount of data input for training the deep reinforcement learning agent. The majority of the literature has mainly been training and testing deep reinforcement learning agents using synthetic data. For example, [HLJB19] uses a dataset generated by the Heston stochastic volatility model, while [CCHP21] utilises Geometric Brownian Motion and SABR stochastic volatility model. The underlying assumption is that an agent trained by synthetic data can be transferred to the real environment successfully. This assumption requires that the synthetic data used for training the agents are able to closely mimic real financial market dynamics. However, this is rarely true in practice. For example, the commonly used Geometric Brownian Motion assumes fixed volatility of the underlying stock, while in reality the volatility constantly changes and exhibits a volatility clustering stylised fact. To enhance the performance of deep hedging agents in real financial markets, advanced financial market simulators are essential to generate a realistic training dataset.

Surprisingly, the literature on deep hedging barely uses synthetic data generated by agent-based models. Agent-based models (ABMs) are computational models used to simulate the actions and interactions of autonomous agents in a network or environment to analyze complex systems. In the context of financial markets, agents could represent traders, investors, or financial institutions, each with their own behaviours, strategies, and decision-making processes. With huge potential academic and industrial value, agent-based financial market simulation has gained extensive research attention in recent years. Agent-based models are particularly useful for generating synthetic financial market data that closely resemble real market dynamics. Empirical financial time series data exhibit various stylised facts [Con01]. Calibrated to reproduce these stylised facts, agent-based models present a clear advantage over traditional equilibrium-based mathematical models when it comes to simulating financial markets.

As mentioned in Section 1.2, the primary **challenge** that we address in the chapter is:

- C2: How to develop and calibrate an agent-based model to produce realistic financial time series data, and then effectively utilise this model to optimise the performance of a deep hedging agent in real-world market conditions.

4.1.2 Our Contributions

We address the above challenge by proposing the Chiarella-Heston model, a novel agent-based model. The model is calibrated to reproduce stylised facts exhibited by empirical data. Realistic financial time series data are generated by the model and used for training a deep hedging agent. To demonstrate the practical value of our methodology, we test the trained deep hedging agent on empirical data. The main **contributions** of our work are the following:

- An innovative agent-based model called Chiarella-Heston model is proposed, comprising momentum traders, fundamental traders, and volatility traders. The model is a generalisation of both the Extended Chiarella model in [MCB20] and the Heston stochastic volatility model in [Hes93]. In particular, the volatility trader in the model is an innovative type of trader whose behaviour is inspired by the Heston stochastic volatility model. The proposed model is calibrated to reproduce empirical stylised facts including fat tails of returns and autocorrelation patterns of returns. Using popular validation metrics we show that our agent-based model is capable of producing more realistic financial time series than baseline models including the Extended Chiarella model and the Heston stochastic volatility model.
- With the calibrated agent-based financial market simulator, a training dataset is generated to train the deep hedging agent for optimal hedging strategies under different transaction cost levels. Empirical results show that at most transaction cost levels, the deep hedging agent trained by data generated by the proposed Chiarella-Heston model outperforms the baseline, where the deep hedging agent is trained by data generated by Geometric Brownian Motion. The deep hedging agent utilises an improved version of the Deep Deterministic Policy Gradient algorithm [LHP⁺15] and is trained towards maximising profits and minimising risks. In addition, our testing process is carried out using empirical data, showing that the trained deep hedging agent is able to perform well in realistic trading environments.

We provide a literature review of applications of deep reinforcement learning in developing hedging strategies in Section 2.4. However, we are aware of no existing work involving agent-based models in the

deep hedging framework, which we aim to address in this chapter. In terms of the deep reinforcement learning method, we adopt the revised DDPG algorithm in [CCHP21]. The expected value of the hedging cost and the expected value of the square of the hedging cost are tracked by two different Q-functions. In this way, the standard deviation of the hedging cost can be included in the objective function. The learning objective of the deep hedging agent is to minimise a function equal to the mean hedging cost plus a constant times the standard deviation of the hedging cost.

The code for the Chiarella-Heston model and the associated hedging methodology are open-sourced¹.

4.2 Proposed Chiarella-Heston Model

We propose an innovative agent-based model called Chiarella-Heston model. The model is inspired by both the Extended Chiarella model in [MCB20] and the Heston stochastic volatility model in [Hes93]. It can also be considered a generalisation of both models. The model is proposed to address the shortcomings of the Extended Chiarella model and the Heston model in generating realistic financial market simulations. For example, the Extended Chiarella model is able to reproduce the "trend" and "value" effect in financial markets but is difficult to replicate the volatility clustering stylised fact. On the other hand, the Heston volatility model is capable of reproducing the volatility clustering stylised fact, but it assumes a constant drift of asset price which is not the case in real financial markets. The proposed model overcomes these shortcomings by extending and combining the two models. Specifically, we replace the noise traders in the Extended Chiarella model with volatility traders, whose demand level adapts according to the volatility signal level. From another perspective, we also replace the unrealistic constant drift in asset price in the Heston model with the demand from fundamental traders and momentum traders. In this way, a novel stochastic dynamic system is derived.

4.2.1 Model Setup

We denote the price of a stock at time t as P_t . The total signed volume traded on the market from t to $t + \Delta$ constitutes the cumulative demand imbalance in the same period. This quantity is denoted as $D(t, t + \Delta)$. This aggregated demand depends on the trading strategies of various types of market participants. Following [MCB20] and [Kyl85], the price dynamics are assumed to be governed by a

¹https://github.com/Ferrari1996gk/Deeper_Hedging

linear price impact mechanism:

$$P_{t+\Delta} - P_t = \lambda D(t, t + \Delta) \quad (4.1)$$

where λ is called "Kyle's lambda", which is related to the liquidity of the market and is a first-order approximation of market price sensitivity to market demand and supply. While a linear price impact mechanism does not reflect real-world market behaviour, the assumption is sufficient to generate realistic price time series that reproduce key stylised facts. The market participants are assumed to be heterogeneous in their trading decisions.

Following the Chiarella model [Chi92], we first populate our model with fundamental traders and momentum traders. Innovative volatility traders are then added to the model. Since traders of the same type exhibit the same behaviours, we only use one agent for each type of trader. This single agent represents the corresponding type of traders in the market and generates the aggregate demand and supply of that group of traders. With only three agents in the model, the simulation process is computationally efficient. Each trader is associated with some parameters that control the trading behaviours and the amount of demand and supply generated by the corresponding trader group. We will show the calibration of these parameters in later sections.

Fundamental Trader

Fundamental traders base their trading decisions on the perceived fundamental value of a stock, which is denoted as F_t . If a stock's price is less than its fundamental value ($F_t - P_t > 0$), these traders are inclined to purchase it, and conversely, they are likely to sell it if it is overpriced. As per the standard set in [Chi92], it is assumed that the aggregated demand from fundamental traders is proportional to the degree of mispricing. Hence, the total demand from these fundamental traders is represented as $\kappa(F_t - P_t)$, with κ controlling the overall demand generated by these traders. The fundamental value F_t is an exogenous signal fed into the model.

Momentum Trader

Momentum traders, often referred to as "Chartists," engage in buying and selling financial assets based on recent price trends. Their strategy aims to capitalise on upward or downward movements in stock prices until the trend starts to fade. Unlike investors who consider the fundamental value of a stock, momentum traders primarily concentrate on analyzing recent price action and movement. They take

long positions when the stock price has been on an upward trajectory and opt for short positions when the price shows a recent decline.

Numerous techniques exist for estimating the momentum of stock prices. One widely used approach is the exponentially weighted moving average of previous returns, where the decay rate is denoted as α . As stated in [MCB20], this trend signal is represented as M_t :

$$dM_t = -\alpha M_t dt + \alpha dP_t \quad (4.2)$$

where α is the decay rate. Given the trend signal M_t , the demand function of momentum traders is represented as $f(M_t)$. According to [MCB20], the demand function $f(M_t)$ must satisfy two conditions:

- $f(M_t)$ is increasing.
- $f''(M_t) * M_t < 0$

where the first condition aligns with the principles of momentum trading, while the second condition incorporates the risk-averse assumption applied to momentum traders. As described in [Chi92], we opt for the function $f(M_t) = \beta \tanh(\gamma M_t)$ with the constraint that $\gamma > 0$. In this context, γ reflects the level of saturation in momentum traders' demand when momentum signals are exceptionally high. This saturation phenomenon can be attributed, in part, to factors like budget constraints and risk aversion, which are commonly observed among real chartists. Meanwhile, β governs the overall demand generated by momentum traders and is assumed to be positive. This means that momentum traders exhibit positive demand when the momentum signal (M_t) is positive and negative demand when the momentum signal is negative. The selection of this demand function for momentum traders precisely fulfils the two specified requirements.

Volatility trader

We have incorporated a novel element into our agent-based model by introducing volatility traders. In addition to fundamental traders and momentum traders, a real financial market involves a diverse range of participants, each following distinct trading strategies. In our simplified model, volatility traders serve as a collective representation of all market participants influencing the dynamics of market volatility. Note that the trading activity of the volatility trader is different from the volatility trading

in the derivative trading literature. In terms of trading logic, the volatility trader adjusts trading behaviour according to a volatility signal that mimics the real volatility dynamics. Heston model [Hes93] is a popular stochastic volatility model used in finance to describe the dynamics of an asset's volatility over time. We choose the Heston model here because of nice features of the model such as the mean reversion of volatility and the correlation between an asset price and its volatility. Inspired by the Heston model, we introduce a volatility signal $\sqrt{\Sigma_t}$:

$$d\Sigma_t = \phi(\theta - \Sigma_t)dt + \sigma\sqrt{\Sigma_t}dW_t^V$$

where θ is the long-term mean of the variance of the asset returns, ϕ is the rate of mean reversion of the volatility process, and σ is the volatility of the volatility process. These parameters can be estimated from empirical data, or be estimated globally to fit the stylised facts. The specific calibration process is presented in later sections. W_t^V is a Wiener Process. $\sqrt{\Sigma_0}$ is the initial volatility level.

With the volatility signal $\sqrt{\Sigma_t}$ on hand, the demand function of the volatility trader is proportional to the volatility signal, with another Wiener Process included in the term:

$$D_{vol}(t, t + \Delta) = \omega\sqrt{\Sigma_t}dW_t^S$$

where ω is a hyper-parameter that controls the overall demand level for volatility traders. W_t^S is a Wiener Process that is correlated with W_t^V :

$$dW_t^S dW_t^V = \rho dt$$

where ρ is the correlation coefficient. As is in the Heston model, the correlation here aims to create a correlation between the asset price and its volatility, which is a key feature of many financial markets and is known as the "leverage effect". The leverage effect refers to the observed phenomenon that asset returns and volatility are often negatively correlated. In other words, when the price of an asset decreases, its volatility tends to increase, and vice versa. By including a correlation between the demand from volatility traders and the volatility signal, the proposed model contributes to a correlation between the asset price and its volatility, making it a more realistic model of financial markets. The correlation is typically negative, reflecting the negative correlation between asset returns and volatility observed in many markets.

Overall, the demand function for the volatility trader is:

$$\begin{aligned}
D_{vol}(t, t + \Delta) &= \omega \sqrt{\Sigma_t} dW_t^S \\
d\Sigma_t &= \phi(\theta - \Sigma_t)dt + \sigma \sqrt{\Sigma_t} dW_t^V \\
dW_t^S dW_t^V &= \rho dt
\end{aligned} \tag{4.3}$$

Model Dynamics

Taking into account the demand of all three types of traders, the total demand is then given by

$$\begin{aligned}
D(t, t + \Delta) &= \hat{\kappa} \int_t^{t+\Delta} (V_r - P_r) dr + \hat{\beta} \int_t^{t+\Delta} \tanh(\gamma M_r) dr \\
&\quad + \hat{\omega} \int_t^{t+\Delta} \sqrt{\Sigma_r} dW_r^S.
\end{aligned} \tag{4.4}$$

In addition, it is assumed that the fundamental value (F) is driven by a Geometric Brownian Motion with volatility σ_F and drift μ . We use g to represent the volatility-adjusted average growth ($\mu - \frac{1}{2}\sigma_F^2$) at log level. Combined with Equation 4.1, the price dynamics for $\Delta \rightarrow 0$ is represented by the following stochastic dynamic system:

$$\begin{aligned}
dP_t &= \kappa(F_t - P_t)dt + \beta \tanh(\gamma M_t)dt + \omega \sqrt{\Sigma_t} dW_t^S \\
dF_t &= gdt + \sigma_F dW_t^F \\
dM_t &= -\alpha M_t dt + \alpha dP_t \\
d\Sigma_t &= \phi(\theta - \Sigma_t)dt + \sigma \sqrt{\Sigma_t} dW_t^V \\
dW_t^S dW_t^V &= \rho dt
\end{aligned} \tag{4.5}$$

where κ, β, ω are equal to $\lambda\hat{\kappa}, \lambda\hat{\beta}, \lambda\hat{\omega}$, respectively. Note that P_t and F_t are **logarithms** of prices and fundamental values, respectively.

Discrete-time Simulation

Equation 4.5 is the continuous version of our proposed model. A discrete-time version of the model is required to simulate and estimate the parameters of the model. Here we present the discrete-time version of the proposed model, where one time step is equal to one day:

$$\begin{aligned}
 p_{t+1} - p_t &= \kappa(f_t - p_t) + \beta \tanh(\gamma m_t) + \omega \sqrt{\Sigma_t} \epsilon_{t+1}^S \\
 m_{t+1} &= (1 - \alpha)m_t + \alpha(p_{t+1} - p_t) \\
 f_{t+1} &= f_t + g + \eta_{t+1} \\
 \Sigma_{t+1} &= \Sigma_t + \phi(\theta - \Sigma_t) + \sigma \sqrt{\Sigma_t} \epsilon_{t+1}^V \\
 \text{Corr}(\epsilon_{t+1}^S, \epsilon_{t+1}^V) &= \rho
 \end{aligned} \tag{4.6}$$

where η_{t+1} is i.i.d. with normal distribution with zero mean and standard deviation σ_F . ϵ_{t+1}^S and ϵ_{t+1}^V both follow a normal distribution with zero mean and unit standard deviation. The correlation between ϵ_{t+1}^S and ϵ_{t+1}^V is ρ , which is generally negative. We remind again that p_t and f_t are logarithms of prices and fundamental values, respectively.

4.2.2 Relationship with Existing Models

Our proposed Chiarella-Heston model is a generalisation of the Extended Chiarella model [MCB20]. Observe that in Equation 4.5 for $\phi = 0$ and $\sigma = 0$ we recover the Extended Chiarella model in [MCB20], with $\sigma_N = \omega \sqrt{\Sigma_0}$. The noise traders in the Extended Chiarella model is described by Brownian Motion and have a constant parameter σ_N describing the size of total demand from noise traders. Instead, the proposed Chiarella-Heston model replaces the noise traders with volatility traders, whose demand level adapts according to the volatility level. Intuitively, the volatility traders create a larger demand (supply) level when the volatility signal is large, and vice versa. The volatility traders also contribute to reproducing the phenomenon that asset returns and volatility usually have a negative correlation. Both features increase the realism of the proposed model and enable an advantage over the Extended Chiarella model. In this sense, our proposed model can also be called an Extended Chiarella Model with Stochastic Volatility.

On the other hand, the proposed Chiarella-Heston model can also be considered a generalisation of the

Heston stochastic volatility model [Hes93]. Observe that in Equation 4.5 for $\kappa = 0$, $\alpha = 0$ and $\omega = 1$ we recover the Heston stochastic volatility model in [Hes93], with $\mu = \beta \tanh(\gamma M_0)$. The Heston model comprises one deterministic part and one non-deterministic part. The proposed Chiarella-Heston model converts the Heston model into an agent-based model by replacing the deterministic part. Specifically, the drift part in the Heston model is replaced by the aggregate demand from fundamental traders and momentum traders. The Heston model assumes a constant drift in asset price, while in reality, the asset price dynamic is not stationary. The proposed model drops the constant drift assumption. The interactions between fundamental traders and momentum traders contribute to the price dynamics in the model, making the model a more realistic replication of real financial markets.

4.2.3 Model Calibration

This section outlines the methodology for calibrating the proposed Chiarella-Heston model. Recall that in Chapter 3, we have emphasised that the calibration process is an indispensable part of developing an agent-based modelling framework. The calibration process involves identifying the best set of model parameters that allow the model to create the most accurate simulation of the financial market. In this section, we discuss the actual data used and the corresponding stylised facts prevalent in financial markets. The workflow for calibrating parameters is then presented.

Data and Stylised Facts

During the model calibration process, acquiring real financial market data is imperative to establish an accurate calibration target. To achieve this, we collected daily price data for the S&P 500 index spanning a total of 6000 trading days, encompassing the entire trading period from August 26th, 1999, to July 1st, 2023. The dataset is divided into two equal halves, each comprising 3000 trading days. The first 3000-day price data are employed for calibrating the model parameters, while the second 3000-day price data is reserved exclusively for testing purposes in the subsequent deep hedging experiments.

As presented in Chapter 3, financial price time series data display some interesting statistical characteristics that are commonly called stylised facts. According to [Sew11], stylised facts in financial markets refer to empirical findings that are so consistently present in asset returns across different markets and time periods that these findings are accepted as fundamental truths or "facts". A stylised fact is a simplified presentation of an empirical finding in financial markets. A successful and realistic financial market

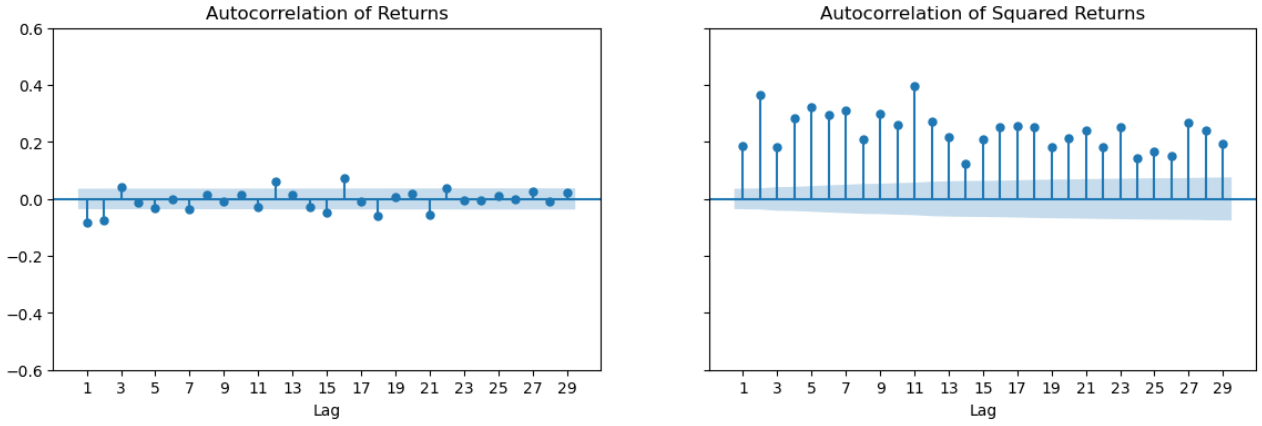


Figure 4.1 Historical stylised facts

simulation is capable of reproducing various stylised facts. As presented in Chapter 3, major stylised facts found in financial markets include fat-tailed distribution of returns, autocorrelation of returns, and volatility clustering. For a thorough review of financial stylised facts, refer to [Con01].

Calibration Workflow

Similar to the calibration process in the XGB-Chiarella method in Chapter 3, we mainly focus on three stylised facts in our calibration process: fat-tailed distribution of returns, autocorrelation of returns, and volatility clustering. The first stylised fact refers to the fact that price return distributions have been consistently observed to be fat-tailed across all timescales, indicating a higher likelihood of large price movements than what is predicted by standard normal distributions. Instead of the K-S statistic used in Chapter 3, here we use the Hill Estimator of the tail index [Hil10] as the primary metric for evaluating the fat-tail characteristic, with a lower value indicating a fatter tail in the return distribution. The K-S statistics is dropped because the Hill Estimator of the tail index is better at capturing global characteristics of a distribution. Details about the Hill Estimator of the tail index will be given in the next Chapter in Section 5.3.1. The second stylised fact refers to the fact that financial returns lack significant autocorrelation, except for small lags in high-frequency. Consequently, our daily simulation is required not to create significant autocorrelation patterns in returns. The third stylised fact, volatility clustering, refers to the phenomenon where periods of high volatility tend to be followed by periods of high volatility, and periods of low volatility tend to be followed by periods of low volatility. This stylised fact is reflected by the significantly positive autocorrelation of **squared** returns, slowly decaying with lags increasing. Figure 4.1 presents the historical autocorrelation of returns and historical autocorrelation of squared returns.

The calibration target is to make the proposed model reproduce realistic stylised facts. A loss function is needed so that minimising the loss function improves the "realism" of a simulated financial market. Consistent with [GVL⁺22] and [GVW⁺24], the loss function, called stylised facts distance, is constructed by measuring the discrepancy between the stylised facts from historical and simulated data. The stylised facts distance involves the above three stylised facts as well as the overall volatility level:

$$D(\boldsymbol{\vartheta}) = w_1 * \Delta_{Hill}(\boldsymbol{\vartheta}) + w_2 * \Delta_V(\boldsymbol{\vartheta}) + w_3 * \Delta_{ACF^1}(\boldsymbol{\vartheta}) + w_4 * \Delta_{ACF^2}(\boldsymbol{\vartheta}) \quad (4.7)$$

where $\boldsymbol{\vartheta}$ is the model parameter vector to be estimated, $\Delta_{Hill}(\boldsymbol{\vartheta})$ is the distance between simulated Hill estimator and historical Hill estimator, $\Delta_V(\boldsymbol{\vartheta})$ is the difference between simulated volatility and historical volatility, $\Delta_{ACF^1}(\boldsymbol{\vartheta})$ is the distance between simulated autocorrelation of returns and historical autocorrelation of returns, $\Delta_{ACF^2}(\boldsymbol{\vartheta})$ is the distance between simulated autocorrelation of squared returns and historical autocorrelation of squared returns. w_1, w_2, w_3 and w_4 are corresponding weights. For detailed calculations of all terms in Equation 4.7, refer to [GVW⁺24]. Finally, the calibration target is:

$$\hat{\boldsymbol{\vartheta}} = \arg \min_{\boldsymbol{\vartheta} \in \Theta} D(\boldsymbol{\vartheta}) \quad (4.8)$$

There are in total 11 model parameters to be estimated: $\kappa, \beta, \gamma, \omega, g, \sigma_F, \alpha, \phi, \theta, \sigma$ and ρ . Theoretically, we could estimate all 11 parameters by minimising the stylised facts distance. However, some of the parameters have economic representations and we could estimate these parameters directly from historical data. In this way, we also reduce computational costs during calibration. Following [MCB20], we fix parameter α . We assume the typical horizon of trend computation for a low-frequency trend-following strategy is one week (five days), which gives $\alpha = 1/(1+5) = 1/6$. We assume the fundamental value to be a Geometric Brownian Motion with drift (μ) equals average historical return and standard deviation (σ_F) equals the average volatility of historical data. Recall that $g = \mu - \frac{1}{2}\sigma_F^2$. σ takes the value of the volatility of the historical volatility, with window size 30 in our calculation. ρ takes the value of the historical correlation between asset price returns and volatility. According to [MCB20], we also fix γ and let γ take the value of 10. The remaining model parameters, $\kappa, \beta, \omega, \theta$ and ϕ are calibrated via grid search by minimising $D(\boldsymbol{\vartheta})$. Figure 4.2 shows the simulated stylised facts after calibration, which resemble the historical stylised facts.

