Master's Thesis

Modelling and analysis of a financial market with slow and fast trading agents acting on time-delayed market information

Halfdan Rump

Department of Computer Science and Engineerring
School of Fundamental Science and Engineering
Waseda University

Student ID	5115BG02-1
Date of Submission	February 5, 2014
Advisor	Prof. Toshiharu Sugawara

Contents

Abstract

This work proposes a model in which multiple heterogeneous agents use time delayed price information to trade an imaginary financial instrument in a market with a continuous double auction. The main innovation of the model is that, just as in the real world, trading agents do not have access to the market information at the same point in time, which means that the agents generally trade on different information. The model contains agent models of slow human traders using a noisy fundamentalist strategy, and high speed software traders using market maker and chartist strategies. Slow traders base their trading decisions on market information which has been delayed several seconds, while the fast traders observe the market with much smaller delays ranging between a few milliseconds to a few hundred milliseconds. By containing agents of such different speeds, the model is relevant to the ongoing discussion of the pros and cons of high frequency trading in financial markets.

The model simulates the market events in the minutes after the advent of bad news represented by sudden negative shock to the fundamental stock price, and does so with a time resolution high enough to register events that unfold from millisecond to millisecond. A genetic algorithm is used to search for model parameters that cause the market to be stable, and parameters that cause the market to crash. Analysis of the results shows that a moderate level of high speed trading activity is not in itself problematic. In fact, high speed market makers are found to reduce price flickering, while high speed chartists are found to decrease the time required for the market to respond to changes in the fundamental price. However, the market is found to respond unfavorably to a large presence of fast traders. First of all, a large presence of fast market makers causes the market to respond sluggishly to the shock, leading to a prolonged disparity between the traded price and the true fundamental price. Secondly, a large presence of fast chartists causes increased flickering of the traded price. Finally it is found that flash-crashes can occur in markets in which ratio of the number of chartists to the number of market makers is high, while at the same time the market makers are faster than the chartists. These results are interesting, as they show both benefits and dangers of having markets where fast computer algorithms trade side by side with human traders.

Contents

List of Figures

List of Tables

Part I
Introduction

In other words, by assigning agents in either group a latency of a certain number of rounds, the model allows the analysis of the impact of agent speed to be quantitative rather than qualitative. Models of the qualitative kind are useful for answering broad questions such as "do fast traders have an advantage over slow traders?' ' or "do fast traders make the markets unstable?". However, as the questions are qualitative, as must the answers necessarily be. Hence, the answers are likely to be along the lines of "sometimes", without providing any insight into which exact circumstances . After all, fast algorithmic traders are a fact of modern markets , and while it has both been shown that these traders make the markets more efficient and that high frequency traders can be linked to instabilities such as flash crashes, it is conceivable these pros and cons do not simply depend on the seed of the agents, but also on *which* agents are fast, and *how much faster* they are than other fast agents, etc. This line of thought was clearly expression in [20], where the authors strongly advocate for the development of a taxonomy of modern trading strategies employed by algorithm traders, as the authors argue that a market should be be thought of as an ecological system. In order to do this, a model must necessarily be able to emulate a trading environment in which the traders all have speeds that are relative to each other. In other words, a model in which the speed of the agents are quantitative rather than qualitative. Hence, this work focuses strongly on emulating real-time delays in the flow of information as they would exist in the real world.

1 Background

One of the great challenges of science today is to model systems in which the system dynamics are influenced human behavior. The purpose of trying to describe such systems through models is to obtain tools which allow prediction, and ultimately makes it possible to enforce control. Traditional economics tries to understand such systems, by using an analytic approach, which often limits the complexity of the model. In spite of these assumptions, classical economic theory has proven itself capable of modeling many of the dynamics of human society where the exchange of goods for money takes place, i.e., market places.

At its core, economic theory is mostly concerned with how large groups people can be expected to behave on average, by interacting with other human beings. As such, the theories are not suitable for explaining situations arising from recent technological developments:

Systems with human-machine interaction Markets have changed drastically in the last few decades. The use of information technology to handle tasks such as trade execution has become the norm. The Internet carries a vast amount of news around the globe in a matter of milliseconds, and computers are ready to react at a moments notice every minute of every day. Thus the need of humans to constantly monitor

markets in order to anticipate fast market changes has been replaced by the need to employ algorithms to perform such tasks. Furthermore, with the rapid development of artificial intelligence and soft computing, algorithms are now also involved in the process of make actual trading decisions. However, even though many of the tasks that before were performed by humans are now largely automatized, humans still play an important role in modern markets. After all, algorithms are (still) created by humans, although attempt are also made to partly automatize even this process: see [3] and [47]. Furthermore, humans still play a vital role in evaluating the significance of news, although (naturally) this process is also being increasingly automatized by trying to predict consensus from online news sources, such as newspaper articles, blogs and social networks: [17], [35], [24], [10]). No matter what the future of increasingly automatized markets will look like[1], there is no doubt that current markets are system where humans and machines interact with each other and a the financial markets.

Technology is the new cunning Situations where a few, exceptional individuals are responsible for a large part of the aggregate activity

As for the first item,

As for the second item, the financial sector has changed as well, due to the invention on a new set of trading strategies called high frequency trading. High frequency trading uses computers and algorithms to analyze market data and trade faster than any human can. By directly accessing the order book information which shows at which prices other traders are currently willing to trade, they are also able to get an accurate picture of the current market state. By placing themselves physically close to the markets (the practice know as co-location), they are able to do so at a very high time resolution. Again, by investing heavily in communication equipment, they are able to process large amounts of information and quickly reach trading decisions.

In other words, economic theory today needs to cope with a reality in which there are a relatively few number of traders who are exceptional in the sense that they have access to technology that other traders do not, and that they use that fact win over other humans traders and less sophisticated trading algorithms. Due to the increased complexity of the analysis required to understand such systems, and due to a number a violent financial crashes in recent years, some economists have started expressing an interest in the use of computational models to understand economic systems as a whole. A few such papers are mentioned in section 2.

In parallel, the idea of multi-agent systems has risen in the field of computer science. Because computers are increasingly linked together in networks, they interact with each other, and hence comprise a system in which their aggregated behavior can be analyzed.

Computers, or rather the software that we call agents, are relatively simple to model. If the software is simple enough, it might not even require modeling, and the software agents can be plugged directly into a simulation engine. Much of the work in done in MAS carried out by researchers from the computer science community the field of multi-agent

[1]The author envisions a scenario of "Skynet vs. W. Buffet"

simulation (MAS) has been concerned with modeling and analyzing such system where machines interact with other machines.

MAS is a well suited tool to deal with the complexities of the modern trading world, because it allows for the construction of models with complex dynamics. Naturally, the creation of realistic agent models is difficult. However, as is done in this paper, it is possible to capture essential characteristics of the problem under investigation. In this paper, we create a model of a simple trading environment with one market, and several agents trading on that market. The aim of creating this model is to find out how artificial markets behave when there is a communication delay between the agents and the market. The contents of this paper is as follows. First, in section 2, we summarize the main influences of this work, and explain a few of the ideas that were central to the development of our own model. Next, in section

2 Related work

This section contains a short summary of the literature that provided the main inspiration for this work. Due to the multi-disciplinary nature of the background of the researchers who contribute (economists, computer scientists, ecologists, sociologists, etc.), it is not within the scope of this section to give comprehensive review of the literature. Rather, the aim of this section is to show which influences lie behind this work, and where this work tries to add new contributions.

[58] proposes a round-based model in which a fast trader using a market maker strategy trades against a large number of slow traders, implemented as the stylized trader model laid out in [13]. The paper shows a link between the frequency with which the traders trade and their trading success. The notion of a trading frequency is also present in our model, as assign each agent with a time delay to the market which means that the agent can never trade faster than it takes for information to flow between itself and the market.

In [33], the authors show that as the time resolution with which information from stock market is observed is increased, the the distribution of price movements of the traded asset changes from the usually observed power law. suggesting that the presence of very fast traders creates markets which behave in fundamentally different ways. The authors then proceed to discuss the phenomenon of strategy crowding, which they argue is one reason why many flash-crashes can be observed in recent years. The crux of the argument is that the faster the agents have to be, the less information they have to trade on, and this means that there are only so possible conclusions that can be drawn. As the agents react similarly, they can cause crashes, especially if those agents are responsible for trading large volumes. Our model adds an extra condition for this to happen. Not only must the agents be similar, but they must also receive the same information at the same time, which may or may not happen.

In cite [6], the authors point out that the share of the volume traded by high frequency traders vary widely from market to market, but that in some markets it is estimated to be as high as 70%. The same paper also goes to great lengths to specify exactly what high frequency trading is, and lists a number of defining characteristics. A few of them are

"Very high number of orders", "rapid order cancellations", "profit from buying and selling (as middlemen)", "low latency requirement", "use of co-location/proximity services and individual data feeds". These are (some of) the typical characteristics that high frequency traders exhibit, and they have all been replicated by our model. If we want to understand markets in which high frequency trading is taking place, these characteristics should be replicated, rather than trying to mimic specific strategies which are used in the real world. Another argument for this is that, unlike the more traditional chartist strategies, modern algorithmic trading trading strategies use complex probabilistic models, which are very difficult (if not impossible) to reverse-engineer. In many cases, the only way to judge the efficacy of a trading algorithm is to implement several types of algorithms and let them trade against each other, as was done in [31].

Because of the high degree of secrecy surrounding which algorithms are actually being used for high frequency trading, and since the practice is relatively new, [39] argues that there is a general lack of knowledge about the topic, even among professionals in the financial field. To remedy that fact, it tries to establish a broad taxonomy of strategies employed by high frequency traders, and basically divides them into market making strategies, arbitrage strategies, and market rebate trading strategies.

[20] is another work arguing for a taxonomy of the trading strategies, but does so from an ecological perspective by assigning roles of prey and predator to traders. This way of thinking is closely related to the discussion of whether or not HFT is beneficial or harmful to markets, as it is intuitive to think of HFTs as the predators and other traders as they prey. The default assumption seems to be that it is not beneficial, partly because of the large profits, but some findings seem to indicate that HFT actually improve liquidity in markets, and argue that the profit gained by the HFT is the cost imposed on other traders. Indeed, since financial institutions expected of employing HFTs themselves seem to affected by flash crashes in their company stock, as was pointed out in [33]. [40] indirectly presents an argument for this notion of predator vs. prey. It is interested in the phenomenon of flickering quotes, i.e., small, momentary differences in trade prices between markets. The paper uses a simple, round-based model, and presents equations for when the fast traders show use their speed advantage to deliberately sell at sub-prime prices to the slow traders.

For works on the MAS methodology in general, see [15, 23, 38]. A review of early works in the field of simulated finance can be found in [36] and provides a good introduction to the field. For a few other works on agent-based computational economics, see [48, 55, 56, 37].

In summary, much interesting work, which aims to understand the role and impact of high frequency trading and algorithmic trading in general, has been done. Some works try to reach conclusions by analyzing real stock-data, whereas other papers approach the problem differently by first building a model, and then perform experiments to see if realistic dynamics can be replicated by the model. The latter approach is typical for papers in the MAS field, and this thesis belongs to that category is one such paper. However, to the limit of the author's knowledge, no MAS-model has yet attempted a real-time simulation approach as is proposed in this paper. Since the timing of trading is very important for high frequency trading, we believe that a model which simulates that

timing is necessary.

Part II
Model

As explained in the previous section, perhaps the most distinguishable aspect of high frequency trading is the speed with which agents can react to new market information. It is therefore essential that a model should capture this aspect, if it is to be used to draw generalized conclusions about the influence of high frequency trading in the markets.

The model consists of a market and agents. Agents and the market communicate by exchanging messages which all arrive one or several rounds after they are issued. A complete simulation consists of the several consecutive rounds. In each rounds, some agents submit orders, while others wait for new market information. Order messages arrive at the order books, and trades are executed when prices match. The following sections will describe the model in detail.

3 Model overview

As has already been pointed out, the primary goal of the model is to simulate the presence of delays in communication between agents and then market. Furthermore, unlike previous models in which agents are divided into slow and fast traders, the model proposed in this work puts emphasis on modeling the real world scenario in which the latency with which the agents trade is an arbitrary period of time. As is discussed in section 4, agents in this model are also divided into two groups of slow and fast traders, but since agents in either group have delays of an arbitrary number of simulation rounds, this division of the agents is much less strict than in previous models, such as [40, 28, 21].

Each round of the simulation corresponds to a period of real-time, but it is not particularly important to specify how long that period is. Instead, what matters is that there is a difference in speed between the agents. In other words, the important thing is that some agents are much faster than other agents. If one thinks of each round as a millisecond of real-time, one realizes that an agent simulating a human trader will require several thousands of simulation rounds to react to market news. On the other hand, fast algorithmic traders may only require a few rounds, making them several orders of magnitude faster than the slow traders.

The model contains four main components and each of these will be described in the following subsections.

- The stock which is traded by the agents
- The market with a set of rules and an order book which handles transactions
- The trading agents
- The various messages that are passed between the agents and the market

Since there is relatively much to say on the agent strategies, section 4 has been devoted entirely to this subject. Figure 1 illustrates the relation between the agents and the market.

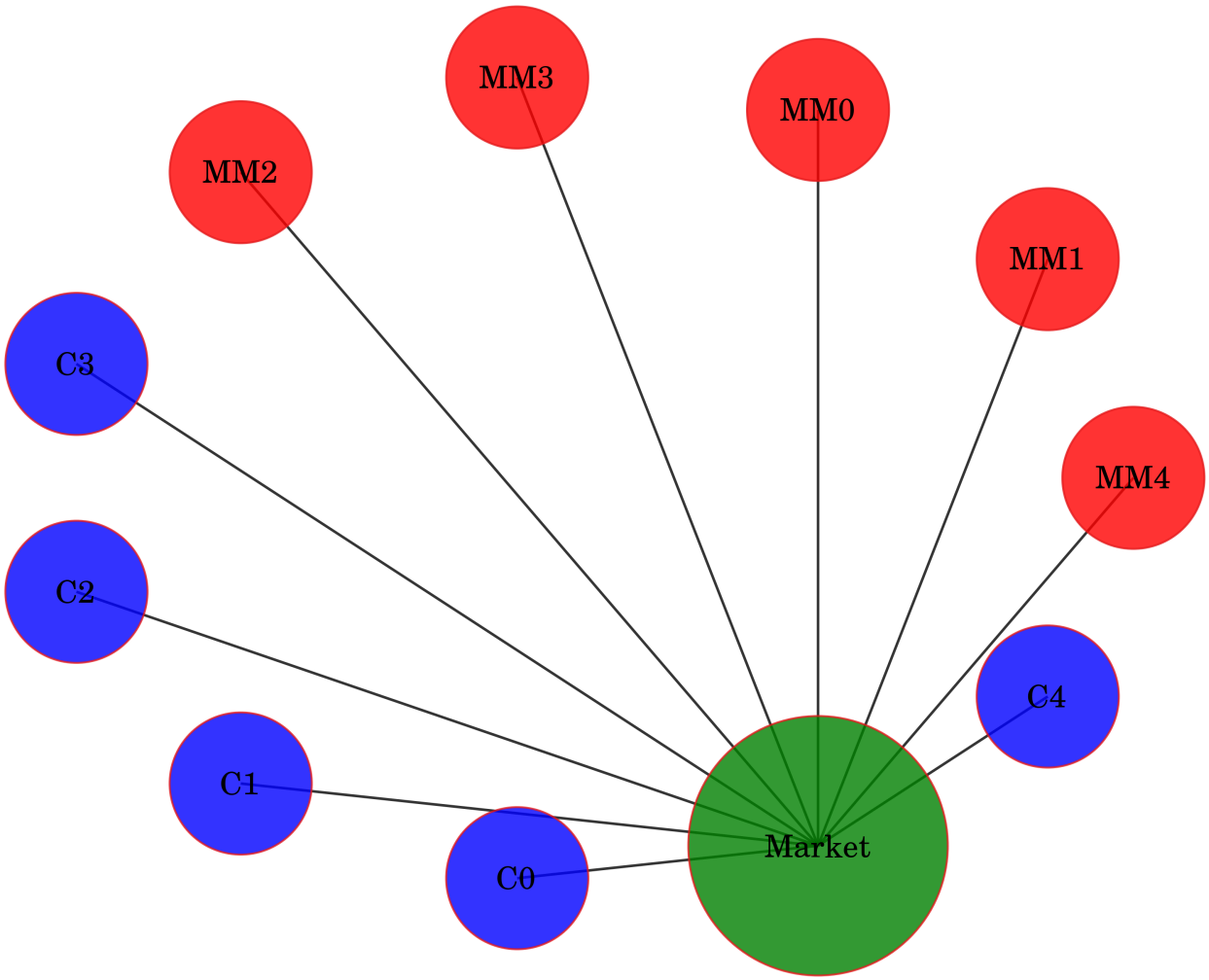


Figure 1: Diagram of the market model. Agents (blue and red nodes) have a various distances to the market (green node), as indicated by the different lengths of the edges.