We also calibrated the Geometric Brownian Motion, the Heston model, and the Extended Chiarella model to match the stylised facts. The minimised stylised facts distances for all four models are

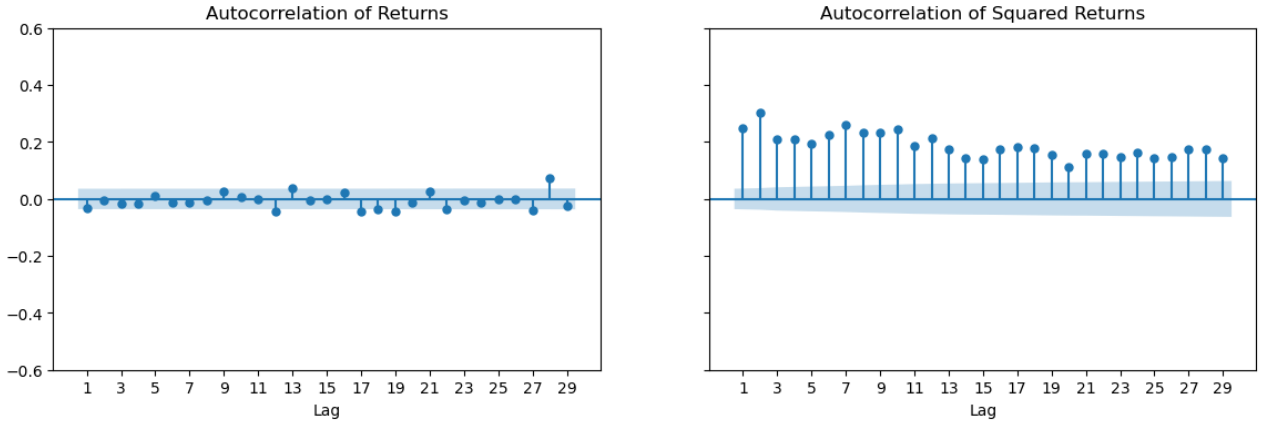


Figure 4.2 Chiarella-Heston model simulated stylised facts after calibration

Model	GBM	Heston	Extended Chiarella	Chiarella-Heston
Δ_V	0.002	0.229	0.003	0.028
Δ_{Hill}	0.172	0.116	0.177	0.048
Δ_{ACF^1}	0.067	0.064	0.069	0.066
Δ_{ACF^2}	0.273	0.145	0.271	0.082
Total Stylised Facts Distance	0.514	0.554	0.520	0.224

Table 4.1 Comparison of stylised facts distance among different models after calibration

shown in Table 4.1. It is illustrated that the proposed Chiarella-Heston model achieves minimum stylised facts distance among the four models. Specifically, the Geometric Brownian Motion and the Extended Chiarella model are good at reproducing realistic volatility level, but are not able to capture the volatility clustering stylised fact (reflected by large Δ_{ACF^2}). The Heston model could in a way replicate the volatility clustering stylised fact but is not good at reproducing realistic volatility level. In contrast, the proposed Chiarella-Heston model can reproduce realistic volatility level as well as the volatility clustering stylised fact. In addition, the proposed model has another advantage over the other three models in that it replicates the fat-tails of returns in the financial market, shown by the much smaller Δ_{Hill} . Overall, the Chiarella-Heston model reduces the stylised facts distance by more than 50%, indicating that it provides a more realistic representation compared to other models.

4.2.4 Model Validation

To confirm the advantage of the Chiarella-Heston model over the baseline model, validation of the proposed model is carried out. Agent-based model validation in finance is the process of ensuring that an agent-based model accurately represents the real-world financial system. This involves comparing

mean GSL-div (Chiarella-Heston)	mean GSL-div (GBM)	t-statistic	p-value
0.183	0.199	4.968	2.8e-6

Table 4.2 Statistical test between two sets of GSL-div values

the outputs of the model under various scenarios to actual financial data. The main metric we use here is the Generalised Subtracted L-divergence (GSL-div), which is proposed in [Lam18].

We compare the GSL-div between the Chiarella-Heston model-generated time series and historical time series with the GSL-div between the GBM-generated time series and historical time series. Specifically, the calibrated Chiarella-Heston model is utilised to generate M price scenarios. For each price scenario, a GSL-div value is calculated using the generated time series and historical time series. The same process is carried out for the Geometric Brownian Motion model. In this way, we obtain two sets of GSL-div values. A statistical test is carried out to compare the two sets. Results are shown in Table 4.2. The p-value is smaller than 0.05, showing that the GSL-div between the Chiarella-Heston model-generated time series and historical time series is significantly smaller than the GSL-div between the GBM-generated time series and historical time series. This result backs the conclusion that the Chiarella-Heston model is able to generate more realistic simulated financial data.

4.3 Deep Hedging Experiments

Deep hedging is a technique utilising deep reinforcement learning to derive optimal hedging strategies for derivatives when there are transaction costs. Most existing methods apply Geometric Brownian Motion or the Heston model to generate simulated data to train and test deep hedging agents [CCHP21]. As an application of the proposed Chiarella-Heston model, we use the Chiarella-Heston model to generate simulated data for training a deep hedging agent. Instead of testing with simulated data, we test the performance of the deep hedging agent with real historical data. This same process is also carried out with the baseline model, where training data are generated by a calibrated Geometric Brownian Motion. We show that the deep hedging agent trained by artificial data generated by the Chiarella-Heston model achieves better performance than the agent trained by data based on Geometric Brownian Motion.

4.3.1 Problem Setting

We consider a scenario where a trader is hedging a short position in a call option, where one option contract represents 100 underlying shares of the underlying asset. The trader hedges her option position solely by buying or selling the underlying asset. We make the assumption that the trader adjusts position at time intervals of 1 day and incurs trading costs. The option's lifespan is N days, which is set to 30 in our experiments. In our model, the cost of a transaction in the underlying asset is proportional to the value of the asset being traded, although the model can be readily modified to suit different assumptions. Following [CCHP21], the state at time t comprises three values:

- Current position of the underlying asset, which is determined by action from the previous time period ($t - 1$ to t)
- Underlying asset price at time t
- Time to maturity at time t

At time $t = 0$ the initial underlying asset position is 0. The action at time t is to determine the amount of the underlying asset to be held for the next period; i.e., from time t to time $t + 1$. The action space in our experiments is a continuous interval between 0 and 100. As for reward calculation, we opt for the accounting P&L formulation, where the reward for each time step is the profit and loss of the combined portfolio including both the option position and the underlying asset position:

$$R_{t+1} = -(V_{t+1} - V_t) + H_t(S_{t+1} - S_t) - \pi|S_{t+1}(H_{t+1} - H_t)| \quad (4.9)$$

for $0 \leq t \leq N$, where V_t is the price of the option at time t , S_t is the underlying asset price at time t , H_t is the holding between time t and $t + 1$, π represents the trading cost as a proportion of the transaction value. The -1 factor before $(V_{t+1} - V_t)$ in Equation 4.9 is because of the negative option position. The target of the problem is to train a deep reinforcement learning agent to drive an optimal trading strategy carried out in the span of N days, which ideally minimises the accumulated hedging cost.

Training and Testing Data

We mentioned in Section 4.2.3 that we collected daily price data for the S&P 500 index spanning a total of 6000 days. The dataset is divided into two equal halves with the first half used for calibrating the

proposed Chiarella-Heston model. The training data in our deep hedging experiments are generated by the calibrated Chiarella-Heston model. The calibrated Chiarella-Heston model is simulated according to Equations 4.6 for M times ($M = 50000$ in our experiments), generating M scenarios. Each scenario comprises a trajectory of the underlying asset price, with N time steps. Option prices are calculated by the Black-Scholes model. The simulated training data constitute the training environment where the deep hedging agent learns to minimise the accumulated hedging cost.

The testing process utilises the second half of the collected S&P 500 index daily price data to avoid any form of overlap between training data and testing data. The testing data consist of 3000 daily price observations. Using a rolling window with length $N = 30$, we obtain 2970 price trajectories. Calculating the returns of each price trajectory and providing an initial price, we obtain 2970 new price scenarios for testing the performance of the trained deep hedging agent. In this way, even though we change the scale of the price level, the testing price scenarios still have exactly the same returns as the historical price returns. The option price is again calculated by the Black-Scholes model. The justification for using the Black-Scholes model is that in reality, the prices of vanilla options are very close to the theoretical Black-Scholes price.

4.3.2 Reinforcement Learning Algorithm

The deep hedging agent in our experiments uses a revised form of the Deep Deterministic Policy Gradient (DDPG) algorithm [LHP⁺15] for learning the optimal hedging strategy. The algorithm here follows the set-up in [CCHP21] and works with hedging costs (negative rewards). The state space and action space are presented in Section 4.3.1. The objective function to minimise is the expected hedging cost plus a constant multiplied by the standard deviation of the hedging cost. As in [CCHP21], two Q-functions are introduced to implement the objective function. One Q-function estimates the expected cost for state-action combinations while the other one estimates the expected value of the squared cost for state-action combinations. Since the detailed reinforcement learning algorithm is not the focus of this chapter, please refer to [LHP⁺15] and [CCHP21] for detailed learning and updating rules of the DDPG algorithm.

The actor and critic neural network architectures in our experiments, though, are different from [CCHP21]. The actor and critic networks all have three hidden layers, with sizes 32, 64, and 32 neurons. The ReLU activation function is applied to each hidden layer. Batch normalisation layers are also

included between the input layer and the first hidden layer, as well as between adjacent hidden layers. For the actor-network, the output layer is followed by a Sigmoid function, mapping output to position levels. The training is carried out on two NVIDIA GeForce RTX 3090 cards. Training for 15000 epochs costs approximately 5 hours.

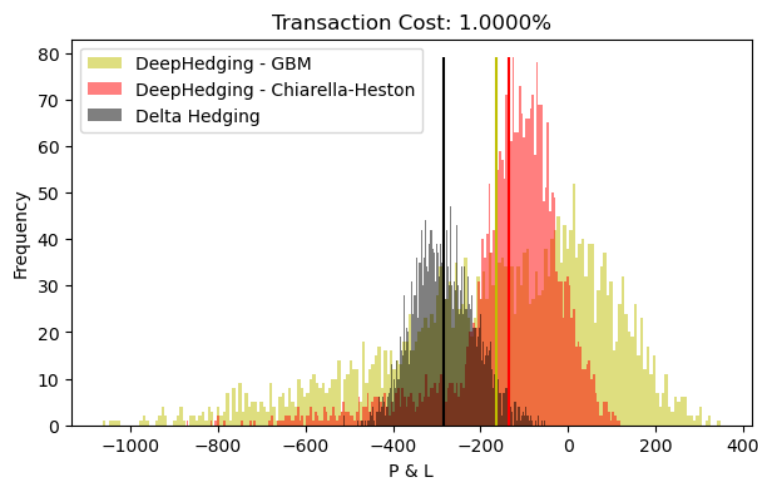
4.3.3 Deep Hedging Experimental Results

We train the deep hedging agent with the training dataset and test the performance of the learned hedging strategy on the testing dataset. The experiment is carried out under various transaction cost levels. In addition, we also obtain two baseline strategies for comparison. The first baseline is the traditional delta hedging strategy, which always keeps delta neutral at the end of the day. The second baseline is the hedging strategy learned by the same deep hedging agent, but the training process is carried out on GBM-simulated data. The chosen performance metrics are the P&L and the expected shortfall. Table 4.3 and Table 4.4 present the hedging performance (P&L and expected shortfall) comparison between the deep hedging strategy trained on Chiarella-Heston model generated data and the two baselines - GBM-driven deep hedging strategy and traditional delta hedging strategy. The results show that the Chiarella-Heston model-driven deep hedging strategy clearly outperforms the other two strategies. In terms of P&L, the two deep hedging strategies outperform the traditional delta hedging strategy in all transaction cost levels except 0.1%. This is consistent with the literature that deep hedging generally performs better than delta hedging when transaction costs exist. According to Table 4.3, the two deep hedging strategies perform similarly in terms of P&L. However in Table 4.4, the strategy trained by Chiarella-Heston model consistently performs better than the strategy trained by Geometric Brownian Motion data, achieving smaller expected shortfall. In terms of risk management, the Chiarella-Heston model reduces the average expected shortfall from -3.93% to -2.74%, representing a 30% improvement. These results demonstrate that the Chiarella-Heston model effectively captures risky patterns observed in real financial markets, which is learned by the deep hedging agent to minimise losses in adverse scenarios. This highlights a successful and practical application of the proposed Chiarella-Heston model. Figure 4.3 shows a histogram of the P&L distribution of the three strategies. It is shown clearly that the deep hedging strategy trained on data generated by the Chiarella-Heston model significantly outperforms the other two strategies.

Transaction Cost	0.01%	0.1%	0.2%	0.4%	0.6%	1.0%
Deep Hedging Chiarella-Heston	54.45	-43.68	-22.62	-80.64	-185.26	-136.37
Deep Hedging GBM	39.23	-69.19	39.36	-90.29	-99.99	-165.33
Delta Hedging	8.62	-17.93	-47.44	-106.45	-165.46	-283.48

Table 4.3 Hedging performance - P&L (\$)

Transaction Cost	0.01%	0.1%	0.2%	0.4%	0.6%	1.0%
Deep Hedging Chiarella-Heston	-0.83%	-3.33%	-2.32%	-2.94%	-3.63%	-3.36%
Deep Hedging GBM	-2.08%	-4.83%	-2.47%	-3.89%	-4.81%	-5.52%

Table 4.4 Hedging performance - Expected Shortfall (%)**Figure 4.3** P&L distribution of three hedging strategies

4.4 Summary

The quantitative finance industry has seen a surge in data science applications, utilising powerful algorithms with extensive data and computing capabilities. Deep learning algorithms are now being explored as replacements for traditional Greek-based methods in hedging derivatives, taking into account factors like transaction costs. Deep hedging, a data-driven approach that minimises risk using neural networks, shows promise in optimising hedge strategies. However, deep hedging demands substantial training data, often generated synthetically from stochastic models, which may not fully replicate real market dynamics. This research addresses this challenge by proposing an innovative Chiarella-Heston model, combining momentum, fundamental, and volatility traders. Calibrated to replicate empirical stylised facts, this model produces more realistic financial time series data, outperforming baseline

mathematical models. The key contributions of this work are the Chiarella-Heston model and the enhanced performance of deep hedging agents in practical environments. The model's ability to generate accurate financial data and the trained agents' effectiveness in real trading situations highlight the potential of this methodology. The proposed model is validated using the Generalised Subtracted L-divergence metric. It generates a training dataset for a deep hedging agent, trained with the DDPG algorithm, to optimise hedging strategies under different transaction cost levels. The reinforcement learning agent, trained with data from the proposed model, outperforms the baseline in most cost levels and performs well in real trading using empirical data. By applying an agent-based approach to enhance deep hedging, this research provides a valuable strategy for improving risk management and hedging operations in the finance industry.

The model proposed in this chapter simulates price time series on a daily timescale, while the model in the previous Chapter operates on a minute-level timescale. However, with advancements in computing technology, modern financial markets now operate at sub-second levels, with high-frequency trading occurring in microseconds or even nanoseconds. In the following chapter, we will explore the domain of high-frequency trading. Building on the model from Chapter 3, we will propose an extended model that operates at sub-second intervals, incorporating comprehensive exchange protocols such as limit order books for transaction matching. The high-frequency market simulator will be used to simulate risk scenarios and analyse high-frequency market dynamics.

Chapter 5

High-frequency financial market simulation and flash crash analysis

5.1 Introduction

In this chapter, we delve into the high-frequency aspect of financial markets. A novel high-frequency agent-based financial market simulator is developed to replicate realistic market conditions. The model incorporates full exchange protocols, including limit order books, and is calibrated using the same machine learning surrogate modelling approach proposed in Chapter 3. It successfully reproduces stylised facts in financial markets, as confirmed by statistical tests. The proposed simulator is employed to simulate risk scenarios such as flash crash events. Detailed analysis of high-frequency market dynamics during those risk scenarios are also presented in this chapter.

5.1.1 Motivation

With the advent of electronic financial markets for the exchange of securities, the electronic centralised limit order book has become the standard market mechanism for transaction matching and price discovery. This form of order book offers market participants a more liquid market system with a small bid-ask price spread, increased market depth and decreased transaction times.

Algorithmic trading is commonly defined as the use of computer algorithms to automatically make trading decisions, submit orders, and carry out post-submission order management. In the past decade,

algorithmic trading has grown rapidly across the world and has become the dominant way securities are traded in financial markets. Constantly improving computer technology and its application by both traders and exchanges, together with the evolution of market micro-structure, automation of price quotation and trade execution have together enabled faster trading. Nowadays the speed of order submission has become a principal characteristic for distinguishing trading agents. Market participants known as high-frequency traders are capable of trading hundreds of times in a second, using fast algorithms and specialised network connections with exchanges. High-frequency traders are often orders of magnitude faster in order submission than other traders and even other trading algorithms.

The rise of algorithmic trading and high-frequency trading has had broad impacts on financial markets, especially on the price discovery process and market price stability. One conspicuous impact is the increasingly frequent "flash crash" in major financial markets. The flash crashes comprise large and rapid changes in the price of an asset that do not coincide with changes in economic fundamental value for the asset. The flash crash events have occurred in markets that are among the largest and most liquid exchanges in the world. One representative flash crash event is the famous 2010 Flash Crash, which happened in the U.S. stock market on May 6th, 2010. During this flash crash event, one market participant's algorithm caused a sharp price drop in the E-mini S&P futures market. The flash crash soon spread to other futures markets and equity markets. The market price fell almost 6% in just several minutes, while the bulk of losses was recovered nearly as quickly. The 2010 Flash Crash led to turmoil market conditions and caused huge market value loss. As for the cause of the 2010 Flash Crash, [KKST17] show that the key events in the 2010 Flash Crash have clear relationships with regard to algorithmic trading.

The 2010 Flash Crash seems to be singular because of the fact that no following events have rivalled its depth, breadth, and speed of price movement. Nevertheless, flash crashes on a smaller scale happen frequently. These events are termed mini flash crashes [JZH⁺12]. According to [JZH⁺12], there are more than 18,000 mini flash crashes that are identified in the U.S. equity market between 2006 and 2011. As scaled-down versions of the 2010 Flash Crash, mini flash crashes are abrupt and severe price changes that occur in an extremely short period of time [GKP12]. Mini flash crashes are attracting great research interest because their frequent occurrence could destabilise the financial market and undermine investor confidence [GKP12].

The flash crash episodes¹, including large flash crashes and mini flash crash events, are of significant

¹We use "flash crashes", "flash crash episodes" and "flash crash scenarios" interchangeably in this thesis.

concern to researchers, practitioners, and policymakers. Financial markets in which price changes are orderly and reflect proper changes in valuation factors are desirable. However, flash crash episodes could potentially disorganise such a desirable market and cause adverse consequences for financial stability if they were to impede investment by undermining investor confidence in the price at which securities could be transacted [KNWB18]. To prevent flash crash episodes from becoming more frequent and longer-lasting, it is important to understand how such episodes arise. In this chapter, we explore the dynamics during both large flash crashes and mini flash crashes. The major objectives are understanding what happens during these events and identifying conditions that would affect the characteristics of these flash crash scenarios. The main methodology employed here is financial market simulation with agent-based models. The agent-based financial market simulation provides realistic synthetic financial market data and a testbed for exploring dynamics during flash crash episodes and conditions that influence the characteristics of flash crash episodes.

Various agent-based simulators have been developed in the literature. However, there are still gaps in creating ideal agent-based financial market simulators that are capable of generating synthetic high-frequency market data that are realistic. Specifically, most existing agent-based financial markets are of lower frequency such as daily or minutely. To explore market dynamics that involve high-frequency trading, a higher simulation frequency at sub-second level is needed. In addition, instead of using limit order books, many simulators make assumptions about the price formation process and use mathematical formulae to approximate the matching engine. This significantly undermines the realism of the simulator. Last but not least, the proper calibration and validation of agent-based financial market simulation are still an open problem.

To sum up, there are two **challenges** that this chapter aims to address:

- C3.1: How to develop and refine a high-frequency agent-based financial market simulator that fully integrates exchange protocols and includes a rigorous calibration and validation process to realistically replicate financial market dynamics.
- C3.2: Under the proposed agent-based financial market simulator, investigate the market dynamics during flash crashes (including both large flash crashes and mini flash crashes) and identify the conditions that influence the characteristics of flash crashes.

Trading Day	May 3rd, 2010	May 4th, 2010	May 5th, 2010	May 6th, 2010
Average gap between transactions (in milliseconds)	242.9	122.8	125.7	69.5

Table 5.1 Average gap between transactions for ESM0 from May 3rd, 2010 to May 6th, 2010².

5.1.2 Our Contributions

Motivated by the above challenges, we developed a novel high-frequency financial market simulator to bridge the existing gaps. The simulator is then employed to explore the dynamics during flash crashes and the conditions that affect flash crashes. Broadly speaking, our **contributions** in this chapter are threefold:

- It is known that sub-second level simulation is needed to replicate high-frequency dynamics in financial markets, which is rarely implemented in previous work. In this work, a high-frequency agent-based financial market simulator that comprises various types of traders is implemented, with each simulation step corresponding to 100 milliseconds. The 100 milliseconds step interval is chosen because the average interval between two transaction messages from the exchange is around 100 milliseconds, as shown in Table 5.1. A shorter step interval is not chosen because of the huge increase in computational expenses, though higher frequency is realisable in our simulation. In addition, our experimental results show that a step interval of 100 milliseconds is sufficient to capture high-frequency dynamics in financial markets, as demonstrated in Section 5.3. Full exchange protocols (limit order books) are implemented to simulate the order matching process, which acts exactly the same as the match engine in real-world exchanges. In this way, we provide a microstructure model of a single security traded on a central limit order book in which market participants follow fixed behavioural rules. The model is calibrated using the machine learning surrogate modelling approach. As for model validation, statistical test and moment coverage ratio results show that the simulation is capable of reproducing realistic stylised facts in financial markets.
- Under the framework of the proposed high-frequency agent-based financial market simulator, the 2010 Flash Crash is realistically simulated by introducing an institutional trader that mimics the real-world Sell Algorithm³ on May 6th, 2010, which is generally believed to have precipitated

²The results represent the average gaps between transactions throughout the entire trading day. These gaps tend to be smaller during busier trading hours and larger during quieter periods.

³See [SC10] for detailed description.

the flash crash event. We investigate the market dynamics during the simulated flash crash and show that the simulated dynamics are consistent with what happened in historical flash crash scenarios. We then explore the conditions that could have influenced the characteristics of the 2010 Flash Crash. According to our Monte Carlo simulation, three conditions significantly affect the amplitude of the 2010 Flash Crash: the percentage of volume (POV) of the Sell Algorithm, the market maker inventory limit, and the trading frequency of fundamental traders. In particular, we found that the relationship between the amplitude of the simulated 2010 Flash Crash and the POV of the Sell Algorithm is not monotonous, and so is the relationship between the amplitude and the market maker inventory limit. For the trading frequency of fundamental traders, the higher the frequency, the smaller the amplitude of the simulated 2010 Flash Crash.