3.1 Simulation rounds

A complete simulation consists of executing 10^5 consecutive rounds. Each round is composed by a number of steps, which can be divided into five parts as described below.

The round is initialized Time is incremented and the fundamental price is updated. In this thesis, the only change to the fundamental price is at time $t_{=}t_{T_\eta}$, where the fundamental price changes from F_0 to $\eta = F_0 - \eta$.

Arriving messages are dispatched to recipients All messages with arrival time t_T are delivered. Arriving transaction receipts delivered to the agents involved in the trades, and the portfolios of these agents are updated. Similarly, arriving orders and order cancellations are delivered to the market and placed in the order book message queue. Finally, all messages containing market information are received by the agents that requested the information.

Slow trader activity The number of slow trader orders that arrive in this round is sampled from equation 3 and the orders are created using the equations in section 4.1.1. The orders are instantly placed in the order book message queue.

Fast trader activity All HFT agents finishing evaluation of their strategies wake up and execute their decisions by sending out new orders containing orders and order cancellations. Each market maker MM_i Each chartist SC_j starts its hibernation period of W_c^j.

Order book is updated All orders expiring this round are removed from the order book. Next, messages waiting in the message queue are processed in random order. Orders targeted by order cancellations are removed from the book, and transaction receipts are created for each pair of buy and sell orders with matching prices. After all messages have been processed, the best bid and ask prices, B_T and A_T, are set equal to the prices of the two most competitive remaining orders. In the case that no changed were made to the order book in round t_T, $B_T = B_{T-1}$ and $A_T = A_{T-1}$.

3.2 The stock and its prices

A single stock is traded in the market. The market uses a continuous double auction to match orders. As with anything that is traded in an auction, the traded asset cannot be said to have a single defining price. Instead, the value of the asset can be defined in (at least) three ways, as described below

Fundamental price Often referred to as the "true" value of the stock. The fundamental price is established by fundamental analysts, who try to estimate the stock price by analyzing the worth and wealth of the corporation that the stock represents. Hence, an accurate estimate of the fundamental price is arrived at only after careful (and time consuming) analysis. The fundamental price changes with the performance of the corporation, and is typically influenced by news. In this work, the fundamental price at round $t_T$is denoted F_T.

Best offered prices In the market, a stock is only worth as much as people are willing to pay for it. The value of the stock falls and rises according to what beliefs people hold about the worth of the stock. The best offered prices show what traders are

currently willing to pay for the stock, and what price they are willing to sell for, and does not necessarily reflect the fundamental price. The best buy and sell price at round t_T (e.g. the highest buy price and the lowest sell price) are denoted B_T and A_T.

Traded price Both of the two previous types of prices provide equally reasonable estimates of what the stock is actually worth. However, the willingness to trade does not always depend on the actual value of the stock. Sometimes traders need to sell their shares to acquire capital, and are therefore forced to lower the price at which they are willing to sell. Hence, a third way to estimate the value of a stock is to record the prices at which the stock was actually sold. The price of the i'th transaction in round t_T is denoted $p_i^t(T)$. The double auction works in such a way that trades are always executed at the best prices currently existing in the order book at the time of the trade. However, since the market depth is finite, a large quantity of orders will move A_T and B_T, and consequently it frequently happens that $p_i^t(N) \neq p_{i+1}^t(N)$.

The fundamental price is used as the benchmark price for the slow traders (also called the fundamentalists), the best buy/sell prices are used by the fast market makers, and the traded price is used by the fast chartists. See section 4 for more details. Whether or not one type of strategy is more accurate than the other, it is a fact that both types are employed by traders. Hence, a model of such an environment needs to include all three types of prices.

3.3 Order book

The order book is a record of all unmatched orders for a single stock. Since any buy-and sell orders submitted at the same price will cause a trade to be executed, and the matched orders to subsequently be removed, there must at any point of time during the simulation be a non-negative price difference between the sell order with he lowest price and the buy order with the highest price. This difference is called the *spread*, and is denoted as follows

$$s_{t_T} = A_{t_T} - B_{t_T} \tag{1}$$

where A_{t_T} and B_{t_T} are the best ask and bid prices in round t_T.

3.3.1 Price updating

Each time an order is added or removed, the order book has to update the best bid and ask prices. Since it often happens that several orders arrive in the same round, the order book spread can fluctuate within a single round. However, since one round is considered the minimum quantum of time, these within-round fluctuations are not recorded in the order book history. Instead, after all orders have been processed, the resulting best bid and ask prices are registered as the best prices for that round. Agents that look at the market will therefore only be able to see the state of the order book after the book has

Sell orders	Price	Buy orders
	⋮	
22	9994	
26	9993	
13	9992	
10	9991	
	9988	12
	9987	10
	9986	16
	9985	25
	⋮	

Table 1: Example of an order book in the price range near the best buy/sell prices

finished processing all price changes due to the arrival or removal of orders. The subscript denoting time in equation 1 therefore refers to the prices at the end of that round.

When no orders arrive or are removed from the order book, the prices are updated as $A_{t_T+1} = A_{t_T}$ and $B_{t_T+1} = B_{t_T}$. Since orders can be removed due to cancellations or because they expire, the order in which incoming messages is processed matters to the outcome of A_{t_T} and B_{t_T}. Messages are therefore processed in random orders, so that no agent is favored.

3.3.2 Order matching

When a trade is executed between orders o_1 and o_2, the traded volume is

$$\Delta v = \min(v_{o_1}, v_{o_1})$$

If $v_{o_1} = v_{o_2}$ the orders are *fully matched*, and both are removed from the order book. In the case of a *partial match*, that is, if the volume of one order is larger than the volume of the other, then the order with the smaller volume is removed, and the volume of the other order is subtracted by Δv. The price of the transaction is the price of the market order which was already in the book.

Each agent knows the volume of every order that it submitted, when the order was dispatched. However, when an order is partially filled by a matching but smaller order, the volume of the order at the market changes. Since it takes time for the order to be transmitted for the agent to the market, the agent cannot immediately update its knowledge of the order volume. In this case the order has one volume at the agent side and another in the market side. The momentary disparity of agent market side and agent side volumes can have several consequences, such as agents short-selling without, agents submitting cancellations for orders which have already been filled. Unlike volumes, the

price of a standing market order does not change, and hence the situation of a disparity between market- and agent-side price knowledge does not occur.

3.4 Messages

All communication between the market and agents is transmitted in messages and all messages take a non-zero number of rounds to arrive. This means that no information is transmitted instantly between agent and market. The latency between any agent and the market is constant throughout the simulation, but different for each agent. The delay between market maker MM_i and the market is λ_m^i, and the distance between chartist SC_j and the market is λ_c^j. All latencies between agents and markets are non-negative integers. A message created in round t_T by an agent with a latency of λ rounds to the market with arrive at the market in round $t_= t_T + \lambda$. Several message types were implemented in order to accommodate the various types of communication.

3.4.1 Market information

One of the key points of simulating delays is that agents always trade on old information. Before an agent can evaluate its strategy, it has to request the most recent market information. In a model without delays, an agent would simply receive the state of the market in the current round, but when information is delayed the process is somewhat more cumbersome. First the agent sends off a request to obtain the information about the market state. When the request arrives at the market some rounds later, the market serves the request and by sending back another message containing the information. The contents of this message depends on the agent strategy, as the various agent strategies require different information. In the case of a single market, it is reasonable to simplify the model such that the market serves the request instantaneously, since any delay inherent in the market is common for all agents. After a further delay, the message containing the market state finally arrives at the agent, and the agent can then start evaluating its strategy.

3.4.2 Orders

An order is a message which is sent from an agent to a market when the agent has decided to trade. An order is an offer to buy or sell a specified number of shares at a certain price at a certain market. Orders can either be limit orders or market orders. A limit order will only result in a trade to be executed if there is a matching order when it arrives to the market. A market order will stay in the order book until a matching order arrives, or until it expires after a number of rounds set by the submitting agent.

When an agent creates an order, the agent first of all decides on whether the order is a sell order or a buy order, a limit order or a market order. The agent also specifies the price and the volume of the order. Details on how each type of agent does this can be found in section 4. Once the price has been decided upon, it stays fixed for the duration of the order lifetime. On the other hand, the volume does not remain constant. If the agent submits a market order, the order sits in the order book for a number of rounds before it

expires. If during that time there is an order on the other side of the book with the same price, a trade is executed. If the two orders involved in the trade do not have identical orders, a partial match occurs (see section 3.3.2), and the volume of the largest order is updated. When this happens, the market instantly sends out a transaction receipt to the agent. However, since it takes time for the receipt to reach the agent, there is a period of time during which the agent is unaware that the order that it places has been involved in a trade. Hence, every order has two simultaneous volumes. The market-side volume specifies the actual remaining volume of the order as it is in the order book, while the agent-side volume reflect what the agent knows about the order. Whenever an order is involved in a trade, a discrepancy between the market-side volume and the agent-side volume occurs, and is resolved as the transaction receipt reaches the agent.

3.4.3 Transaction receipts

When two orders match, a receipt is sent to each of the two agents involved in the trade. The seller receives a receipt specifying the number of shares that it has to deliver, and the buyer gets a receipt for the amount of cash to be paid. Because of the transmission delay, the agents do not update their portfolios when the trade actually happens, but when they receive the receipt. In the case that an agent does not have enough shares or cash in its portfolio, the agent is allowed to borrow the necessary assets, thus bringing its portfolio into negative. An agent cannot submit new sell orders while holding a negative number of shares. Similarly, an agent cannot submit any new buy orders while having a negative amount of cash. In the case that the agent has neither cash nor shares, it simply becomes inactive.

3.4.4 Order cancellations

It can happen that an agent wants to change a previously submitted order, or cancel it entirely. In fact, this is what the market maker agent does frequently, as described in section 4.2. In this case, the agent issues a message to the market requesting that the order should be removed. Due to the presence of delays, the agent's order might be filled before the cancellation reaches the market, in which case the market will ignore the request to cancel.

3.5 Market rules for short selling

Although some market do allow deliberate short selling, this practice is not allowed in the simulation. That is, an agent is not allowed to place a sell order for more stock than it has in its portfolio at the time it places the order. However, due to the presence of delays, it can happen that an agent is required to deliver on a sell order for more stocks than it holds when notified of the order. A sequence of events which causes this to happen is illustrated on figure 2. The agent who is short is required to deliver the stocks, and thus goes into negative on its portfolio, and has to buy back the stocks before it can place further sell orders. Although the sequence of events shown in figure 2 may seem unlikely,

it did in fact occur frequently, making it necessary to implement handling of this special case.

Figure 2: The HFT agent submits a sell order for 100 stocks, and another agent submits a price-matching buy order which fills the sell order. Before the transaction receipt reaches the seller, the seller decides to cancel the order, and submit another order at a different price. When the transaction receipt reaches the seller, the agent promptly sends out a cancellation of its second sell order, as it knows it cannot fulfill the order. However, before the cancellation reaches the market, a third agent fills the sell order, and a receipt is sent to the seller who ends up being short.

4 Agents

A taxonomy of agents which is frequently found in the literature dealing with the speed of traders, is a division of traders into a group of fast agents and a group of slow agents ([21, 28, 40]). While the model proposed in this work uses a similar division, it is less rigid as agents in each group can have arbitrarily large (or small) latencies. Rather, the division of agents into slow and fast traders is intended to capture traits of traders in the real world. Hence, the slow traders are meant to model human traders, which tend to react slowly to market information but are capable of interpreting world events that influence the stock price. The slow traders are also intended to model algorithmic traders which use human-like strategies, but do not benefit from co-location. On the other hand,

the fast traders, also referred to as HFTs (first time explaining it despite referencing it twice earlier in diagrams), are intended to model algorithmic traders which do make use of co-location [11] and use strategies of a lower complexity and variety compared to human traders [12]. In this report, HFT agents will often be referred to as being "fast" or "slow" when talking about agents with small or large latencies to the market. While an agent's latency to the market is unrelated to how fast or slow an agent is (other parameters, such as T_m^i, T_c^i is more related to an agent's speed), the agent's latency does determine the delay with which the agent received market information. The terms fast and slow agents is therefore meant as agents that can or cannot respond quickly to new market events.

Another way to group agents is by the nature of their strategy. In financial markets, every trader is supposed to have access to the same information. However, two traders might disagree on the meaning of some piece of information. The way in which a trader evaluates market information and reaches a conclusion on how to trade is called a strategy. While any function which takes some information relevant to the market as input and gives a decision of how to trade (or not trade) can be termed a strategy, it is useful to divide strategies into two broad categories. In the first category are strategies which are dubbed chartist strategies, which basically tries to extrapolate on the past price movements. An agent modeling this behavior is described in section 4.3. In the other category are strategies which are based on some analysis of the true value of the stock, called the fundamental value. This behavior is replicated as described in section 4.1. The market maker strategy explained in section 4.2 does not really belong to neither of these two groups, as the agent neither tries to extrapolate on past data, not use fundamental analysis for price determination. The above classification of agents is by far meant to be definitive.

4.1 Slow traders

The slow trader model used in this work is inspired by the stylized trader model used in [58, 13, 27]. However, due to the fundamental differences in the way that the simulation works, the model has been modified significantly. The idea of the model is that there are three basic trading strategies techniques that any human trader trader mixes to form his own personal strategy.

Fundamental analysis Traders subscribing to this way of thinking believe that they can know the true value of a stock by estimating the fundamental price (see section 3.2). Furthermore, such traders believe that any deviation from the fundamental price is due to other traders misinterpreting the market, and that such deviations will eventually disappear. In other words, given enough time, the traded price will converge towards the fundamental price.

Technical analysis Traders using technical analysis do not care whether or not the stock is overvalued. Instead, they believe that they can predict future price movements from past data. Traditional technical analyst approaches extrapolates on price movements by using simple mathematical models and a good deal of heuristics.

Noise trading Some traders are in possession of insider knowledge, which means that they think that they know something about the stock that others do not. They use this information to trade the stock, for better or for worse. Such traders were dubbed noise traders, since it is highly unlikely that any one individual would come in possession of information which actually gives that person an advantage in the market. Any such belief is therefore naive, and might as well inflict a loss on the agent than generate a profit.

The orders submitted the stylized traders throughout the run of the simulation should be thought of as being submitted by a variety of different agents, all observing the same historical data, but interpreting it differently using different strategies. In other words, the model for the slow traders proposed in this work should thought of as representing a group of traders [18]. Hence, even though all the slow traders are fundamentalists, they all have slightly differing opinions on what the fundamental price actually is at any given moment. This difference in opinion is modeled by adding normally distributed noise to the price estimate, as described below. In this model the fundamental and noise trader parts of the stylized trader strategy are preserved, but the chartist behavior is removed. Most chartist strategies operate on a timescale of days to weeks to months. However, the simulation proposed in this work merely simulates a few minutes of real-time[2]. Any chartist strategy based on the entire history of price movement over a long period of time will simply be too sluggish to follow with the high frequency price fluctuations occurring within the simulation. In other words, to a slow trader that relies heavily on the chartists part of the stylized trader strategy and calculates the moving average of the traded price over a period of days or months, any price changes that take place during a few seconds will hardly register at all. Even to relatively fast stylized traders that rely of minute-by-minute price changes, the time span simulation by the model is so short that the price changes that take place during the simulation will be smoothed out by trader's moving average calculation. This does not mean that the model neglects to recognize the fact that slow trader chartists do exist, but since these traders base their price estimates on price data from the past days, weeks or even months, it is not possible to simulate these strategies directly as it would require an exorbitant number of simulation rounds. Instead, the presence of chartists are modeled by adding fast agents using a short-term chartist strategy, which is described in section 4.3.

4.1.1 Strategy

The slow trader observed the market with a delay of λ_{ST}. Hence, in round t_T, the slow trader knows the price of the true fundamental in round $t_T - \lambda_{\mathrm{ST}}$, which will be dubbed $F_{T-\lambda_{\mathrm{ST}}}$. The noisy part of the agent strategy [29] adds a normally distributed number to $F_{T-\lambda_{\mathrm{ST}}}$, such that the fundamental price estimated by the slow trader is

$$F_{\mathrm{est}} = F_{T-\lambda_{\mathrm{ST}}} + \epsilon_{\mathrm{ST}} \tag{2}$$

[2]This is not entirely accurate as each round can be interpreted as an arbitrary length of real-time. However, since we are interested in what phenomena occur when we have some agents which are several orders of magnitude faster than other agents, we need a high time resolution, effectively capping the length of real-time which can be simulated

where $\epsilon_{\mathrm{ST}} \sim \mathcal{N}(\mu_\epsilon, \sigma_\epsilon)$. The trader also observed the the best prices in the order book $A_{t_T-\lambda_{\mathrm{ST}}}$ and $B_{t_T-\lambda_{\mathrm{ST}}}$. The slow trader uses his estimate of the fundamental price to determine whether to buy or sell, and at which price.

Place a buy order if	$F_{\mathrm{est}} > B_{t_T-\lambda_{\mathrm{ST}}}$	at buy price	$p_{\mathrm{b}} = B_{t_T-\lambda_{\mathrm{ST}}} + C$
Place a sell order if	$F_{\mathrm{est}} < A_{t_T-\lambda_{\mathrm{ST}}}$	at sell price	$p_{\mathrm{s}} = A_{t_T-\lambda_{\mathrm{ST}}} + C$

Hence, the slow trader will try to acquire shares when he believes that the fundamental price is higher than the current best bid, and sell shares if the fundamental price is lower than the current best ask price. If the estimated fundamental price is between the delayed bid/ask prices, that is if $B_{t_T-\lambda_{\mathrm{ST}}} < F_{\mathrm{est}} < A_{t_T-\lambda_{\mathrm{ST}}}$, the agent chooses to place a sell or buy order with equal probability with the same prices as in the above equations.

4.1.2 Order arrival

The arrival of orders from slow traders is modeled as a Poisson process, as this is a model which has been used in other works as well (see [14] and [4]). Hence, such that the number of new orders any round t_T is a random variable, κ_T sampled from a Poisson distribution:

$$\kappa_T \sim Pois(\xi_{\mathrm{ST}}) \tag{3}$$

In [4] the number of buy order and the number of sell orders are distributed according to two separate Poisson processes. In this work the market is assumed to be in a stable state prior to the shock to the fundamental price, and hence the average number of arriving buy orders is assumed to be the same as the average number of sell orders. Therefore a single Poisson process is used to obtain the number of newly arriving orders, and these orders are then divided into sell and buy orders with equal probability. Furthermore, since the model only simulated a short span of time after the shock, it is reasonable to assume that λ is constant as the population of slow traders can be assumed not to be exhibit any herding behavior which might occur over longer periods of time.

4.2 HFT Market makers

Market making is the term used for strategies in which the trader is simultaneously active on both sides of the order book. The trader attempts to make a profit by submitting sell orders at price A_δ and buy order at price B_δ, making sure that $A_\delta > B_\delta$. Here, δ is used to indicate the the best bid and ask prices that the market maker is trying to match are delayed according to the total latency of the agent. The difference between the bid price and the ask price is the market maker's spread. Trading with a small spread makes the market maker competitive by attracting orders from other traders, but also makes the market maker incur risk by increasing the risk of trading at sub-optimal prices in the case of sudden price fluctuations.

4.2.1 Strategy outline

The market maker aggressively tries to maintain orders at the best buy and sell prices. As other agents submit orders, the buy and sell prices move up and down, and the difference

between the best buy price and the best sell price, also known as the spread, increases and decreases accordingly. Since the agent tries to maximize its own profit, it chooses prices which maximizes the spread. Hence, when the spread increases, the agent is happy to let it do so, as it can then buy and sell with a larger profit on each unit of the traded asset. On the other hand, when the spread decreases, the agent also changes its prices in order to attract trades which would otherwise go to other agents. However, if the spread becomes too small, the profit on each unit of the traded asset becomes too small to justify the risk of holding stock which might suddenly drop in value. Therefore the strategy employed by the agent specifies a minimum accepted spread at which the agent is willing to trade. When the market maker receives a notification that one of its orders has been filled, it immediately uses the most recent locally known market information to submit a new order. For the sake of simplicity, each market maker agent is allowed to have a single market order at each side of the order book. The agent can therefore not stack orders on either side of the order book.

4.2.2 Parameterization

The market maker behavior is controlled by the following parameters

Minimum spread The parameter θ_i controls the smallest spread at which the agent will trade. The agent will never submit orders such that the price difference between his standing buy and sell orders is smaller than λ_m.

Order volume Market maker MM_i submits orders with volume V_m^j. Since the market maker tries to maintain standing orders at the orderbook for as much of the time as possible, V_m^i is usually chosen to be fairly large. λ_m^i is selected when the agent is created, and is constant throughout the simulation.

4.2.3 Price determination

When the agent receives the delayed trade prices from the market, the spread can either increase or decrease relative to the agent's previously known prices. In the former case, the agent always updates its prices to follow the widening gap between the buy side and the sell side. When the spread decreases, the strategy requires the agent to check that the minimum spread condition, that is whether or not $s_r < \theta_i$, where s_r is spread the calculated from the trade prices that the agent has just *received*. If the condition is not violated, then the agent can freely update prices to follow the current trade prices. If the condition is violated, then the agent determines the trade price by the following equations. p_n^b and p_n^s are the *new* buy and sell prices, p_r^b and p_r^s are the best buy and sell prices that the agent currently knows of, and Δp^b and Δp^s are the absolute differences of the prices:

that is $\Delta p^b = |p_n^b - p_r^b|$ and $\Delta p^s = |p_n^s - p_r^s|$.

$$\left.\begin{array}{l} p_n^b = p_r^b \\ p_n^s = p_r^b + \theta_i \end{array}\right\} \quad \text{when} \quad p_n^b = p_r^b, p_r^s < p_r^s \tag{4}$$

$$\left.\begin{array}{l} p_n^b = p_r^s - s_m \\ p_n^s = p_r^s \end{array}\right\} \quad \text{when} \quad \begin{array}{l} p_n^b > p_r^b \; p_n^b > p_r^b, p_n^s = p_r^s \\ p_n^s = p_r^s \end{array} \tag{5}$$

$$\left.\begin{array}{l} p_n^b = p_r^s - (s_m - s_r)\frac{\Delta p^b}{\Delta p^b + \Delta p^s} \\ p_n^s = p_r^b + (s_m - s_r)\frac{\Delta p^s}{\Delta p^b + \Delta p^s} \end{array}\right\} \quad \text{when} \quad \begin{array}{l} p_n^b > p_r^b \\ p_n^s < p_r^s \end{array} \tag{6}$$

The equations in 6 are used in the case where the both the buy and sell prices moves so that the spread becomes smaller than s_m. The equations are designed so that the agent will change its trade price in proportion to the change on either side of the book. For instance, if the buy price suddenly increases a lot while the sell side only decreases a bit, such that $s_t < s < m$, the agent will increase his own buy side price by a lot and decrease his sell price side by a little, so that the spread of between the new orders are $s_n = p_n^s - p_n^b = s_m$.

Figure 3 shows three simple examples of how the market maker attempts to follow the best market prices. δ is used to designate the total amount of time that elapses between the prices at the market change and until the market maker can have a new order at the market, and thus includes both the delay from λ_m and T_m.

In the first case, the spread increases, and the market updates its own prices to follow the increasing spread. In round two to eight, the market maker is selling at a price of 13 ticks, while the best market sell price is two ticks higher at 15 ticks. This means that the delay causes the market maker to momentarily sell at sub-optimal prices, and hence would makes 2 ticks less of profit per share if its order is filled before the market makers new order reaches the market.

In the second example, the spread decreases, and the market maker updates its prices to a lower spread in order to stay competitive. In this case, the delay in the market maker's ability to react to the market events causes the market maker to momentarily be less competitive than possible, causing a risk of losing trades.

In the third and fourth examples, the minimum spread parameter θ is seen to force the market maker to trade at sub-optimal prices, as the spread narrows below four ticks.

4.2.4 Decision cycle

Any market maker MM_i be thought of as having a trading cycle, the length of which depends on the parameters λ_m^i and T_m^i.

Acquire market information The agent requests market information at round t_{T_0}. The request reaches the market at round $t_{T_0} + \lambda_m^i$, and the market instantly sends out a reply.

Evaluate market information The agent receives the market information at round $t_{T_0} + 2\lambda_m^i$, and evaluate its strategy.

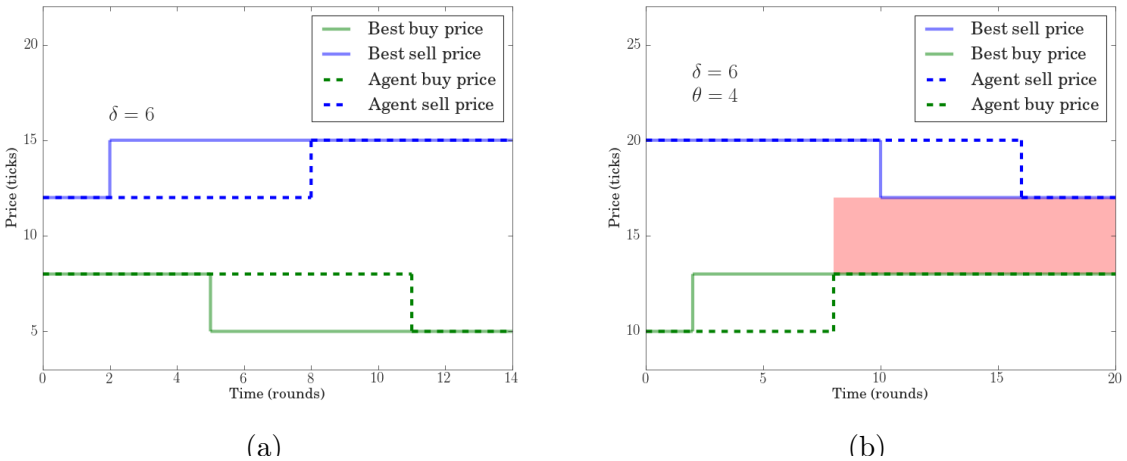


Figure 3: Examples illustrating the how the market maker updates its prices

Execute trade decision The agent reaches a conclusion on how to trade at round $t_{T_0} + 2\lambda_m^i + T_m^i$, and sends off an order to the market. The order reaches the market at round $t_{T_0} + 3\lambda_m^i + T_m^i$.

As soon as MM_i has submitted the new order, it also submits a new request for market information. The agent receives this second batch of market information at round $t_{T_0} + 3\lambda_m^i + T_m^i$, and the cycle restarts. The length of the cycle of MM_i is therefore defined as

$$l_{\text{MM}}^i = 3\lambda_m^i + T_m^i \tag{7}$$

The length of the cycle is important for several reasons. First of all, since the market only requests new market information once during the cycle, the market maker observed the changes in the market with a frequency of $\frac{1}{l_{\text{MM}}^i}$. Any market events that take place within a period smaller than l_{MM}^i can be only partially observed by the market maker.

Secondly, since the market maker can only have a single order on each side of the order book, the latency to the market decides how much volume the market maker is able to buy and sell over a period of time. The market maker i submits orders with a fixed volume V_m^i and the volume that the agent can supply the market with therefore depends on how much of the time the market maker manages to have standing orders at the market.

When an order submitted by the market maker is filled, the market sends out a transaction receipt which takes λ_m^i rounds to reach the market maker. When the market maker receives the receipt, the agent restarts the cycle by submitting a request for market information in order to submit a new order. This means that a total number of $4\lambda_m^i + T_m^i$ will elapse from time komma???time that the order was filled to the time when the market maker can have another order placed in the order book. During this time, the market maker does not have an order in the market on the side of the book that the filled order was at. When the market maker does not have a standing order on either side of the book, it means that the agent does not contribute to increasing the liquidity of the asset. Hence, in this simple model, fast market maker should do a better job of supplying liquidity than slow market makers.

4.3 HFT Chartists

The slow traders do not directly incorporate a chartist element into their strategy, as any changes in the traded price that occurs in the short time that the model simulates would be undetected by the slow traders. However, in order to incorporate chartist behavior in the model, a fast trader agent using a chartist strategy was implemented. These agents will from now on be referred to as HFT chartists, or simply as chartists.

4.3.1 Strategy outline

The chartists use a simple strategy, in which they continually calculate the moving average of the traded prices over a time windows [31]. If the current price drops below the moving average, the chartist believes that it has detected a downtrend. The chartists believes that

the downtrend will continue for a while, and it will therefore try to get rid of any shares that it is currently holding. In order to accomplish this, the chartists starts to submit sell orders. The more urgently the agent is trying to sell its stocks, the lower it will set the price of the sell order, as a sell order at a lower price is more likely to be filled. On the other hand, if current price rises above the moving average, the agent believes that it has detected an up-trend, and that the stock will be worth more in the near future. The agents therefore starts to submit buy orders starts to submit buy orders at a higher price. The more strongly the agent believes in the stock price increasing, the higher it is willing to set the buy price. The agent only submit orders when it has detected a trend. As long as the chartist does not detect a trend, the agent is inactive and merely observes the market data. After submitting an order, the agent waits for a while, as a precaution against submitting too many orders which would make the agent incur great losses the trend detected by the agent turns out to be wrong. This mechanism is also a limiting assumption of the model instated in order to protect the market from a few market makers flooding the market with a huge number of orders. As another step towards preventing the order books from filling up with chartist orders that might never be filled, and thus slow down the execution of the simulation, is to make the chartists submit limit orders.

4.3.2 Parameterization

The agent strategy is parameterized as follows. HFT chartist agent SC_j has the following parameters:

Time horizon The parameter H_c^j determines the width of the window over which the agent calculates the moving average. Larger values of H_c^j makes the agent use trade price data from further in the past and makes the moving average less responsive to sudden change, while lower values makes the moving average react faster to sudden movements in the price.

Sensitivity The sensitivity of SC_j is given by the parameter γ_c^j and refers to the number of ticks that the moving average of the traded price must differ from the currently traded price before the chartist recognizes that it has detected a trend. The higher the value of γ_c^j, the less often the chartist will detect a trend. In the case of $\gamma_c^j = 0$, the agent will detect a trend whenever $M \neq p_c$, which will be true in the majority of rounds.

Aggressiveness One the chartist decides to trade, it needs to decide upon a price. The aggressiveness parameter A_c^j controls how far from $p_c$that the agent will set the price. This parameter reflects the agents confidence in its own belief about the trend, and the willingness of the agent to trade at sub-optimal prices in order to obtain or sell off shares.

Trading frequency Once the chartist has detected a trend, it will continue trading for as long as the trend persists. The parameter W_c^j limits the frequency with which the chartist can submitting orders by specifying the number of round that the agent waits before submitting another order.