- Similar analysis is carried out for mini flash crash events. An innovative type of trader called "Spiking Trader" is introduced to the agent-based financial market simulator, mimicking real-world price shocks to precipitate more mini flash crash events. Market dynamics for a typical simulated mini flash crash event are analysed. We also explore the conditions that could influence the characteristics of mini flash crash events. Experimental results show that the market maker inventory limit significantly affects both the frequency and amplitude of mini flash crash events. However, the trading frequency of fundamental traders shows no obvious influence on mini flash crash events in our experiments. A comparison between large and mini flash crash events is presented in Section 5.5.4, specifying the difference between these two types of events.

The **novelty** of our approach lies in several aspects. First, the proposed agent-based financial market simulator has a higher frequency than most other simulators in the literature. Our simulation step is at milliseconds level, which allows for the investigation of high-frequency dynamics in the simulated financial market, while most simulation models in literature adopt larger simulation steps of 1 second or 1 minute. Second, we explore the influence of different market configurations on the amplitude of the 2010 Flash Crash. To the best of our knowledge, there are few similar experiments in the existing literature. Third, an innovative type of trader named "Spiking Trader" is proposed to precipitate more mini flash crash events. Furthermore, the experiments that explore the conditions that influence the frequency and amplitude of mini flash crash events are also a novelty of this work.

The remainder of this chapter is organised as follows. Section 5.2 shows the structure and details for the proposed agent-based model, while Section 5.3 presents the model calibration process and model validation results. Section 5.4 and Section 5.5 provide simulation and analysis for the 2010 Flash Crash

scenario and mini flash crash scenarios, respectively, in the framework of the proposed agent-based financial market simulation.

5.2 Model Structure

This section presents the setup and components of the proposed agent-based high-frequency financial market simulator.

5.2.1 Model Set-up

The proposed model consists of various traders who submit limit orders or market orders to a central limit order book, which functions according to a continuous double auction mechanism. All the orders submitted by these traders in one simulation period, including limit orders and market orders, constitute the aggregated virtual "demand" and "supply" in that period. The aggregated virtual "demand" and "supply" depend on the trading strategies of various types of market participants. The market participants are assumed to be heterogeneous in their trading decisions. According to the Extended Chiarella model [MCB20], a popular set of traders in an artificial financial market includes momentum traders, fundamental traders, and noise traders. This composition of traders is capable of capturing the trend and value effects in financial markets. We further extend the model by dividing momentum traders into two groups: short-term momentum traders and long-term momentum traders. In this way, heterogeneity is introduced not only between groups of traders, but also inside a specific group of traders. In addition, in an artificial financial market with full exchange protocols, market maker is an indispensable component to create realistic limit order book behaviours. To sum up, five types of traders are included in our model: fundamental traders, short-term momentum traders, long-term momentum traders, noise traders, and market makers. Each type of trader is associated with several parameters which control the trading behaviours. Trading heuristics, specific parameters and the parameter calibration process will be presented in subsequent sections.

5.2.2 Price Formation Process

As mentioned above, we use a continuous double auction mechanism to form the price in our model, which has been adopted by most popular electronic exchanges around the world. The key element in

continuous double auction is the limit order book, which is a record of all outstanding limit orders maintained by the exchange. The price dynamics in our proposed model are exactly the same as the price dynamics in a real-world stock exchange - a limit order book is built to accept all limit orders and market orders. A trade occurs when a market order is entered or a limit order whose price crosses the spread of best bid and best ask. This marketable order transacts against the best opposing orders using a price-time priority, removing them from the order book until this order is either fully or partially filled, leaving the rest of the quantity of the order to become the new best bid or offer. All normal limit orders will join the limit order book according to price-time priority. For a complete description of how continuous double auction works, refer to [SFGK03].

We denote the price of a stock at time t as P_t . The simulated price in this work refers to mid-price, which is the mean value of the best bid price and the best ask price from the limit order book at time t :

$$P_t := 0.5 * b_t^0 + 0.5 * a_t^0$$

where b_t^0 and a_t^0 denote best bid and best ask price at time t , respectively.

5.2.3 Common Trader Behaviours

Traders in our simulation model have some behaviours in common, for example submitting and cancelling orders. We can assume that there is a certain type of agent called "Base Agent", and all traders in our model inherit the functionalities of this base agent. Specifically, all traders in the model have common behaviours as follows.

- Each trader has a parameter θ , which controls the probability of submitting a limit order. The value of θ depends on the type of traders and also varies during simulation, but the behaviour given a θ value is common for all types of traders. That is, for each simulation step, if $p \in U(0, 1) < \theta$, the trader will place a limit order, otherwise, no action is taken⁴. Note that the specific side of the order (buy or sell) depends on the trader type and market conditions. The value of θ could be zero for some traders, which means that the corresponding traders do not submit limit orders.
- Each trader has a parameter μ . The function of μ is identical to parameter θ , except that μ controls the probability of sending a market order. Similar to θ , the value of μ is different

⁴The expression $p \in U(0, 1)$ denotes that p is a random variable uniformly distributed between 0 and 1.

for different types of traders and varies for different timestamps. For each simulation step, if $p \in U(0, 1) < \mu$, the trader will place a market order, otherwise no action is taken. The specific side of the market order also depends on the trader type and market conditions. Similar to θ , the value of μ can also be zero for some particular types of traders.

- The side for all limit orders and market orders is determined by corresponding trader types and market conditions. A market order is submitted directly to the exchange after the side is determined. For a limit order, the price information is required. Given the order side, a buy limit order will have a price lower than the market mid-price at the corresponding timestamp, while a sell limit order will have a price higher than the market mid-price. The distance between the limit order price and the market mid-price follows a particular distribution, whose type is determined by the trader types. Specifically, the price distance for limit orders from market makers is sampled from a uniform distribution with parameters 0 and p_{edge}^{MM} , and the price distance for limit orders from other types of traders is sampled from a common log-normal distribution with parameters μ_ℓ and Σ_ℓ . The parameters of these distributions are calibrated to historical price time series stylised facts. The price for a limit order is calculated after the price distance is sampled from the corresponding distribution.
- Each type of trader has a parameter δ . Regardless of the order price and side, each limit order has a cancellation probability of δ at each simulation step. The value of δ is dependent on the type of trader who places the order, and is calibrated using historical market data to create realistic limit order book behaviours. In the presented model, all traders share the same value for δ except market makers, who have a higher value of δ . This reflects the fact that orders submitted by market makers tend to have a much higher cancellation/replacement rate.
- In the proposed model, short-term momentum traders, long-term momentum traders and noise traders all have a parameter called ρ , which controls the ratio between the number of market orders and limit orders placed by the same trader. That is, for each trader of these three types, there is a fixed relationship between θ and μ : $\mu = \theta * \rho$. This relationship enables a realistic ratio between the number of market orders and limit orders in the simulated financial market and the value of ρ is selected according to historical orders data.
- The volume V for each order is 100, regardless of limit order or market order.

Despite the common trader behaviours, each type of trader follows different trading heuristics and has

different values for the associated parameters. For example, fundamental traders only submit market orders. Market makers submit limit orders in normal trading time. Only after the inventory limit is hit will market makers submit market orders to reduce their inventory risk. The remaining types of traders submit both limit orders and market orders during trading hours, aiming to maintain a fixed ratio between the number of limit orders and the number of market orders. The remainder of the section describes the trading behaviour heuristics for each type of trader in more detail. Descriptions for all the parameters involved in the proposed model are summarised in Appendix A.

5.2.4 Fundamental Trader (FT)

Fundamental traders make their trading decisions based on the perceived fundamental value of the stock. The fundamental value is denoted as V_t . A fundamentalist tends to buy a stock if the stock is under-priced ($V_t - P_t > 0$), otherwise he will sell the stock. Following the convention in [MCB20], in this work we assume the aggregated demand of fundamental traders is polynomial to the level of mispricing. Specifically, the aggregated demand of fundamentalists is:

$$D_{FT} = \kappa_1(V_t - P_t) + \kappa_2(V_t - P_t)^3 \quad (5.1)$$

where κ_1 and κ_2 controls the overall demand generated by fundamental traders. Since we have N_{FT} fundamental traders, each fundamental trader will contribute $\frac{\kappa_1|V_t - P_t| + \kappa_2|V_t - P_t|^3}{N_{FT}}$ to the aggregated demand. In our setup, fundamental traders exclusively use market orders. This choice stems from their strategy of capitalising on significant disparities between market price and fundamental value. As predominantly low-frequency traders, they show little concern for the marginal price distinctions between limit and market orders. Their primary focus lies in promptly executing transactions when identifying distortions between market price and fundamental value. Since the fundamental traders only submit market orders, the value of θ for each fundamental trader is set to 0, while the value of μ for each fundamental trader is set to $\frac{\kappa_1|V_t - P_t| + \kappa_2|V_t - P_t|^3}{N_{FT}}$. The fundamental value V_t is an exogenous signal that is input to the model. Each fundamental trader is also associated with a parameter $S_{interval}^{FT}$. A fundamental trader only attempts to submit a market order once in every $S_{interval}^{FT}$ simulation steps. A detailed description of the fundamental trader's logic is given in Algorithm 1.

Algorithm 1 Fundamental Trader Logic

```

1: for each simulation step do
2:    $\theta \leftarrow 0$ 
3:    $\mu \leftarrow \frac{\kappa_1|V_t - P_t| + \kappa_2|V_t - P_t|^3}{N_{FT}}$ 
4:   if ( $simulation\_steps \bmod S_{interval}^{FT} == 0$ ) and ( $p \in U(0, 1) < \mu$ ) then
5:     if  $V_t - P_t > 0$  then Place a buy market order
6:     else if  $V_t - P_t < 0$  then Place a sell market order
7:     end if
8:   else No market order submitted
9:   end if
10: end for

```

5.2.5 Momentum Trader (MT)

Momentum traders are also called "Chartists". This group of traders buy and sell financial assets after being influenced by recent price trends. The assumption is to take advantage of an upward or downward trend in the stock prices until the trend starts to fade. Instead of looking at the fundamental value of the stock, momentum traders focus more on recent price action and price movement. If the stock price has been recently rising, a long position is established; otherwise, momentum traders will enter a short position if the price has been decreasing. In the proposed model, momentum traders can submit both limit orders and market orders. The ratio between the number of limit orders and market orders is fixed, which is denoted as ρ . The price information for a limit order is determined as follows. A buy limit order will have a price lower than the market mid-price at the corresponding timestamp, while a sell limit order will have a price higher than the market mid-price. The distance between the limit order price and the market mid-price follows a log-normal distribution with parameters μ_ℓ and Σ_ℓ . The parameters of the log-normal distribution are calibrated to historical price time series stylised facts. Price information for the limit order is subsequently calculated after the price distance is sampled from the corresponding distribution.

There are lots of methods to estimate the momentum of stock prices. A common trend signal is the exponentially weighted moving average of past returns with decay rate α . This trend signal is denoted by M_t :

$$M_t = (1 - \alpha)M_{t-1} + \alpha(p_t - p_{t-1}) \quad (5.2)$$

where α is the decay rate. Given the trend signal M_t , the demand function of momentum traders is denoted as $f(M_t)$. The demand function $f(M_t)$ must satisfy two conditions:

- $f(M_t)$ is increasing.

- $f''(M_t) * M_t < 0$

where the first condition is consistent with the nature of momentum trading and the second condition imposes the risk-averse assumption to momentum traders. Consistent with [MCB20], here we choose $f(M_t) = \beta \tanh(\gamma M_t)$ with the requirement that $\gamma > 0$. γ represents the saturation of momentum traders' demand when momentum signals are very large. This phenomenon is partly due to for example budget constraints and risk aversion, which are prevalent in real momentum traders. β controls the overall demand generated by momentum traders. β is also assumed to be positive, i.e. the demand of momentum traders is positive when the momentum signal (M_t) is positive, otherwise the demand is negative. The choice of this demand function for momentum traders strictly satisfies the two requirements.

According to the value of α , we further divide the group of momentum traders into two sub-groups: long-term momentum traders (small α) and short-term momentum traders (large α). The two types of momentum traders also have different values for β ; however, they share the same γ value. The proposed model includes N_{LMT} long-term momentum traders and N_{SMT} short-term momentum traders.

Long-term Momentum Trader (LMT)

Long-term momentum traders are associated with a small value for α . According to equation 5.2, a small α corresponds to slow changes in the momentum signal. Consequently, the momentum signal is smooth and reflects the trend on a longer time scale. In our intra-day simulation, we choose an α value of 0.01 for long-term momentum traders. This would approximately correspond to an intra-day hourly trend. The group of long-term momentum traders will focus on the relatively long-time price trend and calculate their demand function. For limit orders, the virtual aggregated demand from long-term momentum traders is $\beta_L \tanh(\gamma M_t^L)$, while for market orders the virtual aggregated demand from long-term momentum traders is $\rho \beta_L \tanh(\gamma M_t^L)$. As there are N_{LMT} long-term momentum traders, the θ and μ for each trader are corresponding quantities divided by N_{LMT} . Trading decisions are made according to the common order submission rules. A full description of long-term momentum traders' logic is shown in Algorithm 2.

Algorithm 2 Long-term and Short-term Momentum Trader Logic*

```

1: for each simulation step do
2:   for each outstanding limit orders submitted by this trader do
3:     if  $p \in U(0, 1) < \delta$  then Cancel this limit order
4:     end if
5:   end for
6:    $M_t = (1 - \alpha)M_{t-1} + \alpha(p_t - p_{t-1})$ 
7:    $\theta \leftarrow \frac{\beta \tanh(\gamma M_t)}{N}$ 
8:    $\mu \leftarrow \theta * \rho$ 
9:   if  $p \in U(0, 1) < \theta$  then
10:    if  $M_t > 0$  then Place a buy limit order with calculated price
11:    else if  $M_t < 0$  then Place a sell limit order with calculated price
12:    end if
13:  else No limit order submitted
14:  end if
15:  if  $p \in U(0, 1) < \mu$  then
16:    if  $M_t > 0$  then Place a buy market order
17:    else if  $M_t < 0$  then Place a sell market order
18:    end if
19:  else No market order submitted
20:  end if
21: end for
22:
23:  $*\alpha : \alpha_L, \beta : \beta_L, N : N_{LMT}$  for Long-term Momentum Trader
24:  $*\alpha : \alpha_S, \beta : \beta_S, N : N_{SMT}$  for Short-term Momentum Trader

```

Short-term Momentum Trader (SMT)

Compared to the long-term momentum traders, short-term momentum traders are associated with a much larger α . Typically, α for a short-term momentum trader would be a value close to 1. Following equation 5.2, the momentum signal is updated very fast when α has a large value. In this circumstance, M_t represents the trend in an extremely short time scale, in particular, only several ticks in our intra-day simulation. The group of short-term momentum traders aim to exploit the price trend in a very short time scale, which mimics the behaviour of some real-world high-frequency traders. There are N_{SMT} short-term momentum traders in our agent-based financial market simulation. We assign a value of 0.9 to α of short-term momentum traders.⁵ The trading heuristics for short-term momentum traders are exactly the same as that for long-term momentum traders, except that α_L and β_L are replaced by α_S and β_S , respectively. A full description of short-term momentum traders' logic is shown in Algorithm 2.

⁵We also run simulation experiments with α having values 0.95 and 0.99. Similar experimental results are obtained in these settings, showing that minor changes in α value will not significantly affect the results.

Algorithm 3 Noise Trader Logic

```

1:  $\theta \leftarrow \frac{\sigma_{NT}}{N_{NT}}$ 
2:  $\mu \leftarrow \theta * \rho$ 
3: for each simulation step do
4:   for each outstanding limit orders submitted by this trader do
5:     if  $p \in U(0, 1) < \delta$  then Cancel this limit order
6:     end if
7:   end for
8:   if  $p \in U(0, 1) < \theta$  then
9:     if  $p \in U(0, 1) < 0.5$  then Place a buy limit order with calculated price
10:    else if  $p \in U(0, 1) > 0.5$  then Place a sell limit order with calculated price
11:    end if
12:  else No limit order submitted
13:  end if
14:  if  $p \in U(0, 1) < \mu$  then
15:    if  $p \in U(0, 1) < 0.5$  then Place a buy market order
16:    else if  $p \in U(0, 1) > 0.5$  then Place a sell market order
17:    end if
18:  else No market order submitted
19:  end if
20: end for

```

5.2.6 Noise Trader (NT)

Another group of market participants are noise traders. They are designed so as to capture other market activities that are not reflected by trend-following and value investing. As a result, the cumulative demand of noise traders can be described by a random walk, with each step having an equal probability of placing buy/sell orders. Parameter σ_{NT} controls the aggregated demand level from noise traders. There are N_{NT} noise traders in the model, with each noise trader having value $\frac{\sigma_{NT}}{N_{NT}}$ for θ . Similar to momentum traders, noise traders are also associated with a parameter ρ , which controls the ratio between the number of limit and market orders. Note that unlike other traders whose θ and μ vary according to simulated market conditions, θ and μ for noise traders are determined prior to simulation and remain fixed during the simulation process. When noise traders place limit orders, the price determination process is identical to the process used by momentum traders - both involve sampling a distance from an empirically calibrated log-normal distribution. Algorithm 3 gives a description of noise traders' logic in the simulated financial market.

Trading condition	Normal trading	Stressed trading (after inventory limit reached)	Trading suspension
Order type	Limit order	Market order	None

Table 5.2 Order types submitted by market makers under different trading conditions

5.2.7 Market Maker (MM)

The market makers are another group of traders in the model. The introduction of market makers in the proposed model is aimed at creating realistic limit order book dynamics. Market makers in the proposed model are more complex than previous traders. During normal trading time, market makers only submit quotes to the market. A quote includes one buy limit order and one sell limit order. The simplification that market makers only submit limit orders is motivated by [Men13], which finds that around 80% of market makers' orders are passive. The price of the sell (buy) order is calculated by adding (subtracting) a distance from the mid-price at the corresponding timestamp, where the distance is sampled from a uniform distribution. (The price is rounded to the closest multiple of tick size before being submitted to the exchange.) In alignment with the market making behaviours during the 2010 Flash Crash event [SC10], market makers in the proposed model are associated with a position limit. Specifically, once the inventory of a market maker reaches the position limit, the market maker will stop all active quoting and actively submit market orders to reduce the inventory level. This will continue until the inventory reduces to a certain safe level, which is also a parameter of the model. At this stage, the market maker suspends trading for a certain period of time. This resembles the real-world scenarios that market makers tend to suspend trading to check their own trading systems and observe market conditions after some unusual scenarios happen. After this period of time, the market maker will restart the normal trading heuristics. Table 5.2 presents the corresponding order types that market makers will submit in different trading conditions.

Since market makers only submit limit orders during the normal trading time, the μ for each market maker is set to 0. Similar to the case in previous traders, δ and θ control the probability of order cancellation and limit order submission respectively, whose values are calibrated to historical price time series. Note that one difference is that θ controls the probability of submitting a quote (buy and sell limit orders) for market makers. In addition to δ , θ and μ , market makers are associated with several extra parameters that correspond to the trading behaviours presented above. As mentioned, the price information of the sell (buy) limit order is calculated by adding (subtracting) a distance from the mid-price. This distance is the trading edge of market makers. The trading edge is sampled from a

Algorithm 4 Market Maker Logic

```

1: Before simulation starts:
2:    $\theta \leftarrow \theta_{MM}$ 
3:    $\mu \leftarrow 0$ 
4:    $\delta \leftarrow \delta_{MM}$ 
5:    $Flag \leftarrow 0$ 
6:    $Restart\_step \leftarrow 0$ 
7:
8: for each simulation step do
9:   Get current simulation step  $current\_step$ 
10:  if  $|position| \geq \varepsilon_{limit}^{MM}$  then
11:     $Flag \leftarrow 1$ 
12:     $Restart\_step = Current\_step + \varepsilon_{rest}^{MM}$ 
13:  end if
14:
15:  if  $(Flag == 1) \wedge (|position| \leq \varepsilon_{safe}^{MM})$  then  $Flag \leftarrow 0$ 
16:  end if
17:
18:  if  $Flag == 1$  then
19:    Cancel all active limit orders
20:    if  $|position| < 0$  then
21:      Submit a buy market order
22:    else
23:      Submit a sell market order
24:    end if
25:
26:  else if  $Current\_step > Restart\_step$  then
27:    if  $p \in U(0, 1) < \delta$  then
28:      Cancel existing quotes by this market maker
29:    end if
30:
31:    if  $p \in U(0, 1) < \theta$  then
32:      Submit a quote(one limit buy order and one limit sell order, with sampled trading edge.)
33:    else
34:      No limit order submitted
35:    end if
36:
37:  end if
38: end for

```

uniform distribution, which controls the spread of the quotes submitted by market makers. Specifically, the uniform distribution from which the price distance is sampled has a minimum value of 0 and a maximum value of p_{edge}^{MM} . ε_{limit}^{MM} represents the position limit for each market maker, while ε_{safe}^{MM} denotes the safe position level. That is, market makers will actively reduce inventory once ε_{limit}^{MM} is reached until the position is reduced to ε_{safe}^{MM} level. ε_{rest}^{MM} represent the time length for the trading suspension. Algorithm 4 presents a full description of market makers' trading heuristics in the proposed model.

5.2.8 Experimental Setup for Simulation

The above are the five types of agents included in the proposed model. A fully functioning limit order book is implemented, as is the case in most electronic financial markets. The simulation is run in pseudo-continuous time. Specifically, each simulation step represents 100 milliseconds of trading time. Each trading day is divided into $T = 324,000$ steps, corresponding to 9 hours of trading (8:00 to 17:00). The minimum time for execution of transactions is 100 milliseconds, showing that the simulated financial market represents a high-frequency trading environment. We remind here that the fundamental value V_t , which is extracted from the historical price time series, is an exogenous signal that is input to the model.

The whole simulation runs as follows. For each step, each trader collects and processes market information. Internal variables associated with each trader are calculated. Traders take actions based on agent type and values of internal variables. These actions include limit order submission, market order submission, and order cancellation. The programmed matching engine matches these orders and updates the state of the limit order book. Finally, transactions and limit order book status are published to all traders. Market dynamics are analysed after simulated price time series and order flow are generated. The whole simulation procedure is shown in Algorithm 5.

We suggest that the proposed five types of traders reflect a sufficiently realistic and diverse market environment. According to [O'H95], there are three major market-microstructure trader types: uninformed traders, informed traders and market makers. The noise traders in our model correspond to uninformed traders, while market makers in the proposed model obviously correspond to the market makers in literature. The remaining three types of traders represent informed traders in our model. Specifically, fundamental traders utilise exogenous information implied by the fundamental value, while the two types of momentum traders exploit endogenous technical indicator information. In addition, among the informed traders some perceived trading opportunities are based only on an analysis of short-horizon returns, while others focus on market information revealed by long-term return horizons. This is reflected by the division of momentum traders into long-term and short-term momentum traders. Overall, a sea of different informed and uninformed traders in the proposed model compete with each other, with market makers providing liquidity and ensuring realistic limit order book behaviours. Note that in our model, traders' strategies are fixed and traders cannot switch their strategy according to market conditions. One may argue that in real-world financial markets, traders usually choose and

Algorithm 5 Simulation Procedure

```

1: Simulation Parameter Initialization
2: while timestamp < market close do
3:   timestamp +=  $\Delta t$ 
4:
5:   for each trader do
6:     Collect latest market information and update internal states
7:     if fundamental trader then
8:       Fundamental trader logic(Algorithm 1)
9:     end if
10:    if long-term momentum trader then
11:      Long-term momentum trader logic(Algorithm 2)
12:    end if
13:    if short-term momentum trader then
14:      Short-term momentum trader logic(Algorithm 2)
15:    end if
16:    if noise trader then
17:      Noise trader logic(Algorithm 3)
18:    end if
19:    if market maker then
20:      Market maker logic(Algorithm 4)
21:    end if
22:  end for
23:
24:  for each limit order, market order, or order cancellation) do
25:    Update limit order book and publish transactions
26:  end for
27:
28:  Publish latest limit order book status
29: end while

```

switch trading strategies according to market conditions. However, we can still argue that our model captures this type of behaviour. Different types of traders in our model capture different types of real-world trading strategies. A real trader's strategy is captured by one type of agent under one condition, and by another type of agent under a different condition, should the real trader pick another trading strategy under the new market condition. This is another strength of our approach: with the simplest model and the least number of assumptions, complex financial market situations can be analysed. In conclusion, the proposed model with five types of traders represents a complete range of micro-behaviours of real financial markets.