4.3.3 Strategy evaluation

In round t_T, the agent calculates the moving average of the buy and sell prices by subtracting the prices that are no longer inside the windows specified by H_c, and adding the the most recent prices. The agent then calculates the average mid-price $\bar{M}_{\text{tot}}$which is used in the trend detection.

$$\bar{M}_{\text{tot}} = \frac{1}{2H_c}\left[\sum_{n=t_p}^{t_T}(A_n + B_n) - \sum_{n=t_p-H_c}^{t_T-H_c}(A_n + B_n)\right] \tag{8}$$

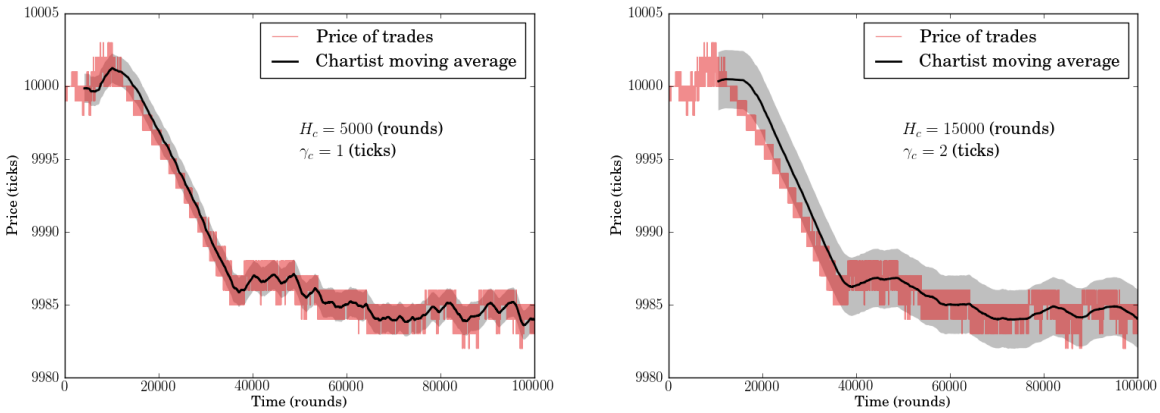
where t_p is the latest round in which the agent was previously active. The agent calculates the current mid-price m:

$$m = \frac{A_T + B_T}{2} \tag{9}$$

The agent then uses a simple rule to decide whether to sell or buy, and at what price:

Place a buy order if	$m > \bar{M}_{\text{tot}} + \gamma_c$	at buy price	$p = m + A_c$	(10)
Place a sell order if	$m < \bar{M}_{\text{tot}} - \gamma_c$	at sell price	$p = m - A_c$	(11)

Figure 4 provides two examples of how the sensitivity parameter γ_c and the parameter for the time horizon of the chartist strategy γ_c influences when the agent will trade or not. In the left figure, the agent has a low sensitivity to price changes, as indicated by the narrow area shaded gray around the black line indicating the moving average calculated by the agent. Since $\gamma_c = 1$, the traded price only has to deviate by a single tick from the moving average before the agent will submit orders to trade. In this case, the agent uses a moving average calculated over a relatively short span of $\gamma_c = 5000$ rounds, which makes the moving average reflect short-lived price trends. As the figure shows, these parameters prompt the agent to detect trends more or less constantly, and the agent even detects a short up-trend in the beginning of the simulation, as is shown where the traded price (red) does not stay within the gray shaded area . In the right figure, the agent has a sensitivity of $\gamma_c = 2$ and a longer time horizon of $\gamma_c = 15000$ rounds. Consequently, the agent detects a downtrend in round 12000 or so, where the traded prices, and begins to submit sell orders. However, just before round 40000, the agent decides that the downtrend has ended and the agent becomes inactive until the next time a trend is detected.



(a) (b)

Figure 4: Examples illustrating the chartist decision strategy

Part III

Exploring market behavior with inverse simulation

This part will cover the methods used in trying to link the various model parameters to different kinds of market behavior. The model has several parameters which must be selected carefully before the simulation can be used to infer knowledge about market behavior. The main focus is on the influence of the fast trader latency parameters.

The model must be calibrated such that it mimics the behavior of real markets. Since virtually every aspect of the simulation behavior depends on the values on the parameters, these must be chosen carefully in order for the simulation to produce "realistic" behavior, i.e., behavior that have could be found to occur in real markets. The way in which this was accomplishes in this work can be divided into four rough stages, which are outlined in the next section.

The process of selecting parameters turned out to consume a significant amount of time, and simply using a genetic algorithm to optimize over the entire space of parameters was not enough. Instead, the process was a slow and iterative one of running the genetic algorithm to create a data set, analyze the data set to find out what was discovered in the search, and then run the genetic algorithm again with different parameters. Thus several data sets were created, each with the purpose of examining some aspect of the simulation, or of the parameter tuning method itself.

5 Overall procedure

The selection of parameters is a fairly complicated process because of the large parameter space, and because it takes a significant time to evaluate the fitness of a given set of parameters[3]. amount of time to execute a simulation. Because of this, the following three-step parameter selection procedure was used.

Model calibration Fix some of the model parameters in order to reduce the search space for the optimization algorithm. This requires us to consider which parameters can be fixed without losing opportunity to gain insight into market behavior. Essentially this step is a question of prioritizing the optimization of some parameters over the optimization of others.

Inverse simulation Use an optimization algorithm to find sets of parameters which yield realistic model behavior. A genetic algorithm was chosen for this purpose, and the details are explained in section 7.Section 7.3 will explain how the behavior of the model was summarized into a few simple performance measures that were used in the inverse simulation.

Prepare data for analysis There might be several different parameter configurations which produce seemingly realistic behavior, but do not correspond to a realistic market setting. An example for this is given in figure 6a, where the market contained no fast traders. Another scenario is that the market behaves in a way which can safely be said not to correspond to a realistic market. Such an example is given in figure 6b (see section 7.5). Finally, some data points might satisfy both of the two above requirements, but still cause problems for some of the data analysis techniques that were utilized (see section 9.2.2).

Analyze data From the set of parameter combinations found by the optimization algorithm, remove the parameter combinations which obviously do not correspond to a realistic setting. After this, construct data matrices from the parameters of the remaining individuals and their fitness values, as described in section 8.2. These data can be analyzed, after appropriate preprocessing techniques have been applied (see section 9). The purpose of the data the analysis is to find parameters which are likely to cause certain desirable (or undesirable) behavior in the market. For instance, we might be interested in determining which parameters causes the traded price to stabilize faster after a shock to the fundamental price.

6 Model calibration

Setting some parameters to fixed values can be thought of as calibrating the model. Naturally, fixing any parameter implies that assumptions about the way in which the parameters affect the model behavior must be made. Furthermore, fixing some parameters

[3]The calculation time depends largely on the parameters, such as the number of agents and how active these are. Typically one to several minutes are required to evaluate a single set of parameters.

may obscure the presence of some interesting market behavior that requires the parameters to have values different from the fixed ones. Unless the model is simple enough to allow for all parameters to be searched, these simplifications are necessary in order to explore the impact of the parameters of most central interest to the research question. For instance, the volume of the orders that the agents submit probably affect how the market responds in different situations, but since the order volumes are not related to the latency of the traders, the parameters have been fixed.

Number of rounds Due to the computational cost of running the simulation for a large number of rounds, the the number of rounds is fixed at 10^5 for all experiments.

HFT order volumes When the HFT agents are initialized in the beginning of the simulation, each agent is assigned with a constant, sampled from a normal distribution, designating the fixed volume of the orders that the agent will submit throughout the simulation. Hence, market maker MM_i will submit orders with a volume of V_m^i, where $V_m^i \sim \mathcal{N}(V_{m,\mu}, V_{m,\sigma})$, and chartist SC_j will submit orders with volume V_c^j, where $V_c^j \sim \mathcal{N}(V_{c,\mu}, V_{c,\sigma})$. Each HFT agent submit orders with a volume that is constant throughout the simulation.

Average arrival rate of slow trader orders

Slow trader parameters The rate of slow trader order arrivals is fixed to

Agent start portfolio All HFT traders start with a fixed amount of cash and a fixed number of shares in their portfolio.

Order book initialization In all simulations, the order book is initialized with $10 * *5$ orders with prices sampled from the distribution $\mathcal{N}(F_0, 10 * *2)$ and volumes sampled from $\mathcal{N}(20, 5)$.

The remaining model parameters will either be fixed for each experiment, or varied by the genetic algorithm. Table 6 provide an overview of the settings of each experiment.

7 Inverse simulation with a genetic algorithm

Inverse simulation refers to the technique of specifying metrics measuring model behavior, and then using an optimization algorithm to search for parameters resulting in desirable (or undesirable) behavior. Simply put, inverse simulation is a way of transforming the problem of finding appropriate model parameters into an optimization. By defining functions that measure the quality (or fitness) of a simulation, numerical optimization algorithms can be used for searching for desired model behavior. In this work, a genetic algorithm was used to search the parameter space. The algorithm proceeds as explained below.

1. Generate a population of healthy individuals, e.g., individuals with valid parameters.

2. Evaluate fitness for every individual in the population.

	f_o	f_s	f_σ	f_t
Ideal market with a fast response time little overshoot and non-flickering prices	$\min(f_\mathrm{o})$	$\min(f_\mathrm{s})$	$\min(f_\sigma)$	$\min(f_\mathrm{t})$
Crashing market	$\max(f_\mathrm{o})$	-	-	-
Market that stabilizes at an undervalued price x	$\min\|f_\mathrm{o} - x\|$	-	$\min(f_\sigma)$	-

Table 2: Optimization criteria for different types of market behavior

3. Repeat n_gen times

   (a) Generate offspring by crossing existing individuals.

   (b) Apply mutation to with a certain probability to each individual (parents as well as children)

   (c) Evaluate fitness of children and mutated parents.

Mutation and crossover are the operators responsible for generating variation in the population, while the selection is responsible for propagating promising individuals to future generation where they might be improved. Several possible methods of performing each of these three steps exist in the literature (see[59], [16]), and section 7.4 briefly covers the method and parameters of the genetic algorithm.

7.1 Optimizing towards desired behavior

The four fitness measures (section 7.3) make it possible to specify the type of market behavior that is likely to be selected by the genetic algorithm. This can be accomplished by specifying which fitness values should be minimized, and which should be maximized. Table 2 suggests a few such strategies. In this work, the focus was on identifying markets such as is described in the first row of the table.

7.2 Scaling genes into parameters

Since the choice of mutation and crossover operators depends on the nature of the genes, the first step towards utilizing the genetic algorithm to search the model parameters is to decide on how to encode the parameters as individuals. A set of parameters is represented by an individual, I, consisting gene for each parameter, represented by a floating point $\mathcal{G}_{I,p}$, where p denotes the index of the parameter. When the population is initialized, each $\mathcal{G}_{i,p}$ sampled from a uniform distribution:

$$\mathcal{G}_{I,p} \sim \mathcal{U}(0,1) \qquad \forall i \in \mathcal{I} \wedge p \in \mathcal{P} \tag{12}$$

where $\mathcal{I}$ denotes the set of initial set of individuals, and $\mathcal{P}$ denotes the set of parameters included in each individual. The sampled values are then scaled into the appropriate ranges by multiplying by constants which are set equal to the desired expected value of each parameter when the genetic algorithm is initialized:

$$P = \left\lceil \mathcal{G}_{I,p} \cdot \mathbb{E}[p] \right\rfloor \qquad \forall i \in \mathcal{I} \wedge p \in \mathcal{P} \tag{13}$$

where $\lceil \cdot \rfloor$ denotes the function that rounds to nearest integer.

7.3 Quantifying model behavior

XXX NOT FINISHED YET. In order to use inverse simulation, it is necessary to decide on how to measure the quality of an instance of the simulation. In this work, the overall goal is to examine which parameter values cause the market to be stable, and which cause it to be unstable. In order to do this, four fitness measures were defined as below. Each of the fitness measures are designed to reflect different properties of the simulation, but some correlation between the measures were found. This is discussed in section 10.1.

7.3.1 Market response time

After the negative shock to the fundamental price, the slow fundamentalists will start submitting orders at lower prices, thus creating a force that drives that traded price towards the new price. The number of rounds that elapse before the asset is traded at the new fundamental can be thought of as the response time of the market and this property of the market is interesting for a few reasons. If a market has a slow response time, it means that the asset is traded for more than it is worth for a longer of period of time. The response time is denoted f_{t} and is calculated as follows:

$$f_{\mathrm{t}} = \arg\min_{T} \left(p^t_{\min}(T) \right), \ \forall n > t_{T_\eta} \tag{14}$$

where $p^t_{\min}(T)$ is the price of the order traded at the lowest price in round n:

$$p^t_{\min}(n) = \min \left(p^t_1(n), \ p^t_2(n), \ldots, \ p^t_K(n) \right) \tag{15}$$

when there are K trades in round n. If the trade price never reaches the new fundamental, f_{t} is undefined.

7.3.2 Market overshoot

In some markets the traded price never becomes smaller than η, while in other markets the share price overshoots the new fundamental by being traded at prices lower than η. In the case that $\eta < 0$, the overshoot reflects an under-valuation of the stock.

The market overshoot is calculated as the number of ticks between the traded price and the fundamental price, in any of the rounds following the round in which the stock is traded at price $p^t_i(n) = F_\eta$ for the first time, denoted t_F:

$$f_{\mathrm{o}} = | \max p^t_i(n) |, \ \forall i \wedge \ n > t_F \tag{16}$$

f_{o} does not measure the time duration of the under-evaluation.

7.3.3 Price flickering

This fitness measure reflects how much the traded price fluctuates by calculating the standard deviation of the prices of all trades executed after t_F:

$$f_\sigma = \sum_{n=t_F}^{\mathtt{N_r}} \sum_{i=1}^{N_n} \frac{\left(\bar{p}_n - p_i^t(n)\right)^2}{N_n} \tag{17}$$

where N_j is the number of transactions in round j, and $\bar{p}_n$ is the average traded price in round n. One problem with the definition of f_σ is that it assumes that the distribution of prices is stationary after the stock has first been traded at the new fundamental. This assumption holds for markets in which the traded price stabilizes around a constant mean, but does not hold when the market crashes, as the mean statistic will constantly change.

7.3.4 Time to stabilize

The margin of two ticks sets a fairly strict requirement for when a simulation is considered stable, as a flicker of just a few ticks

$$\arg\min \forall \tag{18}$$

Figure 5illustrated how the four fitness measures are calculated, while figure XXX gives four examples and lists the respective fitness values calculated for each simulation.

7.3.5 Examples of typical market behavior

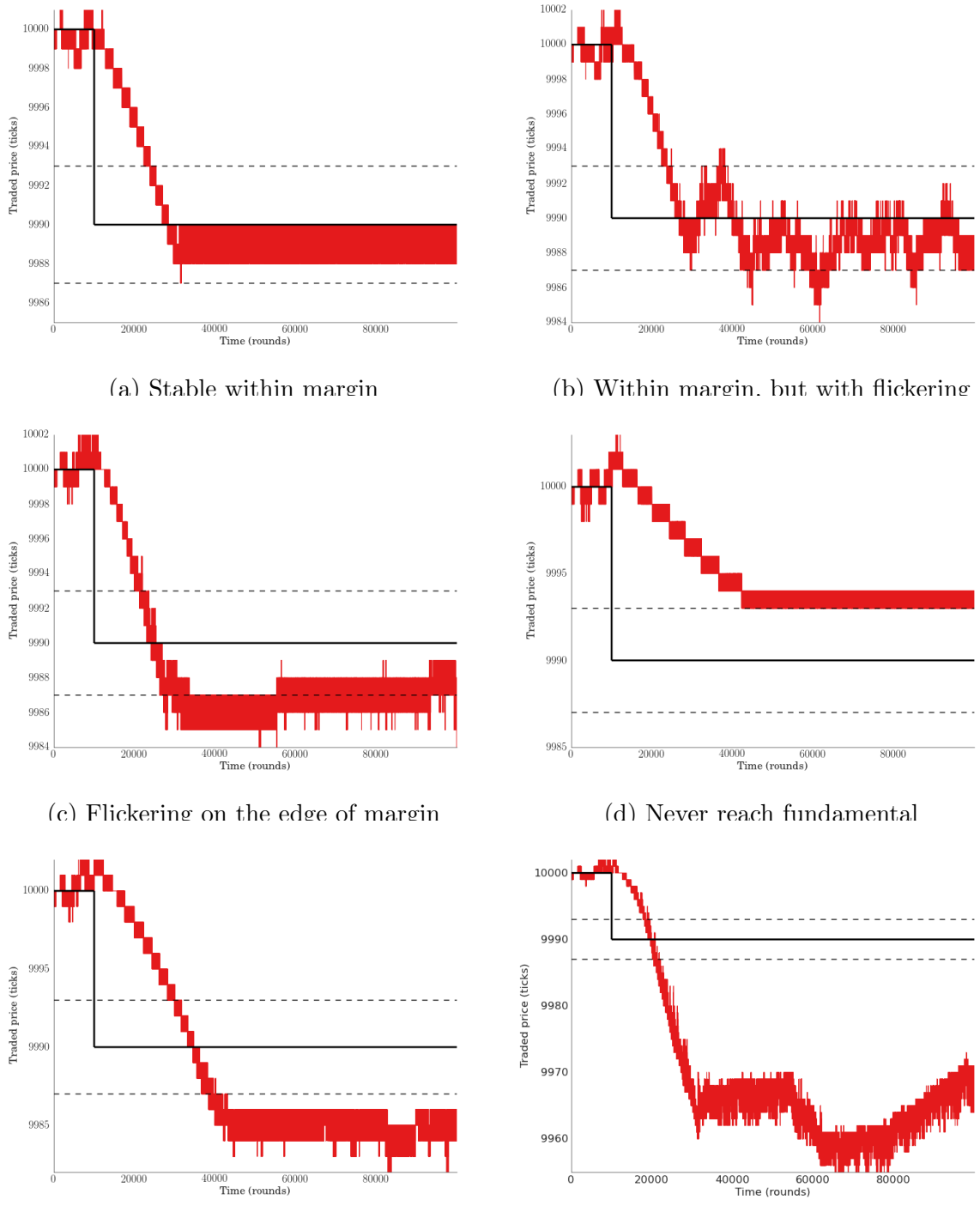
To give the reader a more intuitive idea of how the fitness measures summarize the model behavior, figure 5 gives six examples of some fairly common patterns.

7.4 Configuring the genetic algorithm

Although this is a basic version of genetic algorithm, using it correctly is not necessarily easy, as was encountered. First of all, the parameters for the genetic algorithm itself must be established. The larger and more complex the search space, the more resources the search will require, since the evaluating the fitness function (i.e., the running the simulation) will have to be performed a larger number of times. Table 3 presents an overview of the parameters used in the genetic algorithm. The tournament size was set to three, as this provided the best balance between diversification and speed of convergence [9, 25], considering the high computational of the model.

7.5 Filtering parameters

The model parameters do influence the fitness values as they control model behavior, but they are not directly weighted into the fitness-values. This means that even after the parameters of the genetic algorithm was properly tuned in such a way that higher-fitness



(a) Stable within margin

(b) Within margin, but with flickering

(c) Flickering on the edge of margin

(d) Never reach fundamental

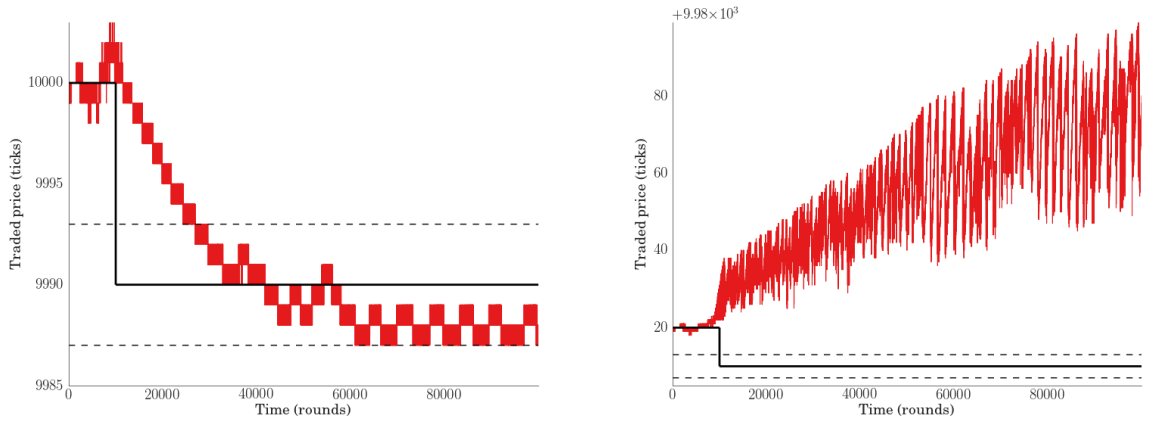
(e) Stabilize under fundamental

(f) Market crash

Figure 5: Six examples of typical market dynamics

Parameter	Assignment
Number of individuals	200
Cross-over points	2
Tournament size	3
Mutation probability	0.1
Mutation distribution	$\mathcal{N}(\mu = 0, \sigma = 0.1)$

Table 3: Overview of parameters used in the genetic algorithm



(a) Market with unrealistic parameters (no fast traders) with seemingly realistic dynamics

(b) Parameters causing unrealistic dynamics

Figure 6: Two examples of why model parameters must be restricted

individuals were produced, these individuals often turned out to be of no interest. Such individuals were discarded according to the filtering criteria described in this section.

As mentioned earlier, it is not enough simply to define a fitness function which assigns high values to parameters causing realistic behavior. In addition, it is important to discard parameters which obviously do not correspond to a realistic setting. Imagine that the simulation scores high fitness values when executed without any market makers. Since it is known that real markets do in fact contain market makers, nothing can be inferred from such a result. Indeed this might be a consequence of poorly designed fitness measures, but since it is easier to use domain specific knowledge to filter out the unrealistic parameters

The first graph is an example of a simulation which is assigned fairly good fitness values, but which was executed with clearly unrealistic parameters: $N_m = N_c = 0$. The simulation reaches the new fundamental price fairly quickly without any undershoot, and stays within the stability margin. The only point where it scores badly is the standard deviation which is slightly high due to the fluctuating trade price.

8 Applying the genetic algorithm

Applying the genetic algorithm to produce markets with desirable behavior turned out to be more difficult that one could have hoped for. First of all, the high computational costs was a hurdle. The high time complexity stems from several factors

- Running a simulation for a given set of parameters required up to several minutes of computation on a single CPU core.
- Large number of model parameters increase the size of the search space. The more parameters Unfortunately there is no magic to the way the genetic algorithm works, and optimizing in a larger search means a larger time complexity.
- Large range of parameters. Some of the parameters are integers while some are real numbers. While most of the parameters have a lower bound, none of the parameters have upper bounds.
- Unstable fitness parameters. Since the same set of parameters can produce varying model behavior, the fitness-values may also vary. Therefore it is necessary to evaluate the simulation several times for each set of parameters.
- The model parameters also influence the time complexity. For instance, evaluating a simulation with many agents takes more time than evaluating a simulation with fewer agents.

Genetic algorithms are naturally suited for parallel computation. Two servers with a total of 40 cores and enough memory to evaluate as many simulations were utilized. With this equipment, evaluating a single strategy (see section 8.1)for generating data sets could take up to several weeks.

8.1 Experiments: Dividing the search into parts

As mentioned earlier, the number of parameters and the range of each parameter influences the complexity of the search. Because of this, it is desirable to keep the number of parameters that are included in each individual as small as possible. However, fixing parameters means that some interesting properties about the model might not be discovered. Furthermore, varying all parameters at the same time makes the analysis and interpretation of the results more difficult. In an attempt to overcome this dilemma, several "experiments" were carried out [4]. Instead of trying to optimize all the model parameters at once, the search was split into several parts, each of which we call an experiment. Each of these experiments produce a data set, each of which were analyzed using the methods described in section 9. Some of the data sets produced interesting results, while others did not. Part IV focuses on the analysis and presents the findings. A brief overview of the experiments is presented in 10, and the next section will explain exactly what a data set is.

[4]The reason for the quotes is that the term experiment might be stretching the common understanding of what an experiment is a little.

	$\lambda_{c,\mu}$	$\lambda_{c,\sigma}$	N_c	$\tau_{c,\mu}$	$\tau_{c,\sigma}$	$H_{c,\mu}$	$H_{c,\sigma}$	$W_{c,\mu}$	$W_{c,\sigma}$	$\lambda_{m,\mu}$	$\lambda_{m,\sigma}$	N_m	$\tau_{m,\mu}$	$\tau_{m,\sigma}$
0	84	11	14	98	9	1071	445	38	17	3	2	48	8	1
1	23	21	74	49	24	529	554	45	13	9	0	8	5	3
2	51	13	53	47	13	3586	536	10	11	9	4	14	4	2
3	18	21	213	70	39	793	1179	33	15	7	2	43	6	3
4	94	41	144	10	25	2668	893	12	15	6	1	49	7	4
5	19	4	130	15	38	1085	1165	39	4	2	3	11	4	4
6	65	15	91	81	46	3867	1991	48	1	7	2	21	4	4
7	36	38	143	77	19	2805	1870	10	9	7	0	3	2	4
8	43	8	10	19	19	3384	1706	33	4	8	4	5	5	0
9	11	33	127	94	49	3597	723	12	2	7	1	33	5	4

Table 4: An example data matrix containing the parameters of ten individuals who lived sometime during the execution of the genetic algorithm. In this case, each individual contained parameters for the number of HFT agents, as well as the latency and thinking time parameters. Hence, the data matrix has a column for each parameter.

8.2 Gene pool as data set

The previous sections contain the details of each of the steps undertaken in order to produce data sets. To summarize, the list below enumerates the steps.

1. Initialize a population in the genetic algorithm with healthy individuals.
2. Evaluate the fitness for every individual several times and obtain fitness-values by calculating averages.
3. Stack all individuals that ever lived into a $N \times \mathcal{L}_i$ parameter data matrix $\boldsymbol{P}$, where N is the number of individuals, and $\mathcal{L}_i$ is the length of each individual. Likewise, stack the fitness values into a $N \times \mathcal{L}_f$ fitness-data matrix $\boldsymbol{F}$, where $\mathcal{L}_f$is the number of fitness values calculated.
4. Filter the data by removing rows in $\boldsymbol{P}$with parameters which can be deemed not to correspond to real markets, and by removing rows in $\boldsymbol{F}$with fitness values that are not realistic. Please refer to section 7.5 for details. Naturally, when a row is removed in $\boldsymbol{P}$, it is also removed in $\boldsymbol{F}$, and vice versa.
5. Likewise, data points which were generated by a simulation crashing before it could complete were removed.

Tables 4 and 5 contain the first rows of $\boldsymbol{P}$and $\boldsymbol{F}$for one of the data sets.

9 Data analysis techniques

Using inverse simulation creates a lot of data, which is aggregated into a data matrix as described in the previous section. This data has to be analyzed in order to obtain

	f_{o}	f_{s}	f_{σ}	f_{t}
0	3	25359	0.382092	29838
1	7	99999	1.289659	23373
2	6	99999	1.253363	18748
3	7	99997	1.695150	22819
4	6	94343	1.329276	22703
5	16	99999	2.439084	31860
6	6	93378	1.287235	25645
7	10	99997	1.858166	19417
8	3	24039	0.935465	27381
9	19	99995	4.092439	24845

Table 5: This table contains the fitness values for each individual in table 4. Note that, in order to increase the reliability of the fitness measure of an individual, the recorded fitness-values are the average of the fitness-values obtained by evaluating each individual ten times

insight into how the various parameters, especially those concerning the latency of the fast traders, influence the market.

9.1 Clustering algorithms

Clustering methods were used to obtain representative nodes in both the fitness space and the parameter space. Clusters in $\boldsymbol{F}$ partitioning the fitness space can be thought of as representing markets with different types of behavior. Likewise, clusters in $\boldsymbol{P}$ represent markets in which various conditions are present. Ideally, there is a large overlap between the assignment of data points to the clusters in the fitness space and to clusters in the parameters space. Such overlaps will signify that markets in which certain conditions are present (such as a large number of fast market makers) will tend to have certain behaviors, and vice versa. $\boldsymbol{P}$ can be thought of as a data matrix containing features, while $\boldsymbol{F}$ can be thought of as a data matrix containing target variables.

The benefit of this method over trying to train a regression model is that it is easier to interpret how parameters are mapped to fitness values by the model. Training, say, a neural network might indeed enable us to make predictions of the market behavior given a set of parameters. However, the goal of this thesis is not to make predictions of market behavior within the model, but rather to understand how certain parameters influence the market behavior. Hence there is no need to employ modeling techniques that are more complex that they have to be.

Three different clustering techniques were used in this work. The motivation for employing each of these techniques is described below. All three clustering algorithms are implemented as part of the `scikit-learn` toolkit[44].

K-means clustering (KM) K-means clustering is probably the most widely used clus-

tering algorithm, and performs well complexity-wise even on reasonably large data sets. The algorithm requires the specification of the number of clusters, which can be a disadvantage. K-means clustering minimizes an objective function (or distortion measure [8]) calculated as the sum of squared distances between the points and their assigned clusters. It is important to note that the assignment of points to clusters is not guaranteed to yield the global minimum of the objective function [30]. The clustering was therefore run several times, and the assignment with the lowest distortion is used. In order to speed up the calculation, the Mini-Batch version of the clustering algorithm was used [52].

Affinity propagation This algorithm has the benefit that it does not require the specification of a number of clusters [22]. Since the algorithm has problems dealing with large data sets, K-means clustering was used to reduce the number of data points by using 1000 clusters as representatives of the data.

Gaussian mixture model (GMM) The Gaussian mixture model is another incredibly popular clustering algorithm since it is both fast and can calculate complex decision boundaries[5]. When the model is allowed to a full covariance matrix, the number of parameters scales quadratically with the dimensionality of the data set, and this might lead to gross over-fitting and the well-known problem of some of the clusters having variances approaching zero. However, as $\boldsymbol{F}$ has only four dimensions and several hundreds of thousands of data points, the GMM can be safely allow to have a full covariance matrix. The GMM was fitted used the EM-algorithm [7, 8].

The best algorithm for dealing with the problem at hand turned out to be the GMM algorithm, for reasons which are discussed in section 13.2. All three clustering algorithms are sensitive to outliers, and the next section discussed the two methods used for dealing with such data points.

9.2 Preprocessing

Each experiment generated two data matrices, $\boldsymbol{F}$ and $\boldsymbol{P}$. The dimensionality of $\boldsymbol{F}$ is always equal to four, since only four fitness measures were defined, whereas the dimensionality of $\boldsymbol{P}$ depends on how many parameters were included in the individuals of the genetic algorithm. Hence, in order to visualize how the data is distributed, the dimensionality must be reduced. Furthermore, since the data set contained abnormal data points, these had to be identified and removed.

9.2.1 Normalization and dimensionality reduction

The three clustering algorithms utilized all have problems with data sparsity. However, since $\boldsymbol{F}$ only has four features, the data set poses no challenge to the clustering algorithms used. Instead, dimensionality reduction was used for the benefit of visualization. Principal

[5]Another delicious property of the model is that it is generative, although this property was not used in this work

component analysis was chosen. As the four fitness measures had very different ranges, the data in $\boldsymbol{F}$ was normalized by subtracting the mean and dividing by the standard deviation, as in [51].

9.2.2 Identifying outliers

The term outliers is often used as a label for data which is considered "invalid" in the sense that is not a product of the true data generation process, but has been corrupted by various sources of noise. In this report, data that was caused by failing simulations corresponds to the usual understanding of outliers. However, data from such simulations are never included in $\boldsymbol{P}$ and $\boldsymbol{F}$ to begin with. Instead, outliers in this report refer to entries in $\boldsymbol{P}$ and $\boldsymbol{F}$ deviate significantly from the majority of the data points by having extreme values.

Such data points caused problems when applying data analysis techniques which rely on a fairly normal distribution of the data points, such as Principal Component Analysis (PCA). PCA looks for a rotation of the data space, such that the axes of the new basis are aligned with the directions of the largest variance in the original space. Since a few points with extreme values come to account for a large portion of the data set variance, the new basis computed by PCA will be aligned along these few data points. When PCA is used to extract lower dimensional features from the data set, outliers will degrade the quality of these features. In the case that PCA is used for data visualization, the scatter plots of the first few principal components will not be very informative, as they merely show the projection of the data onto the axes aligned with the outliers.

In all experiments, outliers were present in both the parameters space and in the fitness space. Outliers in the parameter space occur because of abnormally large mutations, and any dimension of the parameter space is susceptible to such an event. In the fitness space, only a few of the features suffered from the occurrence of outliers. The feature f_{s}cannot contain outliers, because the shock to the fundamental occurs at round $t_{=}10^4$, and because the simulation is terminated at round $t_{=}10^5$, hence $f_{\mathrm{s}} \in [10^4, 10^4 + 1, \ldots 10^5]$. The same is true for f_{t}. $f_{\sigma}$and f_{o}, on the other hand, are susceptible to outliers, because the two features have no upper bound. Note that markets with abnormally large overshoot are interesting in themselves, as something caused these markets to crash. However, since the clustering algorithms are sensitive to the presence of data points that deviate significantly from the majority of data, such data points are removed.

1. Apply a monotonically increasing transformation $f(x)$ to some or all of the features for every data point in the data set. A common choice of f is $f(x) = \log x$, as it efficiently reduces the impact of data points with extremely high values. The log-scaling does a fairly good job of reducing the importance of the outliers in the f_{σ}feature, while the change is less dramatic in f_{t}. While the left scatter plot does not really reveal any structure of the data, the transformation makes it possible to spot to rough clusters when inspecting the right plot.