Fundamental Value from Kalman Smoother

The only remaining unknown variable is the fundamental value of the stock. The simulation can proceed only if the fundamental value is known and is exogenously input to the model. One difficulty

is the non-observability of the fundamental value. According to [MM61], the fundamental value of a stock equals the expected value of discounted dividends that the company will pay to the shareholders in the future. However, this methodology requires extremely strong assumptions about the future dynamics of the stock dividends. Furthermore, this approach can never reflect the intra-day change of fundamental value, while the consensus fundamental value can indeed vary during the trading day due to the continuous feed of events and news.

The fundamental value is derived using the same methodology as outlined in Chapter 3. A Kalman Smoother [RdB05] is applied directly to the stock price time series to get the hidden fundamental value. In accordance with [MCB20], we assume the fundamental value V_t is a hidden variable of a linear dynamical system. The observations are the actual prices traded in real markets. The specific Kalman Smoothing algorithm used here can be found in [BSS04]. The algorithm is applied to the price time series for each trading day to extract the corresponding fundamental value time series for that day.⁶

5.3 Model Calibration and Validation

In this section, we present the methodology for calibrating and validating the agent-based financial market. Calibration means finding an optimal set of model parameters to make the model generate the most realistic simulated financial market. In Chapter 3 a model calibration method based on the XGBoost machine learning algorithm is proposed; In this chapter, we employ the same methodology, with subtle changes implemented. The model validation is carried out via various statistical methods.

5.3.1 Calibration Target: Data and Stylised Facts for Realistic Simulation

In the model calibration process, real financial market data is essential for setting up the calibration target. We collected high-frequency limit order book data of E-mini S&P 500 futures, from May 3rd, 2010 to May 6th, 2010⁷. The data are purchased from the official CME DataMine platform⁸. We select the most liquid contract as the calibration target, which is the contract expires in June 2010. Our dataset comprises high-frequency information for 10 levels of limit order book updates, both the buy side and sell side.

⁶The algorithm is already implemented in Python package "pykalman".

⁷Evening trading sessions are excluded. That is, the data span from 8:00 in the morning to 5:00 in the afternoon for each day.

⁸<https://datamine.cmegroup.com>

Date	20100503	20100504	20100505	20100506
Hill Estimator	0.019	0.046	0.061	0.095

Table 5.3 Hill Estimator values for all trading days

Financial price time series data display some interesting statistical characteristics that are commonly called stylised facts. As is the case in Chapter 3, we choose three major stylised facts for detailed analysis: fat-tailed distribution of returns, autocorrelation of returns, and volatility clustering. The major difference from the analysis in Chapter 3 is that the analysis in this Chapter is carried out in financial time series of higher frequency. Additionally, a different metric called "Hill Estimator of the tail index" replaces the K-S statistic to measure the fat-tailed distribution of returns, which provides more robust results.

Fat-tailed distribution of returns

The distributions of price returns have been found to be fat-tailed across all timescales. In other words, the return distributions exhibit positive excess kurtosis. Understanding positively kurtotic return distributions is important for risk management since large price movements are much more likely to occur than in commonly assumed normal distributions.

Following the convention in literature, we investigate the stylised fact of fat-tailed returns by examining second-level intra-day price returns. Both millisecond-level historical and simulated mid-price time series are resampled into second-level frequency and we examine the mid-price returns for each second. Specifically, the last price snapshot is taken as the price for that specific second. Second-level price returns are calculated accordingly. Our experiments show that different time scales have no significant influence on the final results. The main metric used for evaluating the fat-tail characteristic is the Hill Estimator of the tail index [Hil10]. A lower value of the Hill Estimator implies that the return distribution has a fatter tail. Table 5.3 illustrates the Hill Estimator values for all trading days in our dataset⁹.

⁹To mitigate the impact of the flash crash event and prevent overfitting, the Hill Estimator for May 6th is computed exclusively using morning trading data (8:00 - 12:30).

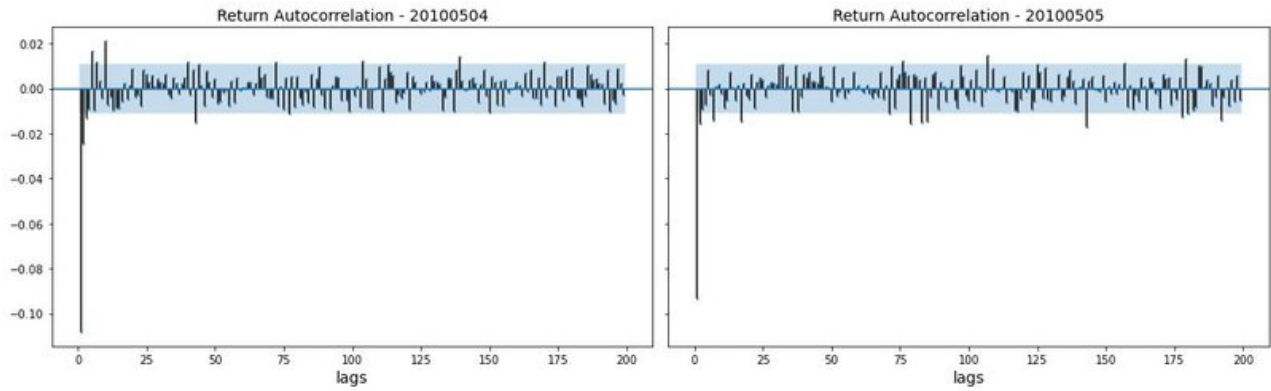


Figure 5.1 Autocorrelation function of second-level returns for E-mini S&P 500 futures on two days

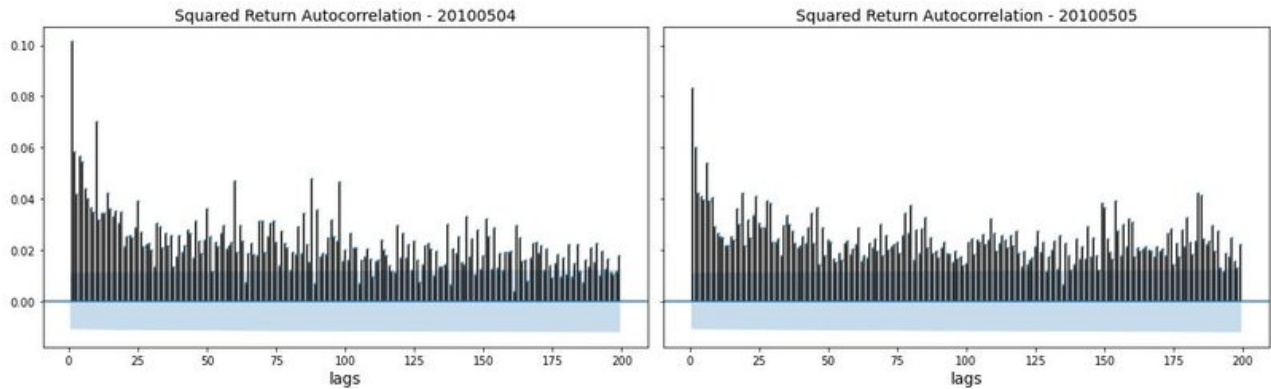


Figure 5.2 Autocorrelation function of squared second-level returns for E-mini S&P 500 futures on two days

Autocorrelation of returns

The specific definition of autocorrelation of returns stylised facts is already presented in Section 3.3.2 of Chapter 3. Examination of our data also reveals this stylised fact. Figure 5.1 shows the autocorrelation function of the second-level return time series for E-mini S&P 500 futures on two days. We can see that the autocorrelation is significantly negative for very small lags (1 and 2), and the negative autocorrelation disappears for larger lags. This reflects the "bid-ask bounce" phenomenon in the market microstructure.

Volatility clustering

The specific definition of volatility clustering stylised facts is already presented in Section 3.3.2 of Chapter 3. Figure 5.2 shows the autocorrelation patterns for squared second-level returns for E-mini S&P 500 futures on two days. It is shown that the volatility clustering stylised fact clearly exists in our collected E-mini S&P 500 futures price dataset.

Stylised Facts Distance as Loss Function

The method for constructing the loss function using the stylised facts distance is the same as the approach in Section 3.3.2 of Chapter 3, except that the K-S statistic metric is replaced by the Hill Estimator of tail index. All other parts are kept the same. **For the convenience of readers, we repeat the description of the loss function construction process here; readers who are already familiar with this process are encouraged to skip this section and jump to the calibration results (Section 5.3.2).**

The target for agent-based model calibration is to find an optimal set of model parameters to make the model generate a realistic simulated financial market. To solve this optimisation problem, it is essential to have a metric that is able to quantify the "realism" of a simulated financial market. First of all, a realistic simulated financial market must exhibit similar characteristics to real financial markets, such as the fat-tailed return distribution and volatility level. In addition, realistic simulated financial data are also required to reproduce other stylised facts such as the autocorrelation patterns in returns and squared returns. Here we design a stylised facts distance to quantify the similarities between simulated and historical financial data. Four metrics are considered in the stylised facts distance: Hill Estimator of the tail index for absolute return distributions, volatility, autocorrelation of returns and autocorrelation of squared returns. For each metric, the differential quantity between simulated value and historical value is calculated. The stylised facts distance is then calculated as the weighted sum of the four differential quantities:

$$D = w_1 * \Delta_{Hill} + w_2 * \Delta_V + w_3 * \Delta_{ACF^1} + w_4 * \Delta_{ACF^2} \quad (5.3)$$

Detailed calculations of the four quantities in the stylised facts distance are presented below.

The Hill Estimator is famous for inferring the power behaviour in the tails of experimental distribution functions. Following [FW12], we use the Hill Estimator of the tail index to estimate the degree of fat-tail in the distribution of absolute returns on the mid-price. Note that the absolute return distribution is considered since there is no need to distinguish between extreme positive and negative returns. In our experiments, the Hill Estimator for simulated absolute returns and historical absolute returns are calculated, respectively. The absolute difference between the two Hill Estimators constitutes the first part of the stylised facts distance:

$$\Delta_{Hill} = |Hill_s - Hill_h| \quad (5.4)$$

where $Hill_s$ and $Hill_h$ are the Hill Estimator for simulated absolute return distribution and historical absolute return distribution, respectively. Intuitively, this part of the stylised facts distance quantifies the distance between simulated and historical return distribution, in terms of the heavy-tailedness. It addresses the requirement that the simulated return distribution exhibits the fat tail property, as is the case in historical return distribution.

The second part of the stylised facts distance is the absolute volatility difference between simulated returns and historical returns:

$$\Delta_V = |V_s - V_h| \quad (5.5)$$

where V_s and V_h denote simulated volatility and historical volatility, respectively. The volatility measure is annualised after being computed from intra-day returns. This part addresses the requirement that a simulated financial market should be similar to real markets in terms of volatility.

The third part of the stylised facts distance is the difference between simulated and historical autocorrelations of returns. This part of the stylised distance measures the model's ability to reproduce autocorrelation patterns commonly found in historical returns. It is shown that financial price return time series lack significant autocorrelation, except for short time scales, where significantly negative autocorrelations exist. This phenomenon is backed by our empirical data. For very small lags the autocorrelations are negative, while for larger lags the autocorrelations become insignificant. To measure the distance in autocorrelation patterns between simulated data and historical data, we invoke the autocorrelation function of returns and calculate the average absolute difference between autocorrelations of simulated return time series and historical return time series for various lags:

$$\Delta_{ACF^1} = \frac{\sum_{l \text{ in } lags} |ACF_s(l, r) - ACF_h(l, r)|}{|lags|} \quad (5.6)$$

where $ACF_s(l, r)$, $ACF_h(l, r)$ are the autocorrelation function of lag l for simulated returns and historical returns, respectively. $|lags|$ denotes the number of lags used in the calculation. Because the empirical autocorrelations are negative for very small lags and close to zero for larger lags, it is not necessary to consider all of the autocorrelation coefficients. Empirical evidence suggests that the autocorrelation pattern is well represented by the coefficients for three lags: 30, 60, and 90. Also, to reduce the effects

of accidental outliers, the autocorrelation function is smoothed by calculating the three-lag average. That is, the lag 30 autocorrelation is calculated as the average autocorrelation of lag 30, 31, 32, and so is the calculation for lag 60 and lag 90. In total, autocorrelations of 9 lags (30, 31, 32, 60, 61, 62, 90, 91, 92) are considered and included in the calculation.

The last part of the stylised facts distance is the difference between simulated and historical autocorrelations of squared returns. The replication of autocorrelation patterns in squared returns indicates the model's capability to reproduce the volatility clustering stylised fact. It is shown empirically that large price changes tend to be followed by other large price changes, known as the volatility clustering phenomenon. Consequently, though there are generally no significant patterns in autocorrelations of returns, the autocorrelations of squared returns are significantly positive, especially for small time lags. Also, as time lag increases, the autocorrelation of squared returns displays a slowly decaying pattern, as shown in Figure 5.2. Similar to the difference between autocorrelations of returns Δ_{ACF^1} , the difference between autocorrelations of squared returns is calculated as follows:

$$\Delta_{ACF^2} = \frac{\sum_{l \text{ in } lags} |ACF_s(l, r^2) - ACF_h(l, r^2)|}{|lags|} \quad (5.7)$$

where $ACF_s(l, r^2)$, $ACF_h(l, r^2)$ are the autocorrelation function of lag l for simulated squared returns and historical squared returns, respectively. $|lags|$ denotes the number of lags used in the calculation. Here we use a slightly different $lags$ from the case for autocorrelations of returns calculation. Since empirical autocorrelations of squared returns are significantly positive and slowly decaying, we consider the autocorrelations of squared returns for four lags: 1, 30, 60, and 90. The three-lag average smoothing is also applied in the calculation. In total, the autocorrelations of 12 lags (1, 2, 3, 30, 31, 32, 60, 61, 62, 90, 91, 92) are considered and included in the calculation.

The above four parts, along with the corresponding weights, constitute the stylised facts distance in Equation 5.3. The next question is how to determine the associated weight for each part of the stylised facts distance. The basic guiding idea is that the higher the sampling variability of a given part in historical data, the larger the difference between simulated value and historical value that can still be deemed insignificant. A natural candidate for each weight is the inverse of the sampling variance for the corresponding part in the stylised facts distance:

$$w_i = \frac{1}{\sigma_i^2} \quad (5.8)$$

Following [FW12], the sampling variance σ_i^2 for each part in the stylised facts distance is estimated by applying the block bootstrap method on the historical return time series. The choice of weighting is inspired by the Method of Moments estimation, where the inverse of the covariance matrix is used. Interested readers are referred to [Woo01] and [FMS95].

Note that the stylised facts distance is a function of model parameters. In other words, given a set of model parameters, there is a unique stylised facts distance calculated from the simulated time series, which corresponds to that particular set of model parameters. Let $\boldsymbol{\theta}$ denote the vector of model parameters to be estimated, Equation 5.3 can be rewritten as:

$$D(\boldsymbol{\theta}) = w_1 * \Delta_{Hill}(\boldsymbol{\theta}) + w_2 * \Delta_V(\boldsymbol{\theta}) + w_3 * \Delta_{ACF1}(\boldsymbol{\theta}) + w_4 * \Delta_{ACF2}(\boldsymbol{\theta}) \quad (5.9)$$

The smaller $D(\boldsymbol{\theta})$ is, the more realistic the simulation is. Thus $D(\boldsymbol{\theta})$ serves as the loss function that the calibration method aims to minimise by finding an optimal set of model parameters. Let Θ denote the admissible set for model parameter vector $\boldsymbol{\theta}$, the calibration target is to find the optimal model parameter vector $\hat{\boldsymbol{\theta}}$ that minimises the stylised facts distance:

$$\hat{\boldsymbol{\theta}} = \arg \min_{\boldsymbol{\theta} \in \Theta} D(\boldsymbol{\theta}) \quad (5.10)$$

5.3.2 Calibration Workflow and Results

The model is calibrated by choosing values for the model parameters so that the dynamics of the simulated price time series match those observed empirical price time series. Specifically, the aim of the calibration process is to find $\hat{\boldsymbol{\theta}}$ that minimises the stylised facts distance, as specified in Equation 5.10. As mentioned, the data used in the calibration are the price time series data of the most liquid contract of E-mini S&P 500 futures, from May 3rd, 2010 to May 6th, 2010. The model parameters are calibrated for every trading day¹⁰. After a large-scale trial and error, seven model parameters are selected to be calibrated. The seven parameters are: μ_ℓ , σ_{NT} , κ_1 , κ_2 , β_L , β_S , θ_{MM} . Specific meanings of these seven parameters are presented in the previous section. In terms of the replication of stylised facts,

¹⁰Due to the flash crash event in the afternoon trading session on May 6th, 2010, only the first half of the trading data (8:00-12:30) is used to calibrate the model parameters for this trading day.

Date	μ_ℓ	σ_{NT}	κ_1	κ_2	β_L	β_S	θ_{MM}	$D(\hat{\theta})$
20100503	0.9093	0.7895	0.1632	0.0235	0.0924	0.0252	0.7805	0.2259
20100504	1.2717	0.6732	0.0933	0.3402	0.6964	0.681	0.8998	0.2269
20100505	1.7396	0.6176	0.2844	0.2098	0.5935	0.055	0.8291	0.2191
20100506	1.0098	0.0107	0.3697	0.1649	0.7494	0.9877	0.3265	0.1666

Table 5.4 Calibrated values for the seven model parameters and the yielded stylised facts distances

our experiments show that the simulation results are less sensitive to the values for other parameters. As a result, it is reasonable to keep other parameters fixed. This choice significantly reduces the computational complexity of the calibration process. These parameters and corresponding descriptions, as well as specific values, are presented in Appendix A and Appendix B.

The calibration workflow in our experiments consists of two stages. The first stage follows exactly the same calibration approach outlined in Chapter 3. Refer to Section 3.3.3 for a detailed description of this process. After the first stage of the calibration, quasi-optimal parameters are given by the machine learning surrogate. Even though global optimum is not guaranteed, it is shown in our experiments that the obtained parameter combination yields small stylised facts distance and is capable of reproducing realistic price dynamics. It is likely that the global optimal parameters are located close to the parameters generated by the first stage. Taking this into consideration, a numerical grid search over a feasible bounded set of parameters is carried out. The feasible set of parameters is centred around the optimal parameters given by the surrogate modelling approach in stage one. The parameter combination that yields the smallest stylised facts distance is selected as the final calibrated model parameter combination. Calibrated model parameter values for each trading day, as well as the yielded stylised facts distance, are presented in Table 5.4.

Figure 5.3 compares the empirical time series of mid-price on May 5th to the simulated mid-price time series. Visual inspection shows that the model produces price time series whose dynamics are very similar to those in empirical data. Nevertheless, a quantitative assessment is required to validate the proposed simulation model, which is presented in the subsequent section.

5.3.3 Model Validation

Table 5.4 shows the stylised facts distance of the calibrated model. However, the value itself does not present an intuitive description of how well the simulated data fit empirical data. A cross-check on the validity of our model topology and calibration strategy is needed. Following [FW12], two metrics are



Figure 5.3 Comparison between empirical mid-price time series on May 5th and simulated mid-price time series on May 5th

used for assessing the quality of the moment matching and the validity of the model simulation: the moment-specific p-value and the moment coverage ratio.

Statistical Hypothesis Testing: Moment-specific p-value

The stylised facts distance provides us with numerical values for the realism of the simulation. However, one more elementary question needs to be addressed: whether the data generated by model calibration and simulation would be rejected by the empirical data. The question is answered by calculating a moment-specific p-value as a statistical hypothesis test.

Recall that in previous sections, the weights are calculated by the block bootstrapping method proposed in [FW12]. While the variance of block bootstrapping samples is the corresponding weight for each part of the loss function, the large number of samples obtained in this procedure can also be used to apply the loss function D to them. In this way, an entire frequency distribution of values of D is available, which can subsequently be contrasted with the simulated distribution of values of D . The fundamental idea is that the set of bootstrapped samples of return time series is a proxy of the set of different return time series samples that could be produced if the hypothetical real-world data generation process exists. Accordingly, if a simulated return time series yields a value of D within the range of the bootstrapped values of D , this simulated return time series is difficult to distinguish from a real-world series.

Note that a Monte Carlo experiment is undertaken and the simulation is repeated many times to create a distribution of model-generated D values. In total, two distributions of D values are obtained: bootstrapped distribution and Monte Carlo simulated distribution. Let N denote the number of bootstrapped samples. For consistency, we also select N Monte Carlo samples to create the simulated

Date	20100503	20100504	20100505	20100506
p-value	0.1833	0.4167	0.0833	0.6167

Table 5.5 P-values for the calibrated model for all trading days

distribution. Let D^b and $D^m(\hat{\theta})$ denote the bootstrapped sample and Monte Carlo sample, respectively¹¹. The distributions of the following two sets of D -values are then compared:

$$\begin{aligned} \text{Bootstrap} : \{D^b\}_{b=1}^N \\ \text{Simulated} : \{D^m(\hat{\theta})\}_{m=1}^N \end{aligned} \quad (5.11)$$

The p-value is obtained as follows. A critical value of D is established from the bootstrapped distribution of values of D . Taking account of the rare extreme events with a significance level of 5%, the critical value is defined to be the 95% percentile $D_{0.95}$ of the bootstrapped D . In this way, a simulated return time series will be considered to be inconsistent with the real-world data, and therefore be rejected, if the corresponding D -value is larger than $D_{0.95}$. Take the May 3rd trading day as an example. Here $D = 0.2114$ is obtained as the critical value of the bootstrapped distance function. As for the Monte Carlo simulation, the model-generated distribution of D is prominently wider to the right. The detailed calculation yields that $D_{0.95}$ corresponds to the 18.33% percentile of the simulated distribution of values of D . Following the general moment matching convention in [FW12], the model is said to have a p-value of 0.1833, with respect to the estimated parameter vector $\hat{\theta}$, the historical data for May 3rd, and the specific moments that we have chosen. According to the conventional significance criteria, for this trading day the model will not be rejected as being obviously inconsistent with the empirical data. Following [FW12], a p-value of 0.1833 is actually believed to be a fairly good performance for an agent-based financial market simulator. The p-values for the calibrated model for all trading days are shown in Table 5.5.

From Table 5.5 we can see that for all trading days in the collected dataset, the moment-specific p-values are greater than 0.05. Consequently, we cannot reject the null hypothesis which specifies that the simulated return time series belongs to the same distribution as the empirical return time series. This statistical hypothesis testing gives further evidence that the calibrated model is capable of generating realistic financial price time series.