2. Manually select one or more criteria for when a data point is to be considered an outlier.

Part IV
Experiments

As mentioned in the previous chapter, the process of finding was an iterative one of running an experiment, analyzing the generated data, draw conclusions and then repeat the steps with a new experiments designed to amend the mistakes of the previous experiment. This chapter will go through the results that were obtained from each of the data sets. A summary and discussion of the results is found in chapter V.

10 Overview of experiments

The model has many parameters, and doing an exhaustive search over the entire parameter space is not possible. Instead, a genetic algorithm (GA) was used to do targeted searches of the parameters. Depending on how the GA was set up, different areas of the search space was searched. Even when using a GA, some parameters had to be fixed. However, fixing parameters means that the effect of the fixed parameter on the behavior of the model remains unknown, since only a subspace of the entire parameter space is searched. For this reason the genetic algorithm was executed several times, each creating a data set containing fitness values for different parts of the parameter space. Each of these data set can be analyzed, providing information which can be corroborated in order to form an understanding of the overall market behavior.

Table 6 contains an overview of the different data sets, showing which parameters were fixed, and which were included as genes in the genetic algorithm.

In the following four experiments, the genetic algorithm was set to minimize all four fitness-measures.

$\mathcal{D}_3$**: Varying the number of HFT agents, and all latency related parameters** This data set was generated by including all the model parameters concerning time latency as well as the number of agents into the individuals in the genetic algorithm. Due to the high number of variables, the data turned out to be difficult to analyze, as too many factors pertaining to the simultaneous change of several parameters influenced the fitness values.

$\mathcal{D}_9$**: Fixing the number of agents while varying latency parameters** The analysis of $\mathcal{D}_3$ showed that when minimizing the four fitness-measures, the genetic algorithm tended to select model containing few or no HFT agents. The case of a market with no market makers and no chartists can safely be said to be trivial. Hence, in experiment $\mathcal{D}_9$, the number of HFT agents were fixed to $N_m = 30$ and $N_c = 100$.

$\mathcal{D}_{10}$**: Fixing the number of HFT chartists** Since $\mathcal{D}_9$ kept N_m and N_c constant, the experiment did not reveal anything on how the market behavior changes when the number of agents changes. In order to investigate the impact of having many or few HFT market makers, N_m was varied in this experiment. Although it is also of

ID	Description	Fixed parameters	As genes
$\mathcal{D}_3$	All parameters varied	$V_{c,\mu} = 10$,$V_{c,\sigma} = 3$,$V_{m,\mu} = 50$,$V_{m,\sigma} = 20$,$\gamma_{c,\mu} = 2$,$\gamma_{c,\sigma} = 5$,$\alpha_{c,\mu} = 3$,$\alpha_{c,\sigma} = 2$	$\lambda_{c,\mu}$, $\lambda_{c,\sigma}$, N_c, $\tau_{c,\mu}$, $\tau_{c,\sigma}$, $H_{c,\mu}$, $H_{c,\sigma}$, $W_{c,\mu}$, $W_{c,\sigma}$, $\lambda_{m,\mu}$, $\lambda_{m,\sigma}$, N_m, $\tau_{m,\mu}$, $\tau_{m,\sigma}$
$\mathcal{D}_9$	Fixed number of HFT agents	$N_m = 30$, $N_c = 100$,$V_{c,\mu} = 10$,$V_{c,\sigma} = 3$,$V_{m,\mu} = 50$,$V_{m,\sigma} = 20$,$\gamma_{c,\mu} = 2$,$\gamma_{c,\sigma} = 5$,$\alpha_{c,\mu} = 3$, $\alpha_{c,\sigma} = 2$	$\tau_{c,\mu}$, $\tau_{c,\sigma}$, $H_{c,\mu}$, $H_{c,\sigma}$, $W_{c,\mu}$, $W_{c,\sigma}$, $\lambda_{m,\mu}$, $\lambda_{m,\sigma}$, N_m, $\tau_{m,\mu}$, $\tau_{m,\sigma}$
$\mathcal{D}_{10}$	Fixed number of HFT chartists and fixed strategy parameters	$N_c = 150$, $\tau_{m,\mu} = \tau_{c,\mu} = 50$, $\tau_{m,\sigma} = \tau_{c,\sigma} = 20$, $H_{c,\mu} = 5000$, $H_{c,\sigma} = 2000$, $W_{c,\mu} = 50$, $W_{c,\sigma} = 20$,$V_{c,\mu} = 10$,$V_{c,\sigma} = 3$,$V_{m,\mu} = 50$,$V_{m,\sigma} = 20$,$\gamma_{c,\mu} = 2$,$\gamma_{c,\sigma} = 5$, $\alpha_{c,\mu} = 3$,$\alpha_{c,\sigma} = 2$	N_m, $\lambda_{c,\mu}$, $\lambda_{c,\sigma}$, $\lambda_{m,\mu}$, $\lambda_{m,\sigma}$
$\mathcal{D}_{11}$	Fixed number of HFT market makers and fixed strategy parameters	$N_m = 52$, $\tau_{m,\mu} = \tau_{c,\mu} = 50$, $\tau_{m,\sigma} = \tau_{c,\sigma} = 20$, $H_{c,\mu} = 5000$, $H_{c,\sigma} = 2000$, $W_{c,\mu} = 50$, $W_{c,\sigma} = 20$,$V_{c,\mu} = 10$,$V_{c,\sigma} = 3$,$V_{m,\mu} = 50$,$V_{m,\sigma} = 20$,$\gamma_{c,\mu} = 2$,$\gamma_{c,\sigma} = 5$, $\alpha_{c,\mu} = 3$,$\alpha_{c,\sigma} = 2$	N_m, $\lambda_{c,\mu}$, $\lambda_{c,\sigma}$, $\lambda_{m,\mu}$, $\lambda_{m,\sigma}$

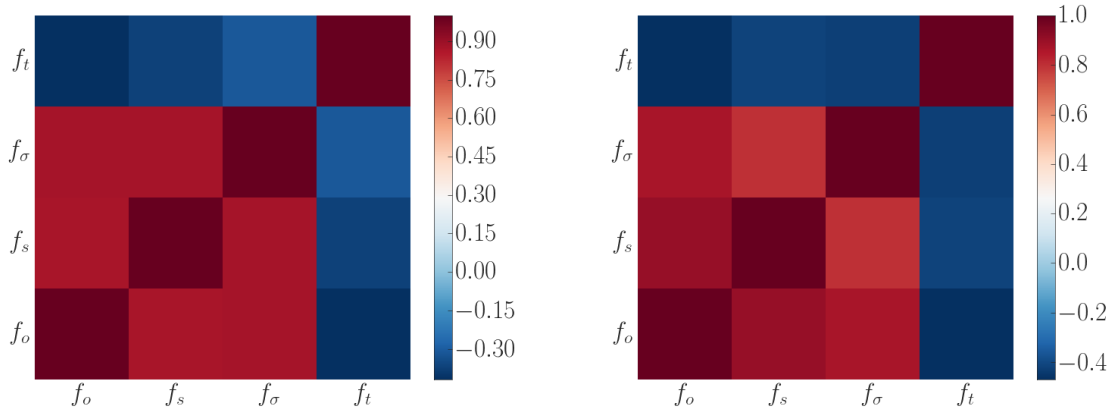
Table 6: Overview of datasets

interest how the market behavior depends on the number of HFT chartists, including N_c as a gene would yield results similar to those obtained in $\mathcal{D}_3$. For this reason the number of HFT chartists was fixed to $N_c = 150$.

$\mathcal{D}_{11}$**: Fixing the number of HFT market makers** This experiment was carried out in order to investigate the impact of the number of HFT chartists on the market behavior, and is supplementary to $\mathcal{D}_{10}$.

10.1 Correlation between fitness measures

A factor which influences the evolution of parameters is correlation between the fitness-measures. If two or more fitness measures have non-negative correlation coefficients, individuals will be statistically more likely to get good scores in the correlated fitness measures at the same time. Since all fitness measures are given equal weight in the selection process, individuals scoring well in the correlated fitness-measures will win over individuals which score well on another, statistically independent fitness measure. It is therefore important to compare the selection tendencies with the correlation between fitness-measures. Figure



(a) $\mathcal{D}_{10}$ (b) $\mathcal{D}_{11}$

Figure 7: Correlation between model fitness measures

7 shows a plot of the correlation matrix for $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$. Since later generations will be affected by the biased selection and therefore contain more individuals which did well on the correlated fitness measures, the correlation coefficients in the figure were calculated over individuals in the first generation only.

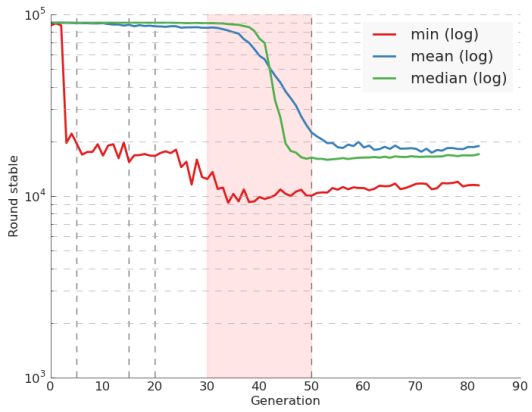
For instance, the correlation between f_o and f_σ means that an individual which scores a good f_o-fitness will be statistically likely to also score a good f_σ-fitness. Since all four fitness measures are weighed evenly in the selection, models with behavior which is assigned good values for f_o and f_σ will score a better overall fitness than a simulation with a good f_t-fitness. In other words, the correlation between f_σ and f_o means that stable individuals will outlive fast individuals as they are selected for breeding more often. This is not a property of the model itself, but something that arises due to the definition of the fitness measures.

The correlations also speak of trade-offs between speed and stability in the model. For instance, a negative correlation between f_o and f_t means that the model is statistically likely to have a large overshoot when it responds quickly to the change in the fundamental.

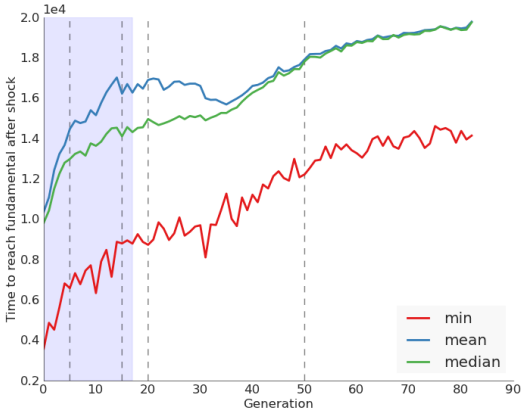
11 Fitness and parameter evolution

11.1 Variable number of market makers

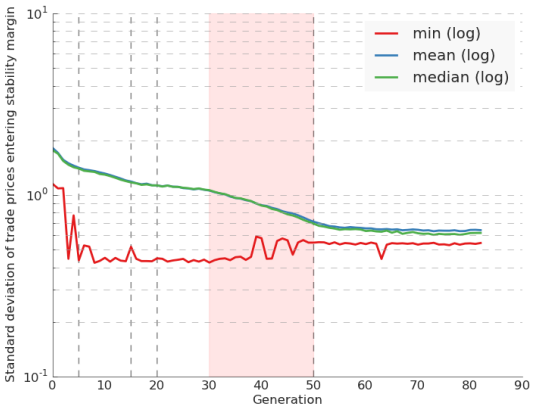
Figure 8 shows the evolution of the four fitness measures. The population wide mean is plotted along the median and minimum statistics. Since all four fitness measures were minimized, the curve for the minimum value shows the best individual alive during each generation, with respect to each fitness measure. While the mean reflects how the overall population is evolving, the median is useful as it gives an insight into how skewed the population wide distribution of parameters is.



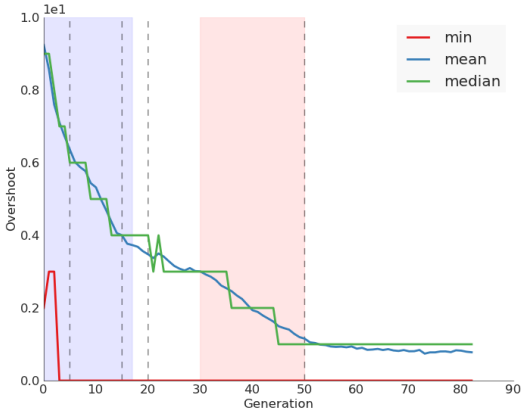
(a) Evolution of f_s



(b) Evolution of f_t



(c) Evolution of f_σ



(d) Evolution of f_o

Figure 8: Evolution of the four fitness measures in experiment $\mathcal{D}_{10}$

Model stability Figure 8a: shows that the GA quickly manages to find some parameters which cause the simulation to stabilize quickly. However, these individuals do not manage to dominate the population evident by the mean and median curves remaining almost the same until generation 30 or so. In the next 20 generations the population undergoes a rapid change, as the population wide average of f_{s} drops from close to 10^5 to around $2 \cdot 10^4$ rounds on average. The disparity between the mean and the median indicates that the population undergoes a rapid change in the same period, from mostly containing unstable individuals to mostly containing stable individuals. In generation 42, the median curve crosses the mean curve, which means that the the population contain as many stable simulations as it contain unstable simulations. From that point on the unstable simulations are quickly replaced by stable individuals.

Price fluctuations and overshoot During the same period, the population average f_σ also decreases fairly rapidly, but the drop is less pronounced than the drop in f_σ. As figures 9a and 9b show, the number of market makers rapidly increased during this period, as did the average latency of the market makers. Since the mean and median are close in both figure[6], the mean is representative of the evolution of the entire population.

Responsiveness f_{t} measures the time it takes for the model to react to the shock in the fundamental, and the evolution of the population wide statistics is shown on figure 8b. Although the GA is instructed to minimize f_{t} in order to look for more faster models, it clearly fails to do this. Indeed, the most responsive simulation took only about 4000 rounds to reach the new fundamental, but this individual died out in favor of slower individuals. In the last generation the most responsive simulation took around 14000 rounds to reach the new fundamental. The reasons for this failure to locate responsive models is discussed in section 10.1. In (the A) large change of the average of f_{t} happens in the rounds five to 15. In this period, the median is lower than the mean, which means that the growth in the mean can be attributed to a minority of individuals.

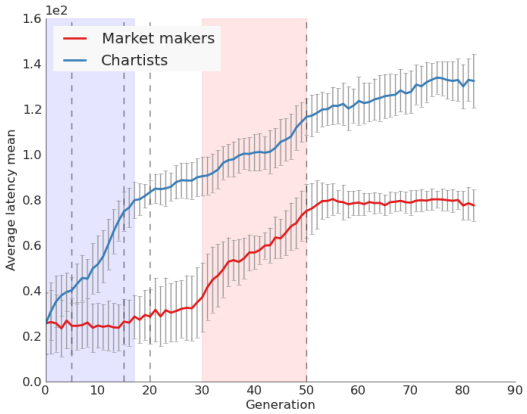
On figures 8 and 9, the two areas shaded in a light blue and light red respectively show the two periods during which there was a drastic change in parameters and fitness-values. By comparing the time at which parameters and fitness-values change, it is possible to get an idea of how parameters influence the fitness-values. To that end, figure 9 shows the evolution of each of the parameters that were varied by the GA[7].

The two periods indicated by the shaded squares seem to reflect some sudden changes in the parameters.

Average agent latency As is shown on figure 9a, individuals containing large latency parameters are selected for both HFT market makers and HFT chartists. $\mathrm{E}_{\mathcal{P}}[\lambda_{c,\mu}]$

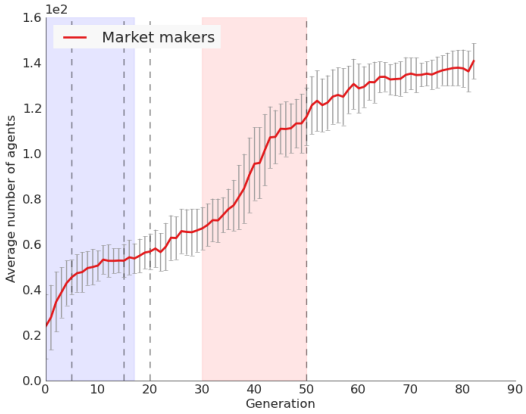
[6]Since f_{o} is discrete, the median and min statistics are also discrete

[7]Since the median was found to follow the mean nicely for all the parameters, the medians are not displayed. Also, the gray error bars show the population wide variance



(a) Evolution of $\lambda_{m,\mu}$ and $\lambda_{c,\mu}$

(b) Evolution of $\lambda_{m,\sigma}$ and $\lambda_{c,\sigma}$



(c) Evolution of N_m

Figure 9: Evolution of the model parameters in experiment $\mathcal{D}_{10}$

grows grows quickly during the first 20 rounds (blue shade). Referring back to figure 8, it is seen that $\mathrm{E}_{\mathcal{P}}[f_{\mathrm{t}}]$ and $\mathrm{E}_{\mathcal{P}}[f_{\mathrm{o}}]$ grows and shrinks respectively. As for $\mathrm{E}_{\mathcal{P}}[\lambda_{m,\mu}]$, it grows from rounds 20 through 50 (red shade), and this seems to be strongly reflected in the growth of $\mathrm{E}_{\mathcal{P}}[f_{\mathrm{s}}]$, and to a lesser degree a decline in $\mathrm{E}_{\mathcal{P}}[f_{\mathrm{o}}]$ and $\mathrm{E}_{\mathcal{P}}[f_{\sigma}]$. Furthermore, the small size of the error bars on both curves show that the population consistently moves towards containing more individuals with larger latency parameters for both HFT agent types. While initially $\mathrm{E}_{\mathcal{P}}[\lambda_{m,\mu}] \approx \mathrm{E}_{\mathcal{P}}[\lambda_{c,\mu}]$, the population wide mean $\mathrm{E}_{\mathcal{P}}[\lambda_{c,\mu}]$ ends up being roughly 1.5 times larger than $\mathrm{E}_{\mathcal{P}}[\lambda_{c,\mu}]$. Finally, note also that the growth of $\mathrm{E}_{\mathcal{P}}[\lambda_{c,\mu}]$ and $\mathrm{E}_{\mathcal{P}}[\lambda_{m,\mu}]$ seem to be somewhat independent, as they sometimes grow together, sometimes not.

Number of market makers The number of market makers increases almost every generation, but grows especially quickly through rounds 20 to 50 (red shade)

Variance of agent latency Figure 9b: The trends for $\mathrm{E}_{\mathcal{P}}[\lambda_{c,\sigma}]$ and $\mathrm{E}_{\mathcal{P}}[\lambda_{m,\sigma}]$ are less clear, as the population-wide variances $\mathrm{Var}_{\mathcal{P}}[\lambda_{c,\mu}]$ and $\mathrm{Var}_{\mathcal{P}}[\lambda_{m,\mu}]$, illustrated by the large error bars, are high compared to the change in $\mathrm{E}_{\mathcal{P}}[\lambda_{c,\mu}]$ and $\mathrm{E}_{\mathcal{P}}[\lambda_{m,\mu}]$. While this could mean that the market behaves more regularly when the difference between the latency parameters of the trading agents is smaller, further experiments would have to be carried out to confirm this fact.

In summary, the genetic algorithm prefers simulations with many, but relatively slow market makers. Apparently simulations with slow chartists also outperformed those with fast chartists, but since the number of HFT chartists was fixed at $N_c =$, this experiment does not reveal how the simulation would perform with more (or less) chartists. It is possible to imagine that the market would perform just as well with a few and fast chartists. Later sections contain the analysis of experiment $\mathcal{D}_{11}$ in which the number of chartists were varied. The discussion above can be summarized as follows:

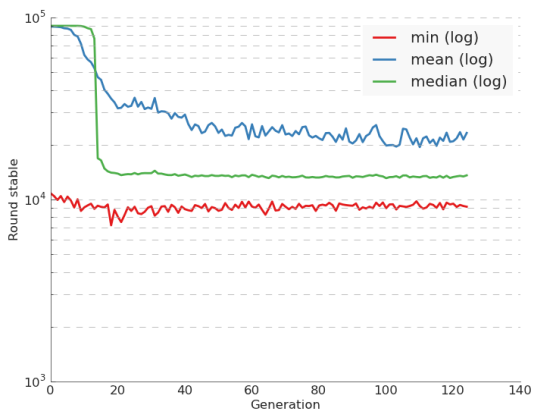
1. The responsiveness of the market is influenced by latency of the chartists. Slower chartists made the market require more time to respond to the fundamental shock.

2. The time it takes for the market to adjust to the shock in the fundamental is by the number of market makers and on the latency of the market makers. More but slower market makers seems to make the market settle within the stability margin faster.

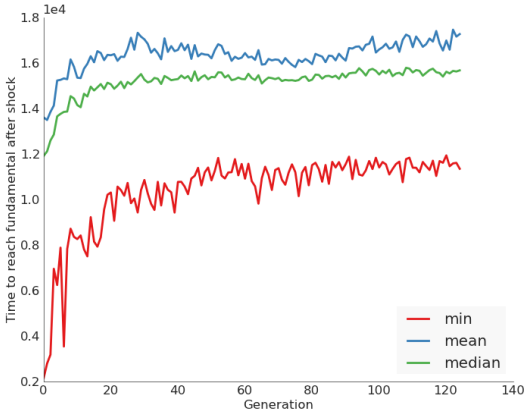
3. The overshoot, as well as the average size of the price fluctuations of the market, are both influenced by the latency of both agent types, as well as the number of market makers.

4. The market was more stable but reacted slowly when the chartists were slower than the market makers.

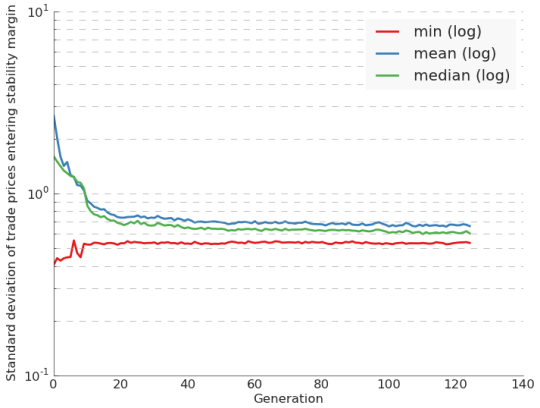
The accuracy of the above analysis is limited as it only looks at population wide statistics at a given point in the duration of the GA. The following section contain an analysis in which the generation to which each data point belongs is considered irrelevant. The analysis will try to confirm each of the four statements above.



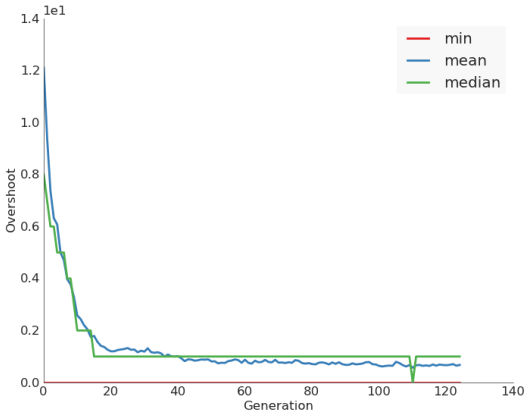
(a) Evolution of f_s



(b) Evolution of f_t



(c) Evolution of f_σ

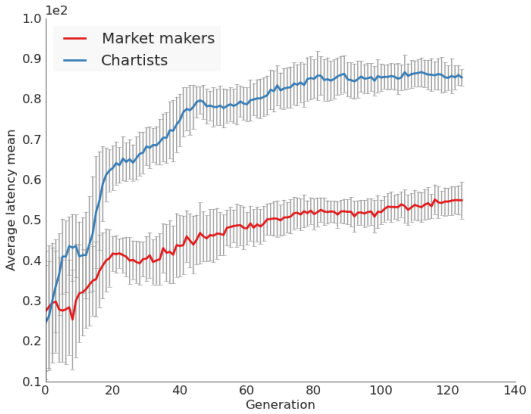


(d) Evolution of f_o

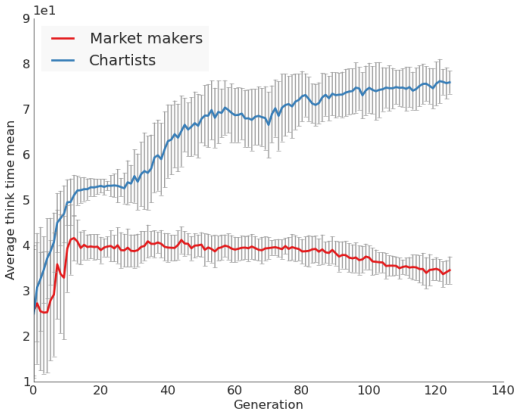
Figure 10: Evolution of the four fitness measures in experiment $\mathcal{D}_9$

11.2 Fixed number of chartists and market makers

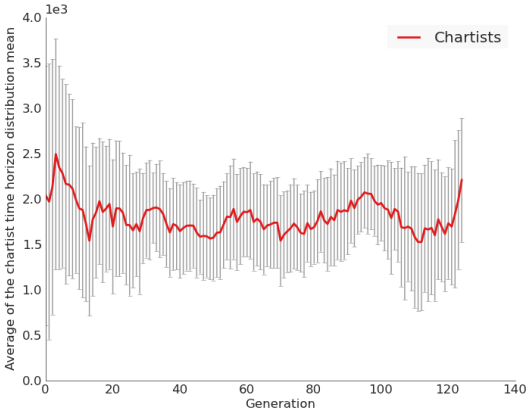
When the GA cannot change the number of chartists and market makers, it has to find better fitness values by selecting the right latency parameters. As shown on figure 10, the GA managed to find models with little or no overshoot, non-flickering prices, and which become stable. The price of having these nice qualities seems to be a slower response time to the shock. The GA find these well-behaving markets by selecting latency parameters such that the chartists are slower than the market makers. $\mathrm{E}_{\mathcal{P}}[H_{c,\mu}]$ and $\mathrm{E}_{\mathcal{P}}[W_{c,\mu}]$ change little or are more or less unchanged, which seems to indicate that they have little effect of the fitness values, at least compared to other time related parameters such as $\lambda_{c,\mu}$, $\lambda_{m,\mu}$, $\tau_{c,\mu}$ and $\tau_{m,\mu}$.



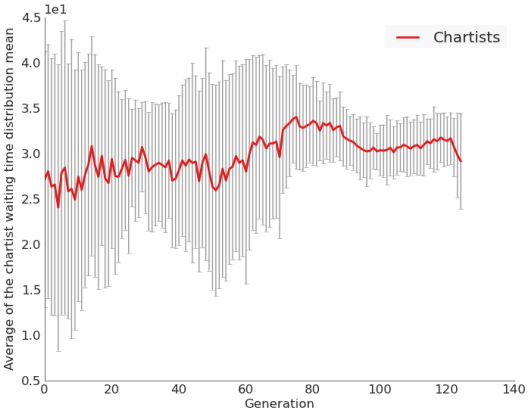
(a) Evolution of $\lambda_{m,\mu}$ and $\lambda_{c,\mu}$



(b) Evolution of $\lambda_{m,\sigma}$ and $\lambda_{c,\sigma}$

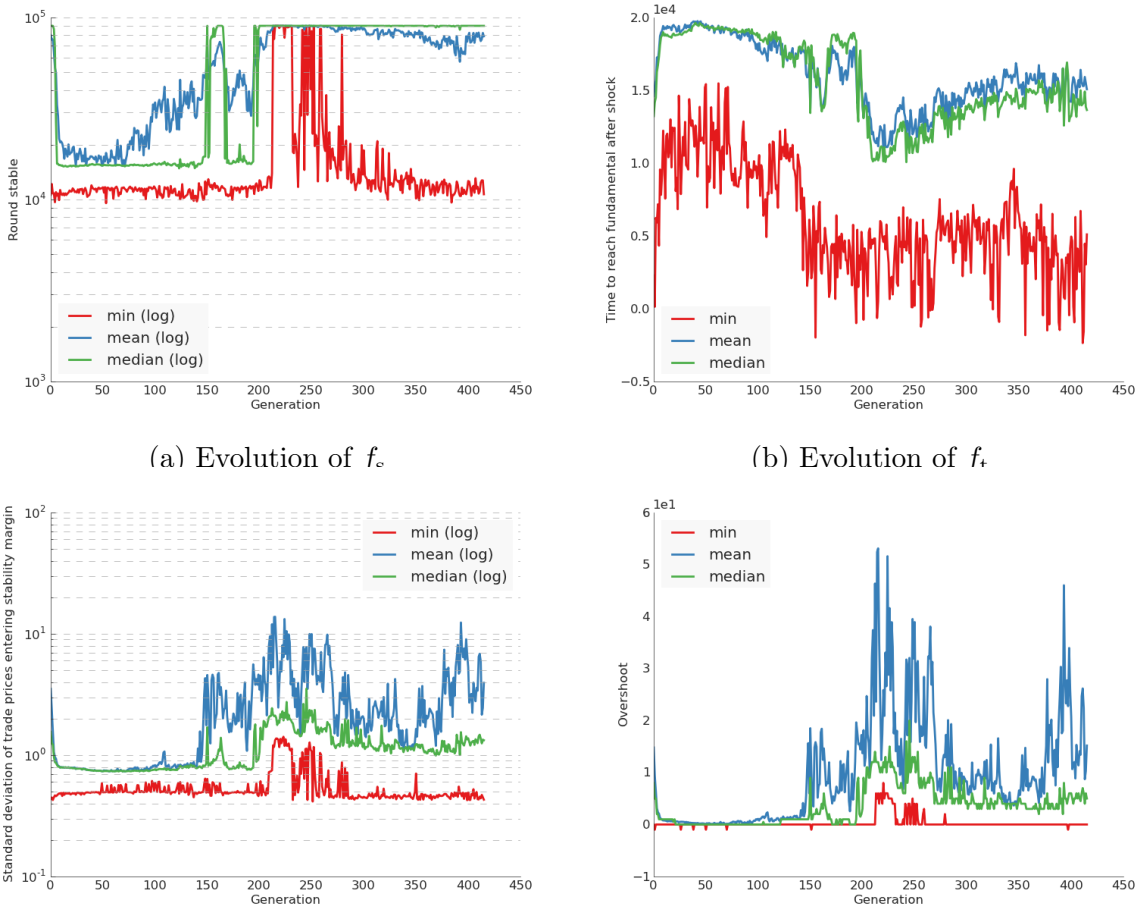


(c) Evolution of N_m



(d) Evolution of N_m

Figure 11: Evolution of the model parameters in experiment $\mathcal{D}_{10}$



(a) Evolution of f_s

(b) Evolution of f_t

(c) Evolution of f_σ

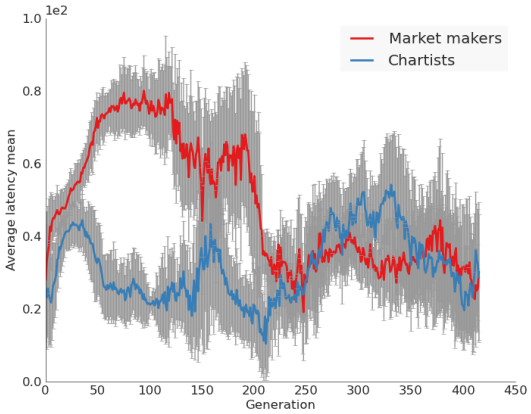
(d) Evolution of f_o

Figure 12: Evolution of the four fitness measures in experiment $\mathcal{D}_{11}$

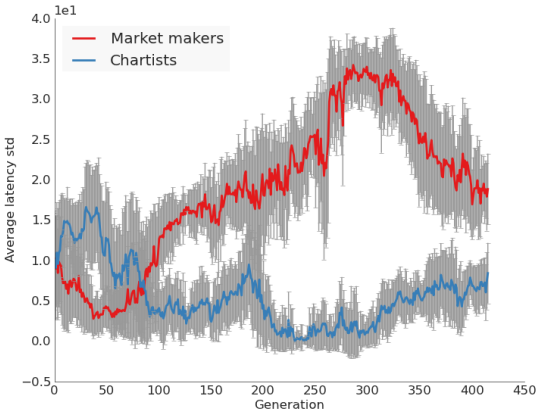
11.3 Variable number of chartists

The evolution of the fitness values in $\mathcal{D}_{11}$, shown in figure 12 fluctuate significantly more than in $\mathcal{D}_9$ and $\mathcal{D}_{10}$. In contrary to the two other experiments, the GA here manages to decrease f_t, and $\text{E}_\mathcal{P}[f_\text{t}]$ drops almost 10000 rounds around generation 200. At the same time $\text{E}_\mathcal{P}[f_\text{o}]$, $\text{E}_\mathcal{P}[f_\sigma]$ and $\text{E}_\mathcal{P}[f_\text{s}]$ all rise, again indicating the there exists a trade-off between speed and stability in the model. At the point in the evolution where $\text{E}_\mathcal{P}[f_\text{t}]$ drops, several interesting things happen with the model parameters that live in the population. First of all the number of chartists increase dramatically, again pointing towards more chartists making the markets fast and unstable. Secondly, the market maker latency also drops to around the same level as the chartist latency. This is interesting because it could mean that the faster market makers help drive the market towards a larger overshoot.