¹¹The Monte Carlo simulated distance value D is dependent on the estimated set of parameters $\hat{\theta}$.

Moment Coverage Ratio

The previous evaluation of the model is based on the values of the stylised facts distance function D . While statistical testing enables us to evaluate the validity of the model simulation, the quality of moment matching for each specific moment is still unknown. Another potential issue is that the stylised facts distance function D is the optimisation target during the calibration process. Thus the evaluation metric involving the distance D may be biased because of the potential overfitting problem. To address the above problems, the "moment coverage ratio" (MCR) metric is adopted to assess the degree of moment matching, taking into account each specific moment. The moment coverage ratio is originally proposed in [FW12]. The basis for moment coverage ratio calculation is the concept of a confidence interval of the empirical moments. Consistent with [FW12], the 95% confidence interval of a moment is considered, which is defined to be the interval with boundaries ± 1.96 times the standard deviation around the empirical value of this moment. The next step is to determine the standard deviation for each empirical moment. [FW12] apply the delta method to the autocorrelation coefficients to calculate the standard deviation. In this work, we use a more direct way to obtain the empirical standard deviation, which is based on the block bootstrapping method. Recall that large quantities of return time series are obtained by the block bootstrapping method. For each specific moment, a moment value can be calculated out of every sampled return series. In total, there will be B values for each moment, where B is the bootstrapping sample number. The standard deviation of those values is considered to be the standard deviation for the corresponding empirical moment. One may feel uneasy about the bootstrapping of the autocorrelation functions at longer lags since the method alters the temporal order of the return series. However, the block size in our block bootstrapping method is 1800, which is significantly larger than the longest lag (90) in the autocorrelation functions. Consequently, the impact of bootstrapped block re-ordering on the autocorrelation functions is negligible.

With the standard deviation on hand, the corresponding confidence interval for each specific moment is immediately available. In this way an intuitive criterion for assessing a simulated return series is obtained: if all of its moments are contained in the confidence intervals, the simulated return series cannot be rejected as being incompatible with empirical data. Nonetheless, one single simulation is not sufficient to evaluate a model as a whole due to the sample variability. In addition, it is likely that some moments are contained in the confidence interval while others are not for one simulated return series. Consequently, considering multiple simulation runs of the model will provide a more exhaustive assessment of the model's performance. Specifically, for each simulation run the confidence interval

Moment \ Date	20100503	20100504	20100505	20100506
Hill estimator (inverse)	98.3%	66.7%	98.3%	100.0%
Volatility	0.0%	85.0%	100.0%	56.7%
Return autocorrelation (30 lags)	85.0%	78.3%	50.0%	90.0%
Return autocorrelation (60 lags)	100.0%	98.3%	96.7%	96.7%
Return autocorrelation (90 lags)	98.3%	60.0%	100.0%	58.3%
Squared return autocorrelation (1 lag)	100.0%	98.3%	95.0%	93.3%
Squared return autocorrelation (30 lags)	78.3%	51.7%	71.7%	98.3%
Squared return autocorrelation (60 lags)	88.3%	10.0%	80.0%	73.3%
Squared return autocorrelation (90 lags)	91.7%	46.7%	76.7%	51.7%

Table 5.6 Moment coverage ratios of the calibrated model

check is repeated. We count the number of Monte Carlo simulation runs in which the single moments are contained in the corresponding confidence intervals. The corresponding percentage numbers out of all Monte Carlo runs are defined as the moment coverage ratio.

Since the model is calibrated by each trading day, a Monte Carlo simulation is run for each trading day and the corresponding moment coverage ratios are calculated to evaluate the calibrated model. Table 5.6 presents the results of the moment coverage ratios calculation. Except for the volatility moment on May 3rd and the 90-lag squared return autocorrelation moment on May 4th, all other moment coverage ratios are higher than 50%. According to the analysis in [FW12], the higher than 50% moment coverage ratio represents a terrific performance of the model. In addition, almost half of the moment coverage ratios are even higher than 90%, which indicates that our calibrated model has an excellent ability to reproduce realistic stylised facts. Overall, with respect to the selected moments, the calibrated model's capability of matching empirical moments and reproducing realistic stylised facts is highly remarkable.

5.4 2010 Flash Crash Scenario

As a real-world application, the proposed model is used to investigate market dynamics during flash crash events and the conditions for the occurrence of flash crash scenarios. This section will present the reproduction and investigation of a famous historical flash crash event - the flash crash on May 6th, 2010. In the next section, we will investigate conditions for the occurrence of mini flash crash events.

5.4.1 Simulating Historical Flash Crash

We simulate the 2010 Flash Crash within the framework of our high-frequency financial market simulator. The flash crash happened in the afternoon trading session on May 6th, 2010, starting at around 14:30. As mentioned before, the model parameters are calibrated to the data in the morning trading session (8:00-12:30) to avoid overfitting¹². According to the CFTC-SEC staff report, an automated execution algorithm, which aimed at selling a large number of contracts, is identified as one important trigger for the flash crash. Consistent with this report, an institutional trader is introduced in the simulator to mimic the automatic Sell Algorithm. The institutional trader will initiate a Sell Algorithm that intends to sell a large quantity of contracts at 14:30 in our simulator. The parameters associated with the institutional traders are tuned to reproduce realistic flash crash behaviours as in historical data. We present the trading configuration of the institutional trader in the subsequent session, followed by a detailed analysis of the market dynamics during the simulated flash crash event.

Introduction of Institutional Trader (INS)

The institutional trader is specially designed to replicate the behaviour of the large institutional trader who initiated the sell program to sell a large quantity of E-mini contracts on May 6th, 2010. In the beginning, the institutional trader has a large inventory of Q E-mini contracts. The institutional trader will start a sell program to sell the inventory at 14:30 in the simulation. Consistent with the historical scenarios, the sell program in the simulation is executed via an automated execution algorithm that is programmed to submit orders into the market to target a certain execution rate, which is a percentage of the market trading volume calculated over the previous minute. The execution rate is denoted as r , which is set to 9% in our simulation according to [SC10]. Note that the Sell Algorithm only considers the total trading volume in the market, taking no account of the market price or time. This procyclical behaviour is potentially one causal factor in the occurrence of flash crash events. As for specific trading behaviours, the institutional trader always keeps count of the total trading volume V of the last minute, and submits a market sell order every n seconds. For each market order, the volume would be $V * r * \frac{n}{60}$, where $\frac{n}{60}$ is a scaling factor ensuring the target execution rate r is achieved by the sell program. The institutional trader continues trading until all the inventory is cleared. Algorithm 6 describes the trading behaviours of the institutional trader in detail.

¹²Except for market maker inventory limit and fundamental trader trading frequency, which are key to generating realistic flash crashes.

Algorithm 6 Institutional Trader Logic

```

1: Initialise  $n, r, Q$ 
2: for each simulation step do
3:    $V \leftarrow$  total trading volume in the market for the last minute
4:   if (simulation timestamp > 14 : 30 )  $\wedge$  ( $Q > 0$ ) then
5:      $vol \leftarrow V * r * \frac{n}{60}$ ;
6:     if ( $step\_seconds^\dagger \bmod n$ ) == 0 then
7:       Submit a market sell order with volume  $vol$ 
8:        $Q \leftarrow Q - vol$ 
9:     else
10:      No action taken
11:    end if
12:  else
13:    No action taken
14:  end if
15: end for
16:
17:  $\dagger step\_seconds$  is the number of seconds since the beginning of the simulation

```

Market Behaviours during Flash Crash

To generate a realistic flash crash event, we fine-tune the parameters of the newly introduced institutional trader. According to [SC10], the flash crash event is potentially caused by the "hot-potato" effect among high-frequency market makers and the mismatch of the trading frequency between different types of traders. Consequently, we also tune the market maker inventory limit and the trading frequency of fundamental traders. The effect of these parameters on the flash crash event will be presented in subsequent sessions. All other model parameters are fixed and are given exactly the same values that are calibrated to the morning trading session data on May 6th, 2010. This configuration helps to reduce the degree of freedom. In addition, the potential problem of overfitting is avoided since the morning trading session data used for calibration is not included in simulating the flash crash event. The resulting simulation of the price time series mimics the real-world flash crash event. Figure 5.4 presents such a single simulation of the price trajectory that undergoes a flash crash scenario. The corresponding parameter configuration is shown in Appendix C. Figure 5.5 presents a comparison between simulated trading volume and historical trading volume during the whole day. Carefully examining such a single simulation provides useful insights into how the trading behaviour of different types of traders interact to bring about a flash crash scenario.

By visual inspection, the simulation accurately replicated the defining characteristics of a flash crash. Figure 5.6 presents the total inventory level for each type of trader around the simulated flash crash event, and the market sell order volume of the institutional trader. To assist with detailed inventory

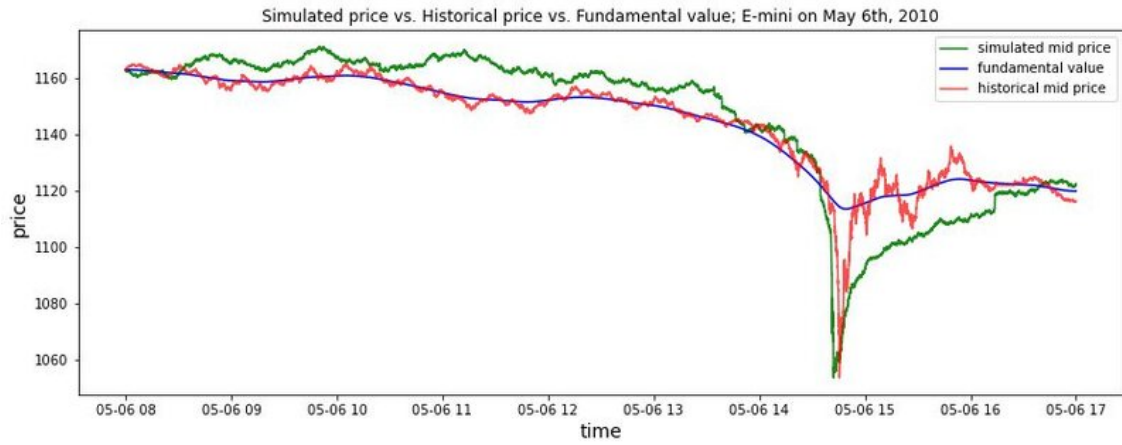


Figure 5.4 One single simulation of the 2010 Flash Crash event. Red line is the historical price; green line is the simulated price; blue line is the fundamental value extracted by Kalman Smoother

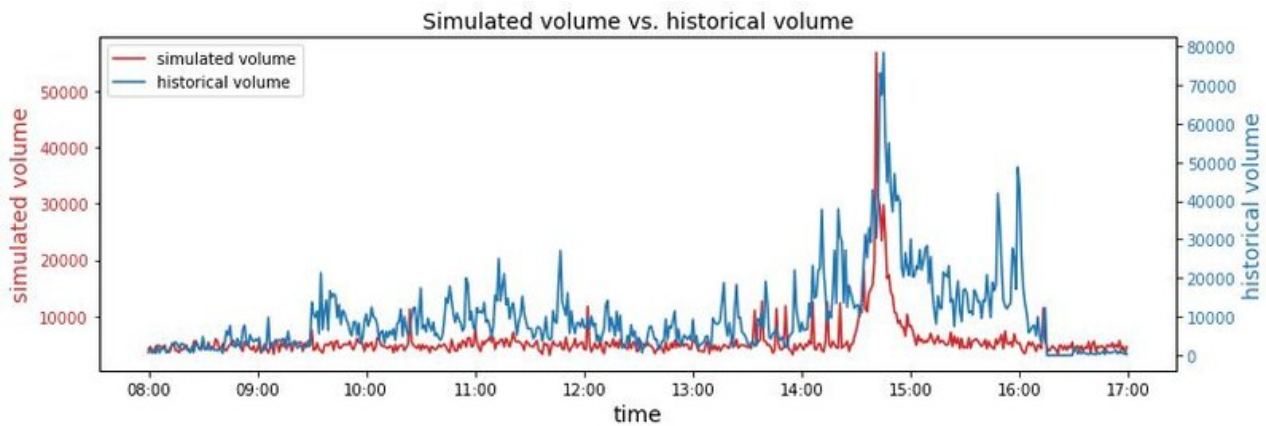


Figure 5.5 Comparison between simulated volume and historical volume on May 6th, 2010

analysis during the flash crash event, Figure 5.7 presents the inventory level for each type of trader against simulated price during the time interval from 14:30 to 14:50. Figure 5.8 displays the market depth for both sides of the simulated limit order book, and the comparison between simulated bid-ask spread and historical bid-ask spread. Detailed examination shows that the simulated flash crash accurately matches the market dynamics during the historical 2010 Flash Crash. The dynamics around the simulated flash crash event are as follows.

- The overall market sentiment in the simulation is negative, as we can see that the fundamental value is broadly decreasing before 14:00. At 14:00, according to Figure 5.6 the fundamental traders accumulate a negative inventory due to continuously decreasing fundamental value, while market makers accumulate a positive inventory. Before the flash crash event, the market depth for both sides of the limit order book is relatively stable, while the simulated bid-ask spread is at a relatively low level.

- At 14:30 in the simulated market, the market price has already dropped by 2.63% compared to the opening price level. Regardless of the downward market trend, the institutional trader initiates a sell program to sell a large number of inventories, which is 120,000 in our simulation. The institutional trader decides to execute the sell program via an automated execution algorithm, which is set to feed market selling orders into the market to target an execution rate of 9%. That is, for each minute the institutional traders aim to sell 9% of the market trading volume calculated over the previous minute. Neither price nor time is considered by the Sell Algorithm. Refer to Algorithm 6 for the specific trading logic of the institutional trader.
- In our simulation, market makers correspond to the high-frequency traders in the real market. The above selling pressure is initially absorbed by market makers in our simulated market, which is consistent with the dynamics in the historical market [SC10]. This is shown by the inventory change for each type of trader in Figure 5.7. From 14:30:00 to 14:40:50, the total inventory for market makers rapidly increased, while the total inventory for low-frequency fundamental traders barely changed. During this approximately 11-minute period, the price undergoes a further 2.61% drop. At 14:40:50, the market depth in the limit order book has slightly decreased and the bid-ask spread has slightly increased. The scale of change for the market depth and bid-ask spread is restricted because of the existence of market makers.
- In addition to the above observations, the market transacted volume is continuously increasing during the above 11-minute time interval due to the trading activities of the Sell Algorithm, as shown in Figure 5.5. Because of its procyclical nature, the Sell Algorithm used by the institutional trader responds to the increased volume by increasing the volume of sell orders that it is feeding into the market, even though the orders that it already submitted to the market were not yet fully absorbed and have caused non-trivial market turbulence. The increasing order volume of the institutional trader is shown in panel (b) of Figure 5.6.
- As market makers are the buyers of the initial batch of orders sent by the institutional trader, they have accumulated massive temporary long positions of the contracts. Nevertheless, the inventory of market makers cannot accumulate infinitely. At 14:40:50, the inventory limits for many market makers are exceeded. At this point, a dramatic and significant market crash starts. Those market makers, who have accumulated excessive positions than their inventory limits, stop providing liquidity and begin to aggressively sell their inventories in order to reduce their temporary long positions. This is consistent with the typical trading practice that a market maker

tends to maintain a relatively small aggregate inventory for the purpose of risk management. Consequently, these market makers contribute further selling pressure to the market in addition to the Sell Algorithm. Still lacking sufficient demand from fundamental traders, the aggregate sell volume is consumed by remaining market makers who are still quoting in the market. As a result, more market makers accumulate excessive positions than their inventory limits, which in turn forces them to sell their long positions to the market. The same positions rapidly pass among all market makers. Such a so-called "hot-potato" effect quickly sweeps almost all market makers in the market, resulting in a dramatic price drop and quoting suspension of almost all market makers.

- The combined selling pressure from the Sell Algorithm and market makers drove the price down by more than 4.16% in less than two minutes from 14:40:50 to 14:42:20, with the price reaching its intra-day low of 1053.5. Along with the price plunge, the market suffers from great liquidity loss. The liquidity loss can be reflected by the sharp decrease in market depth during the simulated flash crash event. In the simulated morning trading session, the average market depth for both bid and ask sides is around 5000. In contrast, during the simulated large price plunge that starts at around 14:40, the market depth for both the bid and ask sides is less than 1000, even reaching 0 on the bid side for more than 1 minute. The significant liquidity loss in our simulation is mostly because of the withdrawal of market makers from the market. The withdrawal of market makers in our simulation is consistent with empirical findings in [SC10], which states that real market makers did stop trading during the 2010 Flash Crash event. According to Figure 5.8, the market depth for both sides of the limit order book dropped dramatically during this period. There is less than 1% of bid-side market depth observed during normal trading hours, with even zero bid-side market depth for around 30 seconds in the middle of 14:41. The bid-ask spread widens dramatically, reaching more than 20 ticks at this short time interval.
- As the price drops quickly, the demand from fundamental traders gradually increases according to Equation 5.1. Figure 5.7 shows that fundamental traders do absorb a portion of selling volume during the sharpest price drop. However, the sudden decline in both price and liquidity indicates that the price was moving so fast that fundamental traders were incapable of providing enough buying support.
- After the price hits the intra-day low level, the demand from fundamental traders finally increases to a level that is able to counteract the selling pressure from the institutional trader. Starting

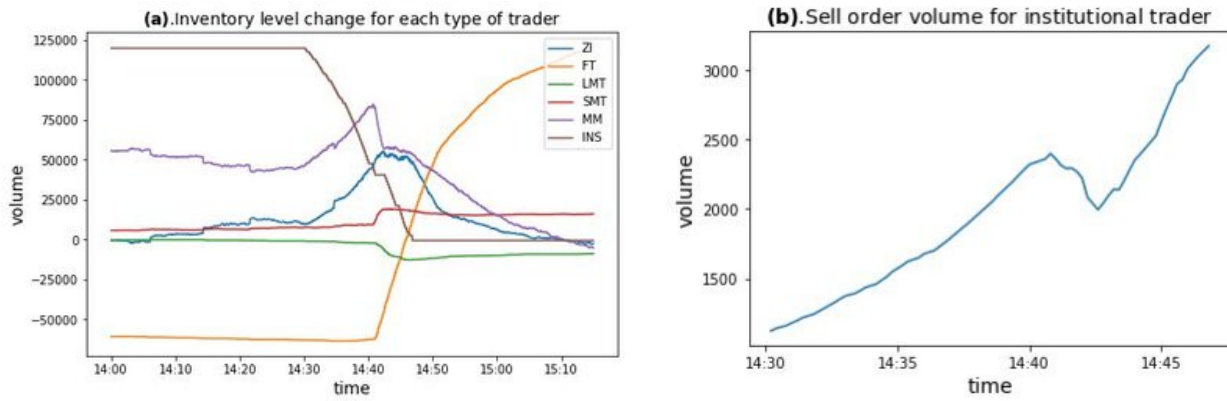


Figure 5.6 (a): Total inventory level for each type of trader, around the simulated flash crash event (14:00 - 15:15). **(b):** The volume change for the selling orders of the institutional trader, during the time when the institutional trader is active

at 14:42:20, the price hovers at the lower level for approximately one minute and then starts to bounce back quickly towards the fundamental value. The Sell Algorithm continues to feed sell orders to the market until about 14:47, at which point the inventory of the institutional trader has been emptied.

The above is a detailed analysis of the market dynamics of a simulated flash crash event. The simulated dynamics accurately match the dynamics of the historical 2010 Flash Crash event. Both simulated flash crash and historical flash crash have an amplitude of around 7% (starting from 14:30:00), and both prices undergo a similar "flash crash" shape. A large Sell Algorithm is replicated to trigger the simulated flash crash, and is executed for around 17 minutes, which is close to the 20-minute execution time of the historical Sell Algorithm. The overall liquidity loss during the flash crash is replicated accurately, with decreased market depth and enlarged bid-ask spread as emergent properties of the simulation. We also reproduce realistic patterns in the simulated market trading volume. The detailed progress of the simulated flash crash also matches the empirical analysis of the historical flash crash event.

In the subsequent section, we will explore the conditions that influence the severity of the flash crash event.

5.4.2 Flash Crash Under Different Conditions

The above presents an analysis of the market dynamics during the simulated flash crash event. A natural question to follow is what factors contribute to the sharp price drop. One significant advantage of the

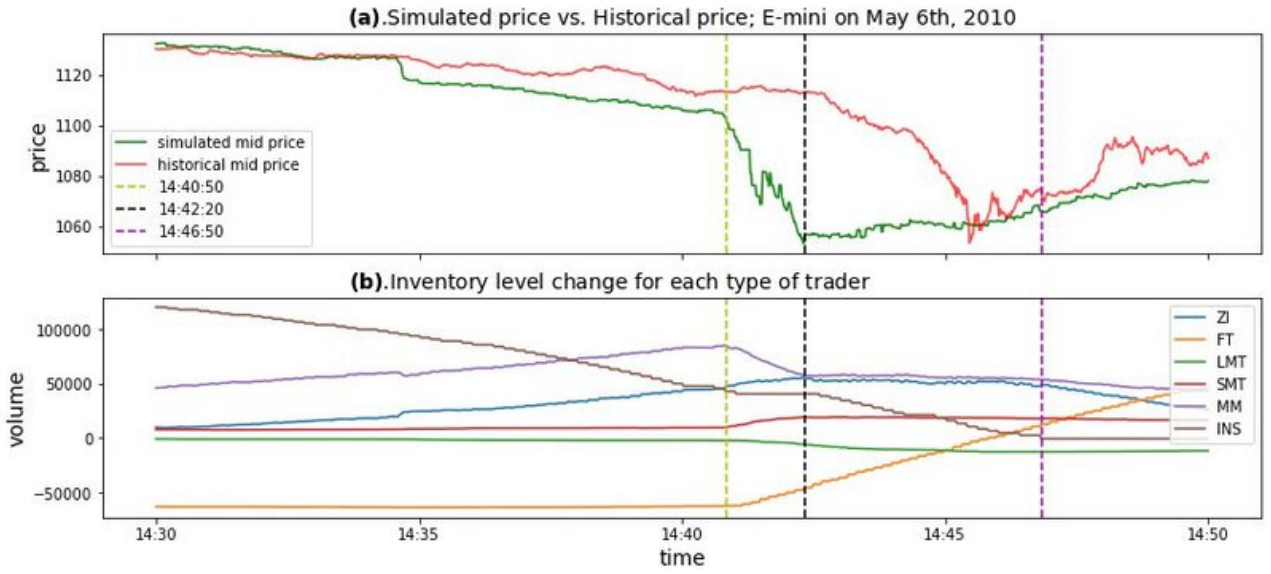


Figure 5.7 Detailed simulated mid-price against the inventories of each category of trader from 14:30 to 14:50. (a): Simulated and historical price from 14:30 to 14:50. (b): Total inventory level for each type of trader during period of time from 14:30 to 14:50

agent-based model is the ability to be simulated multiple times under different conditions to reproduce realistic market events, while the historical event only happens once. This makes the agent-based model a perfect testbed for exploring the conditions that lead to various rare market events, such as the flash crash. In this section, we explore the conditions that would influence the severity of flash crash events.

To measure the severity of flash crash events, we define the amplitude of the flash crash event in our simulation as the maximum percentage price drop after 14:00. Specifically, the flash crash amplitude Amp is calculated as below:

$$Amp = \left| \frac{TWAP_{14:00-14:05} - P_{min}}{TWAP_{14:00-14:05}} \right| \quad (5.12)$$

where $TWAP_{14:00-14:05}$ is the time-weighted average price of the time interval between 14:00 and 14:05, while P_{min} is the lowest price during the flash crash simulation.

The main methodology applied here is Monte Carlo simulation experiments under controlled conditions. The conditions are represented by model parameters. Each combination of model parameters corresponds to a specific market condition configuration. For each combination of model parameters, multiple simulations are carried out. For each simulation, the amplitude of the simulated flash crash is calculated. In this way, a distribution of flash crash amplitudes is obtained. To reduce the influence of extreme values, the 50% quantile of all the Monte Carlo simulated amplitudes is defined as the flash crash amplitude associated with the condition that is represented by the specific combination of model

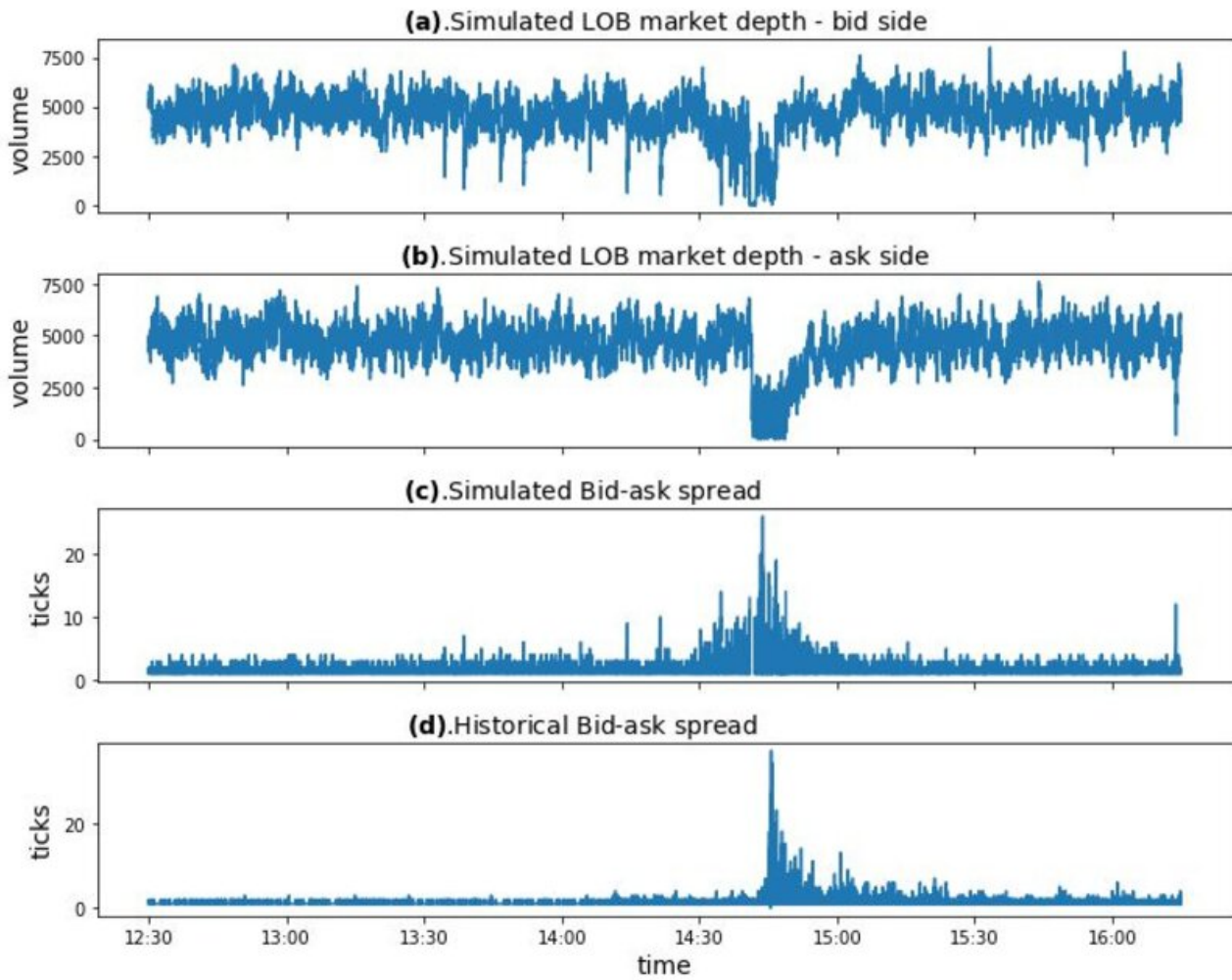


Figure 5.8 (a): Simulated market depth (total resting volume) of bid-side limit order book. (b): Simulated market depth (total resting volume) of ask-side limit order book. (c): Bid-ask spread in the simulated limit order book. (d): Bid-ask spread in the historical limit order book. All four panels display period of time spanning from 12:30 to 16:15

parameters. The 40% and 60% quantiles of the simulated flash crash amplitudes are also recorded. The 40% and 60% quantiles are chosen to reflect the rough variation of the mean simulated flash crash amplitudes.

To explore how one specific condition affects the severity of flash crash events, we change the corresponding model parameter and repeat the above Monte Carlo simulation to get the associated flash crash amplitude. The other model parameters are given the previous calibrated values and are kept fixed during this experimental process. This allows us to observe how the flash crash amplitude changes when only one specific model parameter changes. In particular, we are interested in three model parameters that relate closely to the flash crash amplitude: the percentage of volume for the Sell Algorithm (r), the inventory limit for each market maker (ε_{limit}^{MM}), and the trading frequency for

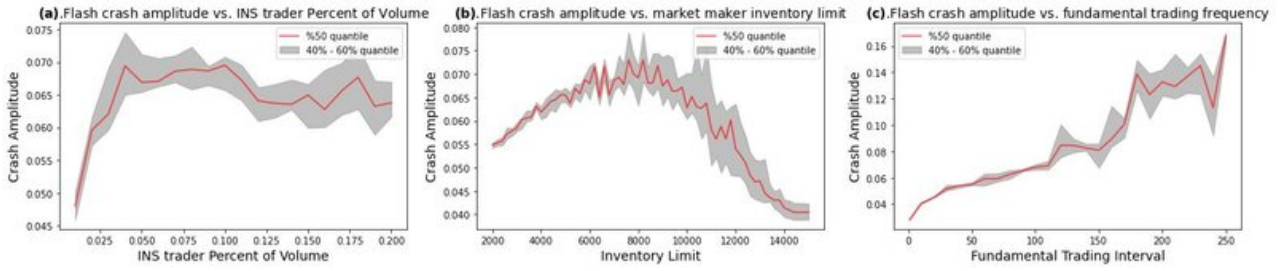


Figure 5.9 (a): Relationship between the amplitude of simulated flash crashes and the inventory limit of market makers. **(b):** Relationship between the amplitude of simulated flash crashes and the percentage of volume r for the Sell Algorithm (Institutional Trader). **(c):** Relationship between the amplitude of simulated flash crashes and the trading frequency of fundamental traders.

fundamental traders (FT_{freq}). Figure 5.9 presents the relationships between these model parameters and the simulated flash crash amplitude. Detailed analysis is as follows.

Percentage of Volume for the Sell Algorithm

One important parameter that characterises the Sell Algorithm is the percentage of volume r , which determines the target execution rate of the Sell Algorithm. Specifically, r is the percentage of the market trading volume calculated over the previous minute that the Sell Algorithm aims to execute. Monte Carlo simulation is carried out with different values for r , while other model parameters are given the previous calibrated values and are strictly kept constant. Panel (a) in Figure 5.9 shows the relationship between the simulated flash crash amplitude and the percentage of volume r .

It is shown that when r is small, the amplitude of the flash crash is limited. When r is increased, the flash crash amplitude is also enlarged. Within a certain range, the more aggressive the Sell Algorithm (larger r), the more severe the flash crash event (larger amplitude). This phenomenon is consistent with our intuition. One interesting phenomenon is that after r exceeds 5%, the flash crash amplitude barely changes even though we continue to increase the value of r . The reason is that as long as the price reaches a certain lower level, the demand generated from fundamental traders is large enough to digest the inventory of the institutional trader. Since the total inventory of the institutional trader is fixed, the lowest price that could be reached during the simulation barely changed. Consequently, the flash crash amplitude stops increasing despite larger values for r .

The lessons learned from this experiment mainly involve the choice of algorithmic trading strategies for large institutional traders. The market impact of an algorithmic trading strategy may contradict our intuition, as is the case here when r is larger than 5%. Instead, the complex interaction between an

algorithmic trading strategy and other market participants is likely to cause unexpected consequences. It is strongly recommended that institutional traders deploy agent-based financial market simulation to explore potential consequences before choosing a specific algorithmic trading strategy.

Inventory limit for market makers

Both empirical and simulation analyses indicate that high-frequency market makers play an important role in the flash crash event. Though market makers initially absorb lots of selling pressure from the institutional trader, they quickly turn into aggressive sellers after their inventory limits are exceeded, creating the "hot-potato" effect that exacerbates the crash. The inventory limit ε_{limit}^{MM} plays an important role in controlling the significance of the "hot-potato" effect. Consequently, the value of ε_{limit}^{MM} also influences the flash crash amplitude. We run Monte Carlo simulations with different values for ε_{limit}^{MM} . Other parameters are given the previous calibrated values and are kept constant. Panel (b) in Figure 5.9 presents the relationship between the simulated flash crash amplitude and the inventory limit for each market maker ε_{limit}^{MM} .

It is shown that the functional relationship between flash crash amplitude and market maker inventory limit is not monotonous. When the ε_{limit}^{MM} is relatively small (smaller than 8000 in our simulation), the flash crash amplitude is an increasing function with regard to ε_{limit}^{MM} . This is because the larger the ε_{limit}^{MM} , the more inventory that the market makers will accumulate. With more accumulated inventory, the market makers will create much more selling pressure at the point when their inventory limits are exceeded, resulting in a more severe crash. However, the flash crash amplitude decreases on increasing the value of ε_{limit}^{MM} if the ε_{limit}^{MM} is larger than a certain level (around 8000 in our simulation). This is because the large enough ε_{limit}^{MM} makes it possible for market makers to absorb a larger portion of selling pressure from the Sell Algorithm. As a result, there are fewer market makers whose inventory limits are reached, curtailing the "hot-potato" effect during the flash crash. In extreme cases when inventory limits for market makers are infinite, the market makers will absorb all selling pressure in the market and there will be no flash crash event.

The analysis here emphasises the importance of risk management for high-frequency market makers. Given that all market makers implement some form of inventory control, the decreasing part of panel (b) in Figure 5.9 is unlikely to occur in a real-world trading environment. Instead, most real-world market makers have inventory limits that lie on the increasing part of the curve. An individual market

maker may consider his own inventory limit to be proper; however, multiple "proper" inventory limits combined may lead to a significant crash under stressed scenarios due to the "hot-potato" effect. In order to foster a smooth and stable market, experimental results suggest that policymakers could impose certain inventory limits on all high-frequency market makers.

Trading frequency for fundamental traders

According to [KNWB18], the mismatch of trading frequency between different types of traders is an important factor that leads to flash crash scenarios. The flash crash happens when the market is dominated by the procyclical behaviours of high-frequency market participants. The price moves so fast that lower frequency fundamental traders are unable to supply enough buy-side liquidity. So what will happen if we change the relative frequency between the market makers and fundamental traders in our simulation? Experiments are carried out with different trading frequencies for fundamental traders. The trading frequency of fundamental traders is changed by changing $S_{interval}^{FT}$, which is the interval between actions from the same fundamental traders. Specifically, once a fundamental trader submits an order to the market, the same fundamental trader is not able to participate in the simulated market within the subsequent $S_{interval}^{FT}$ steps. Only after $S_{interval}^{FT}$ step can this fundamental trader submit another order to the market. In the default calibrated simulation $S_{interval}^{FT}$ is 100, while the same interval for market makers is 1. In other words, market makers can act in every simulation step, while a fundamental trader can only act once in every 100 steps. The smaller the $S_{interval}^{FT}$ is, the higher the trading frequency for fundamental traders. If $S_{interval}^{FT}$ has value 1, the fundamental traders would have the same trading frequency as market makers, creating the imaginary "high-frequency fundamental traders". Panel (c) in Figure 5.9 presents how the flash crash amplitude changes when the trading frequency of fundamental traders changes.

The figure indicates that there is a monotonous relationship between flash crash amplitude and the trading frequency of fundamental traders. Note that the larger the interval is, the lower the trading frequency is. Thus the flash crash amplitude decreases when we increase the trading frequency of fundamental traders. If the fundamental traders have the same trading frequency as the market makers, the flash crash amplitude becomes small enough that the flash crash turns into a small shock in the market.

The analysis here supports the argument that the mismatch of trading frequency between different

types of traders potentially leads to flash crash events. There is hardly an imaginary "high-frequency fundamental trader" in the real-world trading environment. Thus practical trading environment corresponds to the right-hand part of the functional relationship in Panel (c) of Figure 5.9, where fundamental traders act at a lower frequency. The results indirectly show that high-frequency trader is an important factor in the occurrence of flash crash scenarios.

5.4.3 Simulating Other Flash Crash Events

The proposed model effectively simulates the 2010 Flash Crash, demonstrating its realistic representation. Furthermore, its adaptability allows for extensive application in simulating various flash crash events. By fine-tuning model parameters, we successfully extend its capabilities to replicate other notable instances, such as the 2013 Flash Crash in American markets and the 2016 Flash Crash of the British Pound. While constrained by the absence of order book level data, our simulations are able to capture the specific shapes of these flash crash events. The results are not presented here as they are not the primary focus of this thesis. Nevertheless, these simulations showcase the generalisability of the proposed high-frequency agent-based financial market simulator.

5.4.4 Alternative Explanations for Flash Crash Events

The above section focuses on flash crashes caused by internal market dynamics. These include liquidity imbalances, inventory constraints, and the interactions between different types of traders. But flash crashes may also be triggered by external or manipulative actions.

One well-known example is spoofing [FGG21]. Spoofing refers to placing a large number of orders without the intention for them to be executed [CJW20]. These orders are quickly cancelled before they are executed. The primary goal of this behaviour is to deceive other market participants by creating a false supply or demand.

Spoofing can move prices and cause panic in the order book, leading to sudden liquidity withdrawal or price swings. Recent evidence shows that during the 2010 Flash Crash, a trader used spoofing techniques in the E-mini S&P 500 futures market [AGL17]. As a result, a large volume of spoofing orders was submitted to the market and then quickly cancelled. It is believed that this activity also contributed to the flash crash event. While spoofing may not be the main reason for the crash, it contributes to the crash by creating stress in an already fragile market.

This shows that flash crashes can also be the result of market manipulation. Future research could build on this. Our agent-based models could be extended to simulate spoofing behaviour, similar to the work in [WW17]. It would be interesting to test how manipulative strategies impact market stability in our agent-based financial market simulation framework.

5.5 Mini Flash Crash Scenarios

The above section provides an analysis of the occurrence of flash crash events. The 2010 Flash Crash is so large that no following events have rivalled its depth, breadth, and speed of price movement [PHSB17]. However, flash crashes on a smaller scale do occur more frequently. [JZH⁺12] identify more than 18000 mini flash crash incidents between 2006 and 2011 in US equity markets. Those mini flash crashes are characterised by abrupt and severe price changes over a short period of time. One natural question is how those mini flash crashes happened. As a second application for the proposed agent-based model, we investigate and analyse the causes of mini flash crashes in the framework of our agent-based financial market simulation. An innovative "Spiking Trader" is introduced to the market to mimic the trigger of mini flash crash events. A typical mini flash crash event in our simulation is analysed in detail, followed by experiments for exploring the conditions for mini flash crash events.

5.5.1 Introduction of Spiking Trader (ST)

In the current literature, there is a growing consensus that mini flash crashes result from interactions between various trading algorithms that operate at or beyond the limits of human response times, such as the procyclical behaviours of high-frequency market participants. However, there are various triggers for mini flash crash events. According to [KNWB18], one specific trigger could be orders that are large relative to the supply of available limit orders, which bring price shocks to the market. To mimic the occasional price shock in the market, a type of trader agent called "Spiking Trader" is introduced to our agent-based artificial financial market.

The proposed spiking trader mainly creates price shocks to the market, either upward or downward. Initially, the spiking trader is inactive. For each simulation step, there is a small probability that a spiking trader is activated. Once activated, the spiking trader will submit market orders of the same direction for several consecutive simulation steps, creating price shocks in one direction. A spiking

Algorithm 7 Spiking Trader Logic

```

1: Initialise  $N_{spike}$ ,  $\mu_{spike}$ ,  $V_{spike}$ 
2:  $Status \leftarrow 0$ ;  $D \leftarrow \text{Sell}$ 
3: for each simulation step do
4:   if  $Status > 0$  then
5:     Submit a market order with volume  $V_{spike}$ , direction  $D$ 
6:      $Status \leftarrow Status - 1$ 
7:   else
8:     if  $p \in U(0, 1) < \mu_{spike}$  then
9:       if  $p \in U(0, 1) < 0.5$  then  $D \leftarrow \text{Sell}$  else  $D \leftarrow \text{Buy}$  end if
10:       $Status \leftarrow N_{spike}$ 
11:    else
12:      No action taken
13:    end if
14:  end if
15: end for

```

trader is associated with three parameters: N_{spike} , μ_{spike} , and V_{spike} . N_{spike} represents the number of consecutive orders to be submitted after the spiking trader is activated. μ_{spike} is the probability of being activated in a certain simulation step. V_{spike} is the volume of the orders from spiking traders. Detailed trading logic for spiking traders is shown in Algorithm 7. Note that N_{ST} spiking traders are introduced to the agent-based simulation model. In our experiments, N_{ST} has a value of 2.

5.5.2 Mini Flash Crash Analysis

To investigate mini flash crash events, there are some modifications to the simulation configuration. The number of market makers in the simulation is reduced. Our experiments show that this increases the probability of the occurrence of mini flash crash events. The reason is that fewer market makers will lead to thinner market depth, generating more mini flash crash events for scrutiny. The institutional trader is removed from the market; while two spiking traders are introduced to the market simulation. Other model parameters are the same as the simulation for the 2010 Flash Crash. Detailed model parameters for mini flash crash simulation are shown in Appendix D.

Figure 5.10 presents a typical mini flash crash scenario in our simulation. The price drops nearly 80 basis points in just several seconds, and then bounces back towards the fundamental value. Figure 5.11 shows the inventory level for each type of trader against simulated price during the mini flash crash scenario. According to Figure 5.11, the trigger and dynamics for that specific simulated mini flash crash are very straightforward:

- At 12:08:38, a spiking trader is activated to bring downward shocks to the market. The spiking trader submits sell orders to the market in the next 2 seconds. The sell pressure in the 2-second interval is mainly absorbed by market makers.
- Having absorbed the sell orders from spiking traders, one market maker accumulates a relatively large inventory at 12:08:40 and the inventory limit is reached. For risk management purposes, the market maker decides to reduce the position and then temporarily withdraw from the market. The sell orders from this market maker are digested by other market makers.
- The same high-frequency dynamic happens between 12:08:40 and 12:08:42. During this small period of time, the same positions transfer between different market makers, creating the "hot-potato" effect on a smaller scale. Several market makers withdraw from the market temporarily after emptying their inventory.
- At 12:08:42, only 4 seconds after the spiking trader is activated, the market suffers from liquidity loss because of the withdrawal of some market makers. The sell orders from the remaining market makers create dramatic market impacts due to thin liquidity, dragging the price down for more than 60 basis points in 5 seconds.
- At 12:08:47, the price has dropped 80 basis points compared to the price level at 12:08:38. The price then stops dropping and gradually bounces back to the original level.

Note that in the above simulation, the fundamental traders are configured to have a trading interval of 100 steps, which is the calibrated value from Section 4. The specific simulated mini flash crash event happens in a time horizon that is shorter than the horizon over which lower frequency fundamental traders observe the market. This is shown in Figure 5.11, where the inventory for fundamental traders hardly changed during the mini flash crash event. The analysis shows that the mini flash crash events result from the procyclical behaviours among high-frequency market makers, precipitated by price shocks that are created by spiking traders or other market participants.

5.5.3 Conditions for Mini Flash Crash Scenarios

In previous sections, we explore how different conditions influence the amplitude of large flash crash events. Here we address the question of how those conditions influence the characteristics of mini flash crash events. Since the institutional trader is removed from the simulation, two conditions that are



Figure 5.10 One typical simulated mini flash crash event. Red line is the historical price; green line is the simulated price; blue line is the fundamental value

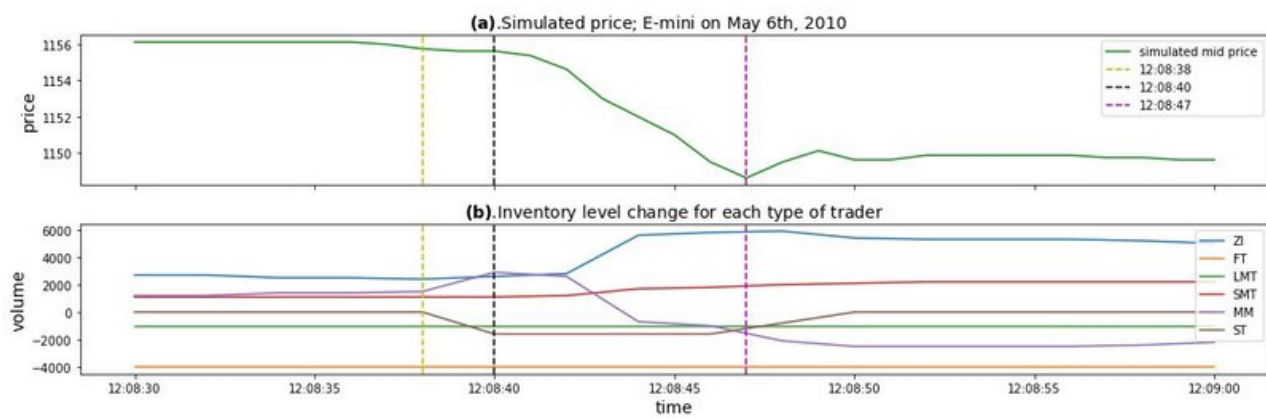


Figure 5.11 Detailed simulated price against the inventories of each type of trader during the simulated mini flash crash event. (a): Simulated and historical price from 12:08:30 to 12:09:00. (b): Total inventory level for each type of trader during period of time from 12:08:30 to 12:09:00

unrelated to the institutional trader are considered: the inventory limit of market makers and the trading frequency of fundamental traders. Unlike the colossal flash crashes that happen extremely rarely, mini flash crashes occur more frequently in modern financial markets where high-frequency trading contributes to a vast portion of transactions. Consequently, the characteristics of mini flash crash events are twofold: the frequency of the occurrence of mini flash crash events and the crash severity once a mini flash crash occurs.

To explore the characteristics of mini flash crashes under different conditions, a Monte Carlo simulation experiment is carried out. For the purpose of consistency, the simulation also mimics the E-mini futures market on May 6th. The simulation spans from 8:00 to 12:30 on May 6th, 2010, excluding the afternoon trading session to reduce the influence of the large historical flash crash event. Except for the two model parameters that represent the two conditions that are of interest, other model parameters are assigned

the same value as the calibration results and are strictly kept fixed¹³. For each parameter combination, which represents a specific condition, we run 60 simulations and calculate the average frequency for the occurrence of mini flash crash events and the average amplitude for the mini flash crash events.