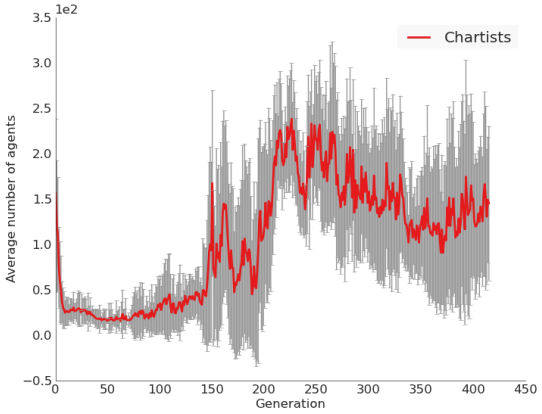
Market makers become slower and the chartists become faster. At the same time, the number of chartists rise rapidly



(a) Evolution of $\lambda_{m,\mu}$ and $\lambda_{c,\mu}$



(b) Evolution of $\lambda_{m,\sigma}$ and $\lambda_{c,\sigma}$



(c) Evolution of N_c

Figure 13: Evolution of the model parameters in experiment $\mathcal{D}_{11}$

$\mathcal{D}_{10}$	$N_m < q_1$	$N_m > q_9$
f_o	7.5	0.8
f_s	89876.2	18546.9
f_σ	1.6	0.6
f_t	12590.5	19832.1

Table 7: Average fitness values for the market with the top 10% highest and 10% lowest number of market makers

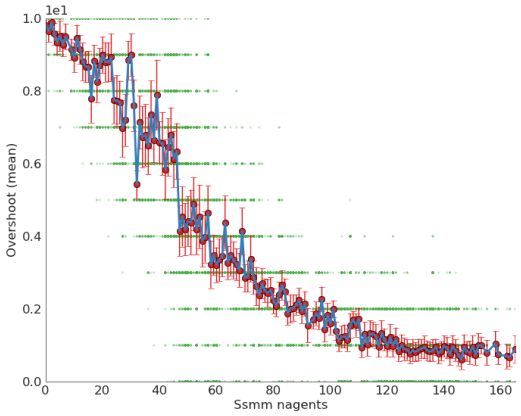
- A high number of market makers enable the market to respond quickly to the shock, but also cause the traded price to flicker more, and for the model to have a larger overshoot.
- Slower market makers also cause the market to respond faster to the shock

12 Population-wide parameter/fitness correlations

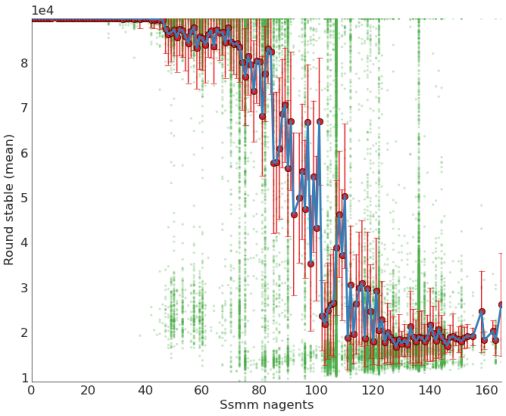
This section contains several figures which illustrate how the model fitness varies with the model parameters. In the figures showing the data from experiment $\mathcal{D}_{11}$, simulations with $f_\mathrm{o} > 10$ are removed in order to make the figures easier to interpret. In the following, the notation $\mathrm{E_p}[\cdot]$ is used for the population wide average of a model parameter of fitness measure. For instance, $\mathrm{E_p}[f_\mathrm{o}]$ is the average market overshoot, where the average is calculated over the total population of individuals that ever lived in the genetic algorithm.

12.1 Number of market makers

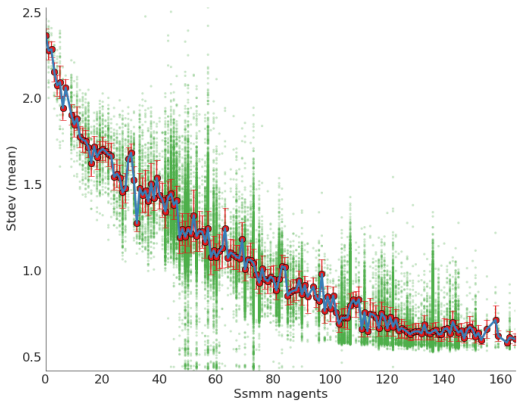
The number of market makers was kept fixed in experiments $\mathcal{D}_9$ and $\mathcal{D}_{11}$, but was varied in experiment $\mathcal{D}_{10}$. Figure 14 shows how the number of market makers correlates with the model fitness in experiment $\mathcal{D}_{10}$. Evidently a large number of market makers reduces the overshoot, whereas the market virtually always have an overshoot when there are few or no market makers. The same is true for the trade prices flicker: few market makers always means flickering prices. A higher number of market makers cause the market to be less responsive, while fewer market makers have the opposite effect. Finally, the number of market makers also influences how quickly the market settles within the stability margin, as markets with more market makers become stable faster than markets with few agents. Table 7 shows the average fitness values for models where the number of market maker agents was respectively below and above the first (q_1) and ninth (q_9) 10-quantiles in the dataset. The two quantiles were at $q_1 = 46$ and $q_9 = 136$. It is seen that the market containing a large number of market makers (more than 136) had a much smaller overshoot, less flickering prices, a slower response time, and stayed within the stability margin faster, compared to the markets with a low (less than 46) number of market makers.



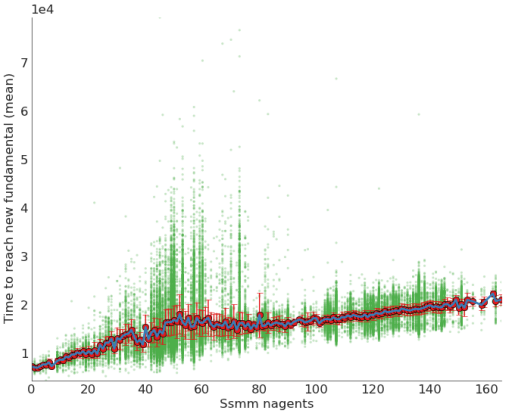
(a) Correlation between N_m and f_o



(b) Correlation between N_m and f_s



(c) Correlation between N_m and f_σ



(d) Correlation between N_m and f_t

Figure 14: Correlation between N_m and the four fitness measures in experiment $\mathcal{D}_{10}$

$\mathcal{D}_{10}$	$\lambda_{m,\mu} < q_1$	$\lambda_{m,\mu} > q_9$
f_o	5.6	0.9
f_s	85927.9	18329.3
f_σ	1.4	0.6
f_t	16649.4	18795.3

$\mathcal{D}_{11}$	$\lambda_{m,\mu} < q_1$	$\lambda_{m,\mu} > q_9$
f_o	14.2	1.7
f_s	81379.2	26153.0
f_σ	3.6	1.0
f_t	15856.2	18107.4

Table 8: Average fitness values for the market with the top 10% highest and 10% lowest number of market makers

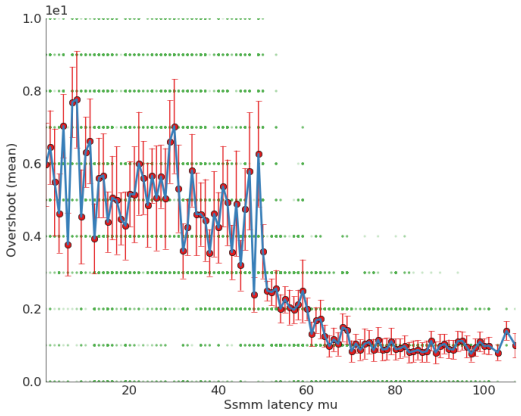
12.2 Market maker latency

The parameters controlling the market maker latency was varied in experiments $\mathcal{D}_9$, $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$. However, since the data from $\mathcal{D}_9$ turned out to be too noisy due to the large number of parameters included in the search, only data from $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$ was used.

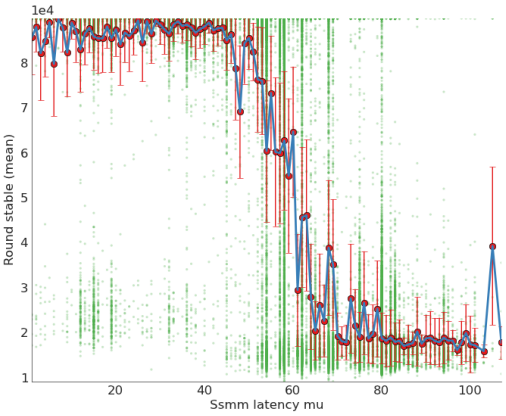
Fixing the number of chartists

In experiment $\mathcal{D}_{10}$, the number of chartists was kept fixed at $N_c = 150$, while N_m was varied by the GA. In this case, the $\lambda_{m,\mu}$ is found to be somewhat correlated with the fitness measures as illustrated on figure 15. Especially for $\lambda_{m,\mu} > 50$, the data seems to be consistent, as the error bars showing the standard deviation of the data are small in this region. However, for $\lambda_{m,\mu} < 50$, the model behavior is no longer predictable by using $\lambda_{m,\mu}$ alone. Figure 16 shows line plots of the four fitness measures plotted against $\lambda_{m,\mu}$. Each line shows the average fitness of markets in which the number of market makers is in a limited range as shown in the legend. The figure shows that even though the market maker latency does influence the market, the effect is secondary to that of the number of agents. For instance, when the market contains less than 25 market makers, all four fitness measures are more or less unchanged, as is evident by the nearly flat red curves. As the number of market makers grow, so does the importance of how fast they are. The average overshoot and the average time to catch up to the new fundamental only change slightly, even with over 100 market makers (yellow line). On the other hand, the average price flickering and the average number of rounds it takes for the market to settle within the stability margin both change significantly with the market maker speed for $N_m > 50$. In summary, figure 16 shows that in a market with only a few market makers, these agents have little influence no matter how fast they are. As the number of market makers grow, so does the collective force of all the market makers, and so does the importance of how slow or fast these agents are. Although the influence of $\lambda_{m,\mu}$ depends on N_m, . q_1 and q_1

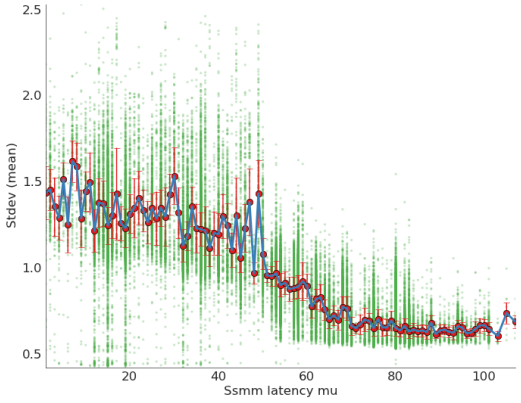
The next section will examine how the market behaves with respect to how many chartists are active in the market, and with respect to the latency of the chartists.



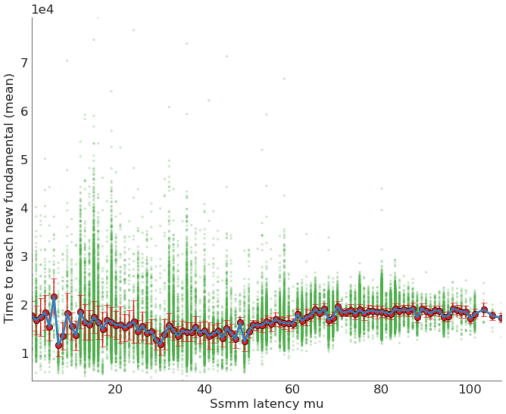
(a) Correlation between $\lambda_{m,\mu}$and f_o



(b) Correlation between $\lambda_{m,\mu}$and f_s



(c) Correlation between $\lambda_{m,\mu}$and f_σ



(d) Correlation between $\lambda_{m,\mu}$and f_t

Figure 15: Correlation between $\lambda_{m,\mu}$ and fitness values (fixed N_c, variable N_m)

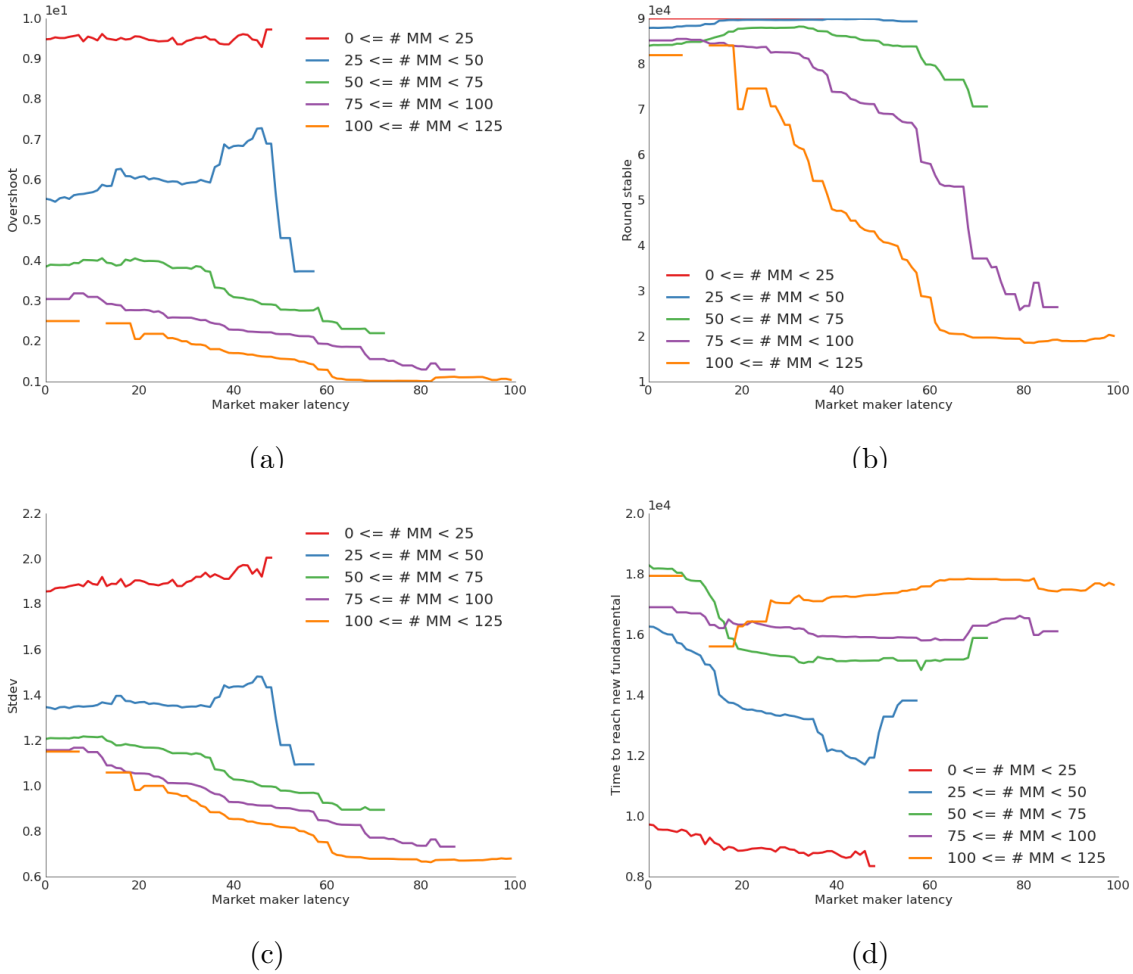