The remaining questions to solve are how to count the mini flash crashes and how to measure the amplitude of a mini flash crash. Following [KNWB18], a mini flash crash is classified as a k standard deviation move in price, which reverses over a horizon that is less than certain periods of time. The k has values 2, 3, and 4 in our experiments. Note that because the spiking trader can create both upward and downward price shock, we also consider the upward "flare up" event as another form of mini flash crash¹⁴. Specifically, in our experiments, a k -sd mini flash crash is defined to be the peak or trough price behaviour inside a 10-minute interval, where the price moves more than k standard deviations and then bounces back. The k standard deviation is calculated by reference to the minute-level return. Inspired by topography, the amplitude of the mini flash crash event is then defined to be the prominence of the peak or trough price trajectory. A specific method to calculate the prominence can be found in [VGO⁺20].

We calculate the average number of occurrences of mini flash crash events and the average amplitude for the mini flash crash events under different conditions. The results are shown in Figure 5.12 and Figure 5.13.

Figure 5.12 presents how the average number of mini flash crashes per simulation changes when the inventory limit of market makers and trading frequency of fundamental traders vary. As shown in panel (a), increasing the inventory limit will decrease the average number of mini flash crashes in our simulation, indicating a lower frequency for the occurrence of mini flash crash events. This result is consistent with our intuition. The larger the inventory limit for market makers, the more selling pressure market makers are able to absorb. Thus the "hot-potato" effect is less likely to appear with a larger inventory limit for market makers, resulting in fewer mini flash crashes. As for the trading frequency of fundamental traders, however, the frequency of the occurrence of mini flash crash events barely changes when we vary the trading frequency parameter of fundamental traders. This is presented in panel (b) of Figure 5.12. One possible explanation is that mini flash crash events happen in an extremely short time scale and the amplitude is relatively small. Consequently, fundamental traders

¹³Due to no calibration results, the model parameters for the newly introduced spiking trader are given values heuristically.

¹⁴The definition for "flare up" and "flash crash" are symmetrical: "flare up" is a rapid price rise followed by a price drop.

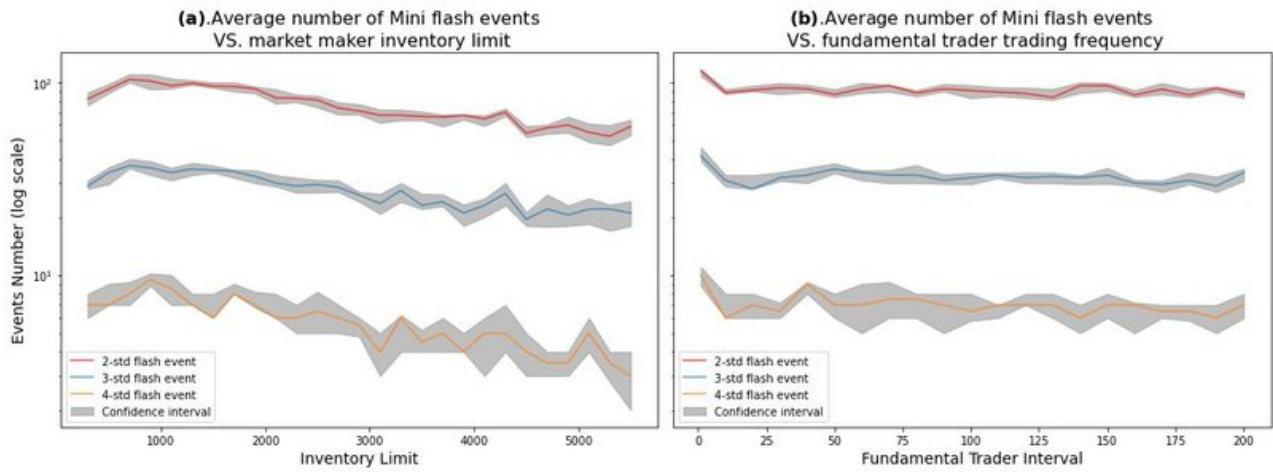


Figure 5.12 (a): Relationship between the average number of mini flash crashes per simulation and the inventory limit of market maker. **(b):** Relationship between the average number of mini flash crashes per simulation and the trading frequency of fundamental traders.

are hardly able to nor willing to participate during the course of the mini flash crash events.

Figure 5.13 presents how the average amplitude of mini flash crashes changes when the inventory limit of market makers and trading frequency of fundamental traders vary. As demonstrated by Panel (a) in Figure 5.13, the functional relationship between mini flash crash amplitude and market maker inventory limit is not monotonous. The average mini flash crash amplitude is an increasing function with regard to the market maker inventory limit when the inventory limit is small, while it turns into a decreasing function when the market maker inventory limit is large enough. The logic here is very similar to the previous analysis. While the inventory limit of market maker is small, increasing the inventory limit will result in more selling pressure when the inventory limit is hit, leading to a larger crash amplitude. However, a large enough market maker inventory limit would absorb most shocks in the market, restricting the scale of the mini flash crash events. Moving on to panel (b) in Figure 5.13, we observe no obvious influence of fundamental traders' trading frequency on the amplitude of mini flash crash events. This phenomenon once again demonstrates that fundamental traders hardly participate in the course of mini flash crash events.

5.5.4 Comparison Between Large and Mini Flash Crash Events

Generally speaking, mini flash crash scenarios can be viewed as scaled-down versions of larger flash crash events, with the latter occurring less frequently than the former. In our simulation, both scenarios share a similar underlying mechanism. In both cases, the initiation involves relatively large orders,

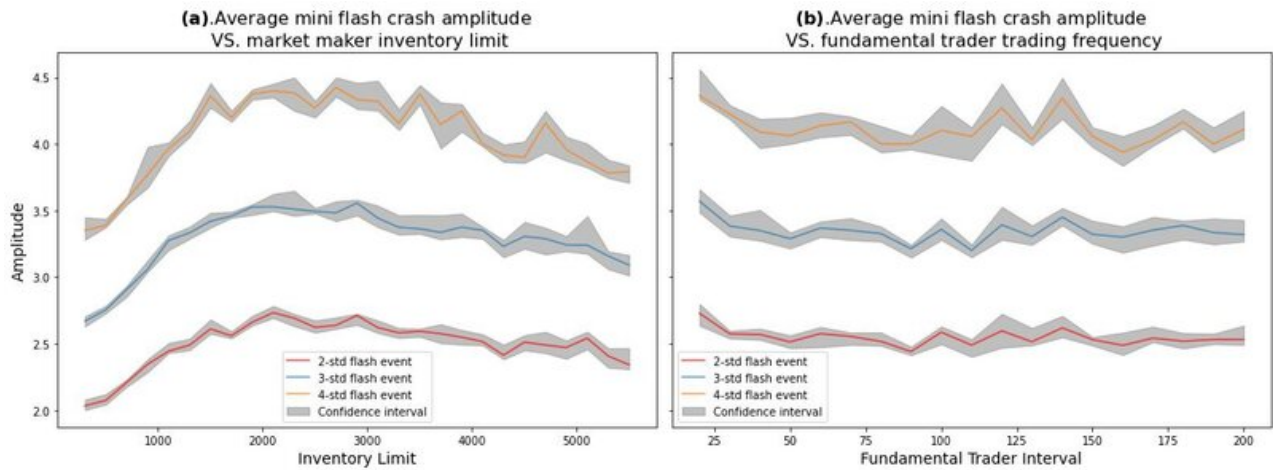


Figure 5.13 (a): Relationship between the average amplitude of simulated mini flash crashes and the inventory limit of market maker. **(b):** Relationship between the average amplitude of simulated mini flash crashes and the trading frequency of fundamental traders.

followed by exacerbation from high-frequency market makers.

The primary distinction between a large flash crash event and mini flash crash scenarios lies in the scale and magnitude of the market disruption. A "large flash crash event" typically refers to a significant and abrupt decline in asset prices across various markets, often involving major financial instruments. These events are characterised by a rapid and substantial market downturn, resulting in a sharp drop in prices and high volatility. Large flash crashes can have widespread implications for the financial system and are often associated with systemic risks.

On the other hand, "mini flash crash scenarios" refer to smaller-scale disruptions that share similarities with large flash crashes but are less extensive in their impact. These events may involve specific stocks, sectors, or asset classes rather than affecting the entire market. While still characterised by sudden and sharp price movements, mini flash crashes tend to have a more localised effect and may not pose the same level of systemic risk as their larger counterparts.

5.6 Summary

A novel high-frequency agent-based financial market simulator is implemented to generate a realistic high-frequency simulated financial market. Each simulation step corresponds to 100 milliseconds in the real-world trading environment. Full exchange protocols (limit order books) are implemented to simulate the order matching process. In this way, we provide a microstructure model of a single security traded on a central limit order book in which market participants follow fixed behavioural rules. The

model is calibrated using the machine learning surrogate modelling approach. Statistical test and moment coverage ratio results show that the simulation is capable of reproducing realistic stylised facts in financial markets.

The simulator is then employed to explore the dynamics during flash crash episodes and the conditions that affect flash crash episodes. Under the framework of the proposed high-frequency agent-based financial market simulator, the 2010 Flash Crash is realistically simulated by introducing an institutional trader that mimics the real-world Sell Algorithm on May 6th, 2010. We investigate the market dynamics during the simulated flash crash and show that the simulated dynamics are consistent with what happened in historical flash crash scenarios. We then explore the conditions that could influence the characteristics of the 2010 Flash Crash. It is found that three conditions significantly affected the amplitude of the 2010 Flash Crash: the percentage of volume of the Sell Algorithm, the market maker inventory limit, and the trading frequency of fundamental traders. In particular, we found that the relationship between the amplitude of the simulated 2010 Flash Crash and the POV of the Sell Algorithm is not monotonous, and so is the relationship between the amplitude and the market maker inventory limit. For the trading frequency of fundamental traders, the higher the frequency, the smaller the amplitude of the simulated 2010 Flash Crash.

A similar analysis is carried out for mini flash crash events. An innovative type of trader called "Spiking Trader" is introduced to the agent-based financial market simulator, creating more price shock and precipitating more mini flash crash events. Market dynamics for a typical simulated mini flash crash event are analysed. We also explore the conditions that could influence the characteristics of mini flash crash events. Experimental results show that market maker inventory limit significantly affects both the frequency and amplitude of mini flash crash events. However, the trading frequency of fundamental traders shows no obvious influence on mini flash crash events in our experiments.

Chapter 6

Conclusion

6.1 Summary of Thesis Achievements

This thesis explores the modelling of financial markets through an agent-based approach. It covers the design, calibration, and validation of various agent-based models aimed at replicating realistic financial time series. Our research contributes significantly to three key areas using this methodology: (a) We refine the calibration of agent-based models and provide a detailed analysis of the intra-day price formation process. (b) We combine agent-based financial market simulation with deep reinforcement learning algorithms to optimise hedging strategies in financial markets. (c) We develop a high-frequency financial market simulator and conduct comprehensive studies on flash crash scenarios. The following elaborates on these contributions in more detail.

Chapter 3 (Contribution 1) introduces the XGB-Chiarella method, an innovative adaptation of the Extended Chiarella model to generate minute-level intra-day financial price data. The approach employs the XGBoost machine learning algorithm for efficient calibration of the model. An innovative exploration-exploitation mechanism is introduced in this process. This approach, tested on 75 stocks across NASDAQ, LSE, and HKEX, uses a simple model structure with three agent types—fundamental, momentum, and noise traders. Compared to the Expectation-Maximisation baseline, the XGB-Chiarella method reduces stylised facts distance by 10%, demonstrating superior performance in replicating empirical stylised facts. The proposed approach also demonstrates that the agent choice in the model effectively captures the intra-day price formation process. With prediction errors under 10%, the surrogate model validates the efficiency of the calibration process, which intelligently explores the

parameter space. Furthermore, this approach highlights the crucial roles of trend and value effects, as well as noise trading, in intra-day price dynamics. This suggests a potentially universal mechanism for intra-day price formation and illustrates the practical applications of this methodology in financial market simulations.

Chapter 4 (Contribution 2) introduces the Chiarella-Heston model, an innovative agent-based approach designed to enhance deep hedging strategies in quantitative finance. The model integrates momentum, fundamental, and volatility traders, drawing on elements from both the Extended Chiarella model and the Heston stochastic volatility model to closely mimic empirical market behaviours like fat tails and autocorrelation patterns of returns. Compared to baseline models, the Chiarella-Heston model reduces the stylised facts distance by more than 50%, demonstrating its ability to generate more realistic financial time series. The model is validated using the Generalized Subtracted L-divergence metric. Data generated from the Chiarella-Heston model is utilised to train a deep hedging agent with an improved version of the Deep Deterministic Policy Gradient algorithm, aiming to optimise hedging strategies across varying transaction cost levels. Empirical tests demonstrate that the reinforcement learning agent, trained on data from the Chiarella-Heston model, outperforms the agent trained on traditional models like Geometric Brownian Motion. In terms of risk management, the proposed approach reduces the average expected shortfall from -3.93% to -2.74%, representing a 30% improvement. In particular, our testing experiment is carried out in real-world time series data, showcasing the application of the proposed method in practical trading environments. This methodology provides robust tools for risk management and optimal hedging in financial markets.

Chapter 5 (Contribution 3) presents a novel high-frequency agent-based financial market simulator that addresses challenges in replicating realistic high-frequency financial market dynamics and analysing flash crash events. The simulator operates at sub-second intervals to mirror high-frequency trading. The simulator incorporates full exchange protocols and uses a machine learning surrogate modelling approach for calibration, successfully reproducing various stylised facts in financial markets. We successfully simulate the 2010 Flash Crash and present a detailed analysis of high-frequency market dynamics during the flash crash event. This simulation revealed that the percentage of volume of the Sell Algorithm, market maker inventory limits, and the trading frequency of fundamental traders significantly influence the amplitude of flash crash scenarios. Additionally, the introduction of a "Spiking Trader" in the simulator allows for the exploration of mini flash crash dynamics. Experiments show that the market maker inventory limit affects both the frequency and amplitude of these events, whereas the trading

frequency of fundamental traders does not significantly impact mini flash crashes. These insights underline the simulator’s capability to provide valuable analysis of the conditions affecting major and minor market disruptions. The proposed simulator offers a robust tool for understanding risk scenarios and potentially mitigating such risk events in real-world scenarios.

6.2 Future Work

Agent-based modelling is a promising approach that has many applications in financial services. In this thesis, we propose various agent-based models for simulating financial markets. We address three practical challenges, focusing on risk management practices. There are several areas where further research could be conducted to improve the performance of our proposed models and advance the field of agent-based modelling for finance. Some are short-term improvements while others are part of a long-term plan that we expect to explore in the future.

6.2.1 Robust calibration and validation techniques

Future research in the calibration and validation of agent-based models should prioritise improving the robustness and applicability of these models in financial analysis. The promising results of the XGB-Chiarella method highlight the potential benefits of integrating advanced machine learning algorithms to refine the calibration process. This suggests a shift towards utilising newer, more capable algorithms that are suitable for handling complex calibrations beyond the current mainstream Method of Simulated Moments (MSM). Additionally, there is a critical need to address and mitigate biases inherent in MSM, with an emphasis on conducting extensive computational experiments to better evaluate their effectiveness and limitations. Following the foundational work of [GRT17], one interesting direction is exploring ways to incorporate temporal dependencies into the calibration process. In this way, the dynamics of financial markets will be replicated more accurately. This comprehensive approach will significantly enhance the precision and utility of agent-based models in financial services.

The validation methods used in this thesis for agent-based models primarily involve statistical tests and moment coverage ratios. Future work could focus on enhancing these validation methods by leveraging advancements in machine learning and big data. The integration of machine learning offers a more flexible approach, reducing the number of assumptions needed in econometric models. Additionally,

utilising big data enables more detailed comparisons of models with real-world scenarios, especially at the micro-level. These improvements could significantly refine ABM validation techniques, helping researchers determine more effectively whether a model inaccurately represents the underlying data. This approach will improve the applicability of ABMs in analysing financial markets.

6.2.2 More effective deep hedging strategy

A possible next step in optimising deep hedging strategies involves a deeper integration of real-time data analytics and the refinement of existing neural network architectures. Future research could explore the use of real-time data feeds to dynamically adjust hedging positions based on current market conditions, potentially integrating elements of unsupervised learning to detect new market patterns as they emerge. This approach aims to make deep hedging more responsive and adaptive to sudden market changes, thereby increasing the robustness of hedging strategies.

Additionally, experimenting with different neural network structures, such as recurrent neural networks (RNNs) or transformers, may yield improvements in predicting risks associated with volatile market conditions. Transformers, in particular, excel at identifying patterns in historical time series data, making them valuable for forecasting future risk events. Accurate prediction could lead to the development of more adaptive and resilient hedging strategies that maintain efficacy across diverse market environments. Enhancing these computational models with advanced machine learning and deep learning techniques could significantly improve the effectiveness of hedging practices.

6.2.3 Prediction of flash crash events

In this thesis, the market dynamics during flash crash events are scrutinised, raising the question of whether it is possible to predict the occurrence of such events. Existing research, such as [PHSB17], suggests that stability indicators derived from limit order book information can signal a high likelihood of an imminent flash crash. Based on our proposed model, one promising direction is to explore whether there are specific patterns in the simulated limit order books that could indicate an imminent flash crash. As an example, [PHSB17] demonstrates that the Volume-Synchronized Probability of Informed Trading (VPIN) measure [EdPO19] provides a predictive signal for the 2010 Flash Crash. We apply the VPIN measure in our simulation of the 2010 Flash Crash, as depicted in Figure 6.1. Consistent with [PHSB17], there is a significant surge in the VPIN measure between 13:30 and 14:30, and the

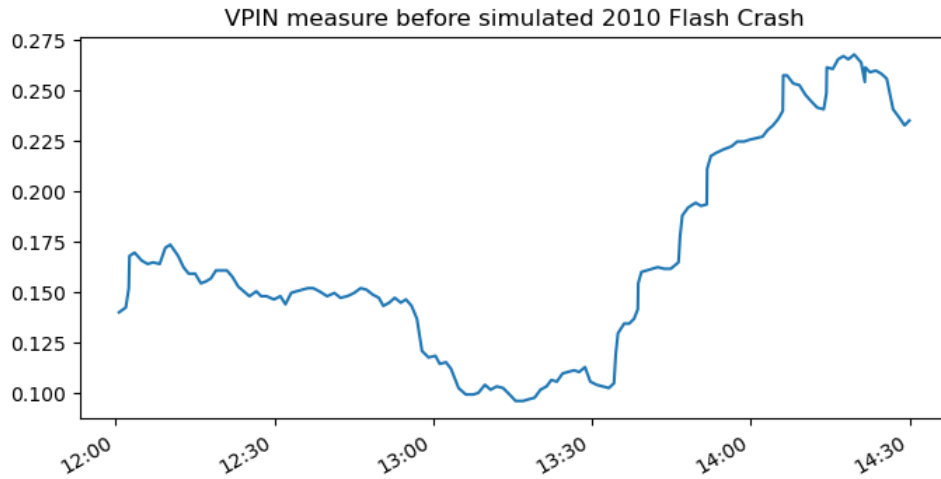


Figure 6.1 The VPIN measure before the **simulated** 2010 Flash Crash

measure reaches a peak just before the simulated flash crash. at a very high level just before the happening of the simulated flash crash event. This further validates the realism of our simulation and indicates that the simulator can be utilised to identify early warning signs of flash crashes or other extreme market events. Exploration of these emerging patterns could enhance our understanding of the mechanisms underlying flash crashes. Such insights could be instrumental in developing pre-emptive risk management strategies.

In addition, future studies could develop predictive analytics tools that utilise machine learning to analyse patterns in simulated limit order books. For example, [ZZR19] proposes DeepLOB, which is a large-scale deep learning model aimed at predicting price movements from limit order book data. By integrating these machine learning methods with real-time market data, researchers could refine predictive models to offer timely warnings about potential market disruptions. For instance, [WCGW24] designs a risk monitoring system based on real-time data, which utilises machine learning algorithms including the Long Short-Term Memory (LSTM) networks, Random Forest and Gradient Boosting Trees. Such systems enable market participants to take precautionary measures and implement downside protections against potential risk scenarios.

6.2.4 Multi-agent reinforcement learning and multi-agent simulation

Multi-agent reinforcement learning (MARL) examines the interactions and behaviours of multiple learning agents within a shared environment, where each agent independently pursues its own rewards, often resulting in complex group dynamics due to conflicting interests. MARL represents a natural

evolution from agent-based models and multi-agent simulations, particularly in real-world dynamic systems where participants continuously learn and adapt to changes. Multiple intelligent agents are essential to effectively simulate such a system. For example, there is an interesting framework called "Phantom" [AVG⁺23] that combines MARL with multi-agent simulation to model these complex systems.

While MARL is beyond the primary scope of this thesis, the research presented here suggests several potential avenues for future exploration. In this thesis, the models primarily consist of agents following fixed rules, with their behaviours adapting accordingly. In terms of intelligent agents, the models in this thesis only allow one intelligent agent such as the deep reinforcement learning agent in the Chiarella-Heston framework in Chapter 4. One promising direction for further research involves investigating the interactions between multiple deep hedging agents within the Chiarella-Heston framework. It is well-established that the proliferation of identical trading strategies can lead to performance degradation; thus, there is massive research interest in exploring whether multiple deep hedging agents would cooperate or compete for profits when placed in the same system. Additionally, examining whether interactions between intelligent agents might exacerbate risk scenarios, such as flash crashes, presents another compelling research direction.

6.3 Final Thoughts

The exploration undertaken in this thesis represents a significant step in modelling financial markets through an agent-based approach. The proposed models elucidate both the complexities and the dynamic nature of financial systems. The development of the XGB-Chiarella method, the enhancement of deep hedging strategies using the Chiarella-Heston model, and the insightful simulation of flash crash scenarios exemplify the robust application of agent-based models in financial services.

Reflecting on the achievements, it is evident that agent-based models have substantial utility in financial analysis, providing insights into the price formation process and offering robust tools for risk management. These models serve as powerful instruments to interpret the often turbulent financial markets. The application of sophisticated machine learning techniques to refine these models highlights a pivotal shift towards more predictive and responsive financial modelling, essential for both understanding and anticipating market movements.

Looking ahead, the research on multi-agent simulation systems evolves naturally toward multi-agent reinforcement learning (MARL) due to its capacity to enable agents to learn and adapt dynamically within complex environments. Agent-based models traditionally rely on predefined rules and behaviours, which can limit their ability to respond to novel scenarios. MARL, on the other hand, allows agents to learn optimal strategies through interactions, which makes the models more flexible and capable of handling complex market conditions. For example, [HZCL24] use MARL to simulate trading strategies that adapt to changes in volatility or liquidity, thereby enhancing the robustness of such trading strategies. Such integration of MARL can extend the capabilities of ABMs by incorporating more realistic behavioural patterns, as agents learn not only from their own experiences but also from the actions and feedback of other agents in the system. This progression will require a continuous interplay between theoretical innovation and empirical validation, ensuring that these models remain both relevant and effective in a rapidly changing financial landscape.

The journey of this thesis has established a solid foundation for further research and innovation in financial modelling. As agent-based models continue to develop, they offer not only a deeper comprehension of market dynamics but also equip financial practitioners with more accurate simulation, forecasting and risk management. The long-term aim of this work is to develop a dynamic, intelligent system capable of simulating, analysing, and monitoring financial markets. One example of a similar system is the Nasdaq SMARTS Market Surveillance system¹. This system is a comprehensive suite of tools designed for monitoring and analysing trading activity across financial markets, used by regulators, exchanges, brokers, and other market participants. The contributions of this thesis are able to supplement the functionality and enhance the intelligence of such systems, advancing their ability to adapt to the evolving complexities of financial markets.

¹<https://www.nasdaq.com/solutions/fintech/nasdaq-trade-surveillance>

Bibliography

- [AGL17] Eric Aldrich, Joseph Grundfest, and Gregory Laughlin. The flash crash: A new deconstruction. *Available at SSRN 2721922*, 2017.
- [AVG⁺23] Leo Ardon, Jared Vann, Deepeka Garg, Thomas Spooner, and Sumitra Ganesh. Phantom - a RL-driven multi-agent framework to model complex systems. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS '23*, page 2742–2744, Richland, SC, 2023. International Foundation for Autonomous Agents and Multiagent Systems.
- [Bar19] Söhnke M Bartram. Corporate hedging and speculation with derivatives. *Journal of Corporate Finance*, 57:9–34, 2019.
- [BCGV07] Carlo Bianchi, Pasquale Cirillo, Mauro Gallegati, and Pietro Vagliasindi. Validating and calibrating agent-based models: A case study. *Computational Economics*, 30(3):245–264, 2007.
- [BCP13] Paul Brewer, Jaksa Cvitanic, and Charles R Plott. Market microstructure design and flash crashes: A simulation approach. *Journal of Applied Economics*, 16(2):223–250, 2013.
- [BDSY10] Milan Borkovec, Ian Domowitz, Vitaly Serbin, and Henry Yegerman. Liquidity and price discovery in exchange-traded funds: One of several possible lessons from the flash crash. *The Journal of Index Investing*, 1(2):24–42, 2010.
- [BEG12] Philip Bond, Alex Edmans, and Itay Goldstein. The real effects of financial markets. *Annual Review of Financial Economics*, 4(1):339–360, 2012.
- [BG80] Avraham Beja and Barry Goldman. On the dynamic behavior of prices in disequilibrium. *The Journal of Finance*, 35(2):235–248, 1980.

- [BGT⁺20] Hans Buehler, Lukas Gonon, Josef Teichmann, Ben Wood, Baranidharan Mohan, and Jonathan Kochems. Deep hedging: Hedging derivatives under generic market frictions using reinforcement learning. *Available at SSRN*, 2020.
- [BS73] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.
- [BSS04] Yu Byron, Krishna Shenoy, and Maneesh Sahani. Derivation of kalman filtering and smoothing equations. In *Technical report*. Stanford University, 2004.
- [CCHP21] Jay Cao, Jacky Chen, John Hull, and Zissis Poulos. Deep hedging of derivatives using reinforcement learning. *arXiv preprint arXiv:2103.16409*, 2021.
- [CF96] Timothy N Cason and Daniel Friedman. Price formation in double auction markets. *Journal of Economic Dynamics and Control*, 20(8):1307–1337, 1996.
- [CG16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.
- [CH11] Andrew T Crooks and Alison J Heppenstall. Introduction to agent-based modelling. In *Agent-based Models of Geographical Systems*, pages 85–105. Springer, 2011.
- [Chi92] Carl Chiarella. The dynamics of speculative behaviour. *Annals of Operations Research*, 37(1):101–123, 1992.
- [CJW20] Álvaro Cartea, Sebastian Jaimungal, and Yixuan Wang. Spoofing and price manipulation in order-driven markets. *Applied Mathematical Finance*, 27(1-2):67–98, 2020.
- [Con01] Rama Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 1(2):223, 2001.
- [Con07] Rama Cont. Volatility clustering in financial markets: Empirical facts and agent-based models. In *Long Memory in Economics*, pages 289–309. Springer, 2007.
- [CT11] Bidisha Chakrabarty and Konstantin Tyurin. Market liquidity, stock characteristics and order cancellations: The case of fleeting orders. In *Financial Econometrics Modelling: Market Microstructure, Factor Models and Financial Risk Measures*, pages 33–65. Springer, 2011.

- [DO07] Vincent Darley and Alexander Outkin. *A NASDAQ market simulation: insights on a major market from the science of complex adaptive systems*, volume 1. World Scientific, 2007.
- [DPV18] Giovanni Dosi, Marcelo Pereira, and Maria Enrica Virgillito. On the robustness of the fat-tailed distribution of firm growth rates: A global sensitivity analysis. *Journal of Economic Interaction and Coordination*, 13(1):173–193, 2018.
- [EdPO19] David Easley, M López de Prado, and Maureen O’Hara. *The exchange of flow toxicity*. SSRN, 2019.
- [FF86] Jeffrey Frankel and Kenneth Froot. Understanding the us dollar in the eighties: The expectations of chartists and fundamentalists. *Economic Record*, 62(1):24–38, 1986.
- [FF12] Artur J Ferreira and Mário AT Figueiredo. Boosting algorithms: A review of methods, theory, and applications. *Ensemble Machine Learning: Methods and Applications*, pages 35–85, 2012.
- [FGG21] Merritt B Fox, Lawrence R Glosten, and Sue S Guan. Spoofing and its regulation. *Colum. Bus. L. Rev.*, page 1244, 2021.
- [FHBMG11] Marta Faias, Carlos Hervés-Beloso, and Emma Moreno-García. Equilibrium price formation in markets with differentially informed agents. *Economic Theory*, 48(1):205–218, 2011.
- [FMS95] Jeffrey C Fuhrer, George R Moore, and Scott D Schuh. Estimating the linear-quadratic inventory model maximum likelihood versus generalized method of moments. *Journal of Monetary Economics*, 35(1):115–157, 1995.
- [Fra09] Reiner Franke. Applying the method of simulated moments to estimate a small agent-based asset pricing model. *Journal of Empirical Finance*, 16(5):804–815, 2009.
- [FW12] Reiner Franke and Frank Westerhoff. Structural stochastic volatility in asset pricing dynamics: Estimation and model contest. *Journal of Economic Dynamics and Control*, 36(8):1193–1211, 2012.
- [GKP12] Anton Golub, John Keane, and SerHuang Poon. High frequency trading and mini flash crashes. *Available at SSRN 2182097*, 2012.

- [GM94] Mason S Gerety and J Harold Mulherin. Price formation on stock exchanges: The evolution of trading within the day. *Review of Financial Studies*, 7(3):609–629, 1994.
- [GPW⁺13] Martin D Gould, Mason A Porter, Stacy Williams, Mark McDonald, Daniel J Fenn, and Sam D Howison. Limit order books. *Quantitative Finance*, 13(11):1709–1742, 2013.
- [GRT17] Jakob Grazzini, Matteo G Richiardi, and Mike Tsionas. Bayesian estimation of agent-based models. *Journal of Economic Dynamics and Control*, 77:26–47, 2017.
- [GS93] Dhananjay K. Gode and Shyam Sunder. Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *Journal of Political Economy*, 101(1):119–137, 1993.
- [GS97] Jeremy Greenwood and Bruce D Smith. Financial markets in development, and the development of financial markets. *Journal of Economic Dynamics and Control*, 21(1):145–181, 1997.
- [GVL⁺22] Kang Gao, Perukrishnen Vytelingum, Wayne Luk, Stephen Weston, and Ce Guo. Understanding intra-day price formation process by agent-based financial market simulation: Calibrating the extended chiarella model. *Wilmott*, 2022(119):22–38, 2022.
- [GVW⁺24] Kang Gao, Perukrishnen Vytelingum, Stephen Weston, Wayne Luk, and Ce Guo. High-frequency financial market simulation and flash crash scenarios analysis: An agent-based modelling approach. *Journal of Artificial Societies and Social Simulation*, 27(2):8, 2024.
- [HB16] John C Hull and Sankarshan Basu. *Options, futures, and other derivatives*. Pearson Education, 2016.
- [Hes93] Steven Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6(2):327–343, 1993.
- [HHLS15] Nicolas Heess, Jonathan J Hunt, Timothy P Lillicrap, and David Silver. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455*, 2015.
- [Hil10] Jonathan B Hill. On tail index estimation for dependent, heterogeneous data. *Econometric Theory*, 26(5):1398–1436, 2010.

- [HLJB19] Buehler Hans, Gonon Lukas, Teichmann Josef, and Wood Ben. Deep hedging. *Quantitative Finance*, 19(8):1271–1291, 2019.
- [HTC⁺14] Roy Hayes, Andrew Todd, Nachapon Chaidarun, Scott Tepsuporn, Peter Beling, and William Scherer. An agent-based financial simulation for use by researchers. In *Proceedings of the Winter Simulation Conference 2014*, pages 300–309, 2014.
- [HTZ21] Blanka Horvath, Josef Teichmann, and Zan Zuric. Deep hedging under rough volatility. *Risks*, 9(7), 2021.
- [HZCL24] Yuling Huang, Chujin Zhou, Kai Cui, and Xiaoping Lu. A multi-agent reinforcement learning framework for optimizing financial trading strategies based on timesnet. *Expert Systems with Applications*, 237:121502, 2024.
- [Jac91] Matthew O Jackson. Equilibrium, price formation, and the value of private information. *The Review of Financial Studies*, 4(1):1–16, 1991.
- [Jan05] Marco A Janssen. Agent-based modelling. *Modelling in Ecological Economics*, 155(1):172–181, 2005.
- [Jan18] Zuzana Janková. Drawbacks and limitations of black-scholes model for options pricing. *Journal of Financial Studies and Research*, 2018:1–7, 2018.
- [JLNRF16] Sandrine Jacob Leal, Mauro Napoletano, Andrea Roventini, and Giorgio Fagiolo. Rock around the clock: An agent-based model of low-and high-frequency trading. *Journal of Evolutionary Economics*, 26(1):49–76, 2016.
- [JM15] Grazzini Jakob and Richiardi Matteo. Estimation of ergodic agent-based models by simulated minimum distance. *Journal of Economic Dynamics and Control*, 51:148–165, 2015.
- [JZH⁺12] Neil Johnson, Guannan Zhao, Eric Hunsader, Jing Meng, Amith Ravindar, Spencer Carran, and Brian Tivnan. Financial black swans driven by ultrafast machine ecology. *arXiv preprint arXiv:1202.1448*, 2012.
- [KAS15] Sergei Kucherenko, Daniel Albrecht, and Andrea Saltelli. Exploring multi-dimensional spaces: A comparison of latin hypercube and quasi monte carlo sampling techniques. *arXiv preprint arXiv:1505.02350*, 2015.

- [KKST17] Andrei Kirilenko, Albert S Kyle, Mehrdad Samadi, and Tugkan Tuzun. The flash crash: High-frequency trading in an electronic market. *The Journal of Finance*, 72(3):967–998, 2017.
- [KLM96] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.
- [KNWB18] Geir Karvik, Joseph Noss, Jack Worlidge, and Daniel Beale. The deeds of speed: an agent-based model of market liquidity and flash episodes. *Bank of England Working Paper*, 2018.
- [KO20] Albert S Kyle and Anna Obižaeva. *Large bets and stock market crashes*. CEFIR, 2020.
- [Kyl85] Albert Kyle. Continuous auctions and insider trading. *Econometrica: Journal of the Econometric Society*, pages 1315–1335, 1985.
- [Lam18] Francesco Lamperti. An information theoretic criterion for empirical validation of simulation models. *Econometrics and Statistics*, 5:83–106, 2018.
- [LeB01] Blake LeBaron. A builder’s guide to agent-based financial markets. *Quantitative finance*, 1:254–261, 2001.
- [LeB06] Blake LeBaron. Agent-based computational finance. *Handbook of Computational Economics*, 2:1187–1233, 2006.
- [LHP⁺15] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- [LLLL16] Aimé Lachapelle, Jean-Michel Lasry, Charles-Albert Lehalle, and Pierre-Louis Lions. Efficiency of the price formation process in presence of high frequency participants: a mean field game analysis. *Mathematics and Financial Economics*, 10:223–262, 2016.
- [LN19] Sandrine Jacob Leal and Mauro Napoletano. Market stability vs. market resilience: Regulatory policies experiments in an agent-based model with low- and high-frequency trading. *Journal of Economic Behavior and Organization*, 157:15–41, 2019.
- [Lo17] Andrew W. Lo. *Adaptive Markets: Financial Evolution at the Speed of Thought*. Princeton University Press, 2017.

- [LRS18] Francesco Lamperti, Andrea Roventini, and Amir Sani. Agent-based model calibration using machine learning surrogates. *Journal of Economic Dynamics and Control*, 90:366–389, 2018.
- [Mad12] Ananth Madhavan. Exchange-traded funds, market structure, and the flash crash. *Financial Analysts Journal*, 68(4):20–35, 2012.
- [MBGC19] Frank McGroarty, Ash Booth, Enrico Gerding, and VL Raju Chinthalapati. High frequency trading strategies, market fragility and price spikes: An agent based model perspective. *Annals of Operations Research*, 282(1):217–244, 2019.
- [MBM⁺16] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937. PMLR, 2016.
- [MC94] William J Morokoff and Russel E Caflisch. Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, 15(6):1251–1279, 1994.
- [MCB20] Adam Majewski, Stefano Ciliberti, and Jean-Philippe Bouchaud. Co-existence of trend and value in financial markets: Estimating an extended chiara model. *Journal of Economic Dynamics and Control*, 112:103791, 2020.
- [Men13] Albert Menkveld. High frequency trading and the new market makers. *Journal of financial Markets*, 16(4):712–740, 2013.
- [MKS⁺13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.
- [MKS⁺15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.
- [MM61] Merton H Miller and Franco Modigliani. Dividend policy, growth, and the valuation of shares. *Journal of Business*, 34(4):411–433, 1961.

- [MY19] Albert Menkveld and Bart Zhou Yueshen. The flash crash: A cautionary tale about highly fragmented markets. *Management Science*, 65(10):4470–4488, 2019.
- [NMTY11] Giuseppe Nuti, Mahnoosh Mirghaemi, Philip Treleaven, and Chaiyakorn Yingsaeree. Algorithmic trading. *Computer*, 44(11):61–69, 2011.
- [O’H95] Maureen O’Hara. *Market Microstructure Theory*. Wiley, 1995.
- [PAD22] Fernando de Meer Pardo, Christoph Auth, and Florin Dascalu. A modular framework for reinforcement learning optimal execution. *arXiv preprint arXiv:2208.06244*, 2022.
- [PCW19] James Paulin, Anisoara Calinescu, and Michael Wooldridge. Understanding flash crash contagion and systemic risk: A micro–macro agent-based approach. *Journal of Economic Dynamics and Control*, 100:200–229, 2019.
- [PHSB17] Mark Paddrik, Roy Hayes, William Scherer, and Peter Beling. Effects of limit order book information level on market stability metrics. *Journal of Economic Interaction and Coordination*, 12(2):221–247, 2017.
- [PHT⁺12] Mark Paddrik, Roy Hayes, Andrew Todd, Steve Yang, Peter Beling, and William Scherer. 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 and Economics (CIFEr)*, pages 1–8, 2012.
- [Ras03] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.
- [RdB05] L Ralaivola and F d’Alche Buc. Time series filtering, smoothing and learning using the kernel kalman filter. *Proceedings of 2005 IEEE International Joint Conference on Neural Networks*, 3:1449–1454, 2005.
- [RS11] Thorsten Rheinlander and Jenny Sexton. *Hedging derivatives*, volume 15. World Scientific, 2011.
- [SB⁺99] Richard S Sutton, Andrew G Barto, et al. Reinforcement learning. *Journal of Cognitive Neuroscience*, 11(1):126–134, 1999.
- [SC10] SEC and CFTC. Findings regarding the market events of May 6, 2010. *Washington DC*, 2010.

- [SC19] Justin Sirignano and Rama Cont. Universal features of price formation in financial markets: Perspectives from deep learning. *Quantitative Finance*, 19(9):1449–1459, 2019.
- [Sew11] Martin Sewell. Characterization of financial time series. *Rn*, 11(01):01, 2011.
- [SFGK03] Eric Smith, J Doyne Farmer, László Gillemot, and Supriya Krishnamurthy. Statistical theory of the continuous double auction. *Quantitative Finance*, 3(6):481, 2003.
- [SH97] Jaap Spronk and Winfried Hallerbach. Financial modelling: Where to go? with an illustration for portfolio management. *European Journal of Operational Research*, 99(1):113–125, 1997.
- [SLA12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems*, 25, 2012.
- [SWD⁺17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [TBSY16] Andrew Todd, Peter Beling, William Scherer, and Steve Y Yang. Agent-based financial markets: A review of the methodology and domain. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–5, 2016.
- [TD20] Abhinava Tripathi and Alok Dixit. Liquidity of financial markets: a review. *Studies in Economics and Finance*, 37(2):201–227, 2020.
- [TGL13] Philip Treleaven, Michal Galas, and Vidhi Lalchand. Algorithmic trading review. *Communications of the ACM*, 56(11):76–85, 2013.
- [VGO⁺20] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

- [VW14] Tommi A Vuorenmaa and Liang Wang. An agent-based model of the flash crash of May 6, 2010, with policy implications. *Available at SSRN 2336772*, 2014.
- [WCGW24] Liyang Wang, Yu Cheng, Xingxin Gu, and Zhizhong Wu. Design and optimization of big data and machine learning-based risk monitoring system in financial markets. *arXiv preprint arXiv:2407.19352*, 2024.
- [Woo01] Jeffrey M Wooldridge. Applications of generalized method of moments estimation. *Journal of Economic Perspectives*, 15(4):87–100, 2001.
- [WW17] Xintong Wang and Michael P Wellman. Spoofing the limit order book: An agent-based model. In *AAAI Workshops*, 2017.
- [Zee74] E.C. Zeeman. On the unstable behaviour of stock exchanges. *Journal of Mathematical Economics*, 1(1):39–49, 1974.
- [ZZR19] Zihao Zhang, Stefan Zohren, and Stephen Roberts. Deeplob: Deep convolutional neural networks for limit order books. *IEEE Transactions on Signal Processing*, 67(11):3001–3012, 2019.

Appendix A

Parameter	Description	Parameter	Description
κ_1	FT: Coefficient for linear demand component	κ_2	FT: Coefficient for polynomial demand component
N_{FT}	FT: Number of FT	$S_{interval}^{FT}$	FT: Interval between fundamental trading activities.
β_L	LMT: Coefficient for demand calculation	α_L	LMT: Coefficient for demand calculation
N_{LMT}	LMT: Number of LMT	β_S	SMT: Coefficient for demand calculation
N_{SMT}	SMT: Number of SMT	α_S	SMT: Coefficient for demand calculation
σ_{NT}	NT: Coefficient for demand calculation	N_{NT}	NT: Number of NT
γ	LMT & SMT: Coefficient for demand calculation	δ	NT, LMT & SMT: Limit order cancellation rate
ρ	NT, LMT & SMT: Ratio between probability of submitting a market order and probability of submitting a limit order	V	NT, FT, LMT, SMT, MM: Order volume
μ_ℓ	NT, LMT & SMT: Mean of the log-normal distribution from which limit order price distance is sampled	Σ_ℓ	NT, LMT & SMT: Standard deviation of the log-normal distribution from which limit order price distance is sampled
N_{MM}	MM: Number of MM	δ_{MM}	MM: Limit order cancellation rate
θ_{MM}	MM: Probability of submitting a quote	p_{edge}^{MM}	MM: Maximum price distance of submitted order
ε_{limit}^{MM}	MM: Position limit	ε_{safe}^{MM}	MM: Safe position level
ε_{rest}^{MM}	MM: Time length for the trading suspension	N_{ST}	ST: Number of ST
N_{spike}	ST: Number of consecutive orders to be submitted after ST is activated	μ_{spike}	ST: Probability of being activated in a certain simulation step
V_{spike}	ST: Order volume	r	INS: Target execution rate
Q	INS: Number of initial inventory	n	INS: Number of seconds between consecutive market sell orders

Table 6.1 Descriptions for all parameters involved in the proposed agent-based model in Chapter 5.

Appendix B

Parameter	κ_1	κ_2	N_{FT}	$S_{interval}^{FT}$
Value	N.A.	N.A.	30	100
Parameter	β_L	α_L	N_{LMT}	β_S
Value	N.A.	0.001	30	N.A.
Parameter	N_{SMT}	α_S	σ_{NT}	N_{NT}
Value	30	0.9	N.A.	30
Parameter	γ	δ	ρ	V
Value	10	0.005	0.2	100
Parameter	μ_ℓ	Σ_ℓ	N_{MM}	δ_{MM}
Value	N.A.	0.3	20	0.05
Parameter	θ_{MM}	p_{edge}^{MM}	ε_{limit}^{MM}	ε_{safe}^{MM}
Value	N.A.	4	5000	101
Parameter	ε_{rest}^{MM}	N_{ST}	N_{spike}	μ_{spike}
Value	12000	N.A.	N.A.	N.A.
Parameter	V_{spike}	r	Q	n
Value	N.A.	N.A.	N.A.	N.A.

Table 6.2 Values for fixed model parameters in the calibration process in Chapter 5.

Appendix C

Parameter	κ_1	κ_2	N_{FT}	$S_{interval}^{FT}$
Value	0.1390	0.4562	30	100
Parameter	β_L	α_L	N_{LMT}	β_S
Value	0.3017	0.001	30	0.1273
Parameter	N_{SMT}	α_S	σ_{NT}	N_{NT}
Value	30	0.9	0.3403	30
Parameter	γ	δ	ρ	V
Value	10	0.005	0.2	100
Parameter	μ_ℓ	Σ_ℓ	N_{MM}	δ_{MM}
Value	1.9349	0.3	20	0.05
Parameter	θ_{MM}	p_{edge}^{MM}	ε_{limit}^{MM}	ε_{safe}^{MM}
Value	0.6624	4	7000	101
Parameter	ε_{rest}^{MM}	N_{ST}	N_{spike}	μ_{spike}
Value	12000	N.A.	N.A.	N.A.
Parameter	V_{spike}	r	Q	n
Value	N.A.	9%	120,000	12

Table 6.3 Values for model parameters in simulating the 2010 Flash Crash in Chapter 5.

Appendix D

Parameter	κ_1	κ_2	N_{FT}	$S_{interval}^{FT}$
Value	0.1390	0.4562	30	100
Parameter	β_L	α_L	N_{LMT}	β_S
Value	0.3017	0.001	30	0.1273
Parameter	N_{SMT}	α_S	σ_{NT}	N_{NT}
Value	30	0.9	0.3403	30
Parameter	γ	δ	ρ	V
Value	10	0.005	0.2	100
Parameter	μ_ℓ	Σ_ℓ	N_{MM}	δ_{MM}
Value	1.9349	0.3	5	0.05
Parameter	θ_{MM}	p_{edge}^{MM}	ε_{limit}^{MM}	ε_{safe}^{MM}
Value	0.6624	4	4000	101
Parameter	ε_{rest}^{MM}	N_{ST}	N_{spike}	μ_{spike}
Value	12000	2	4	0.005
Parameter	V_{spike}	r	Q	n
Value	100	N.A.	N.A.	N.A.

Table 6.4 Values for model parameters in simulating the example Mini Flash Crash in Chapter 5.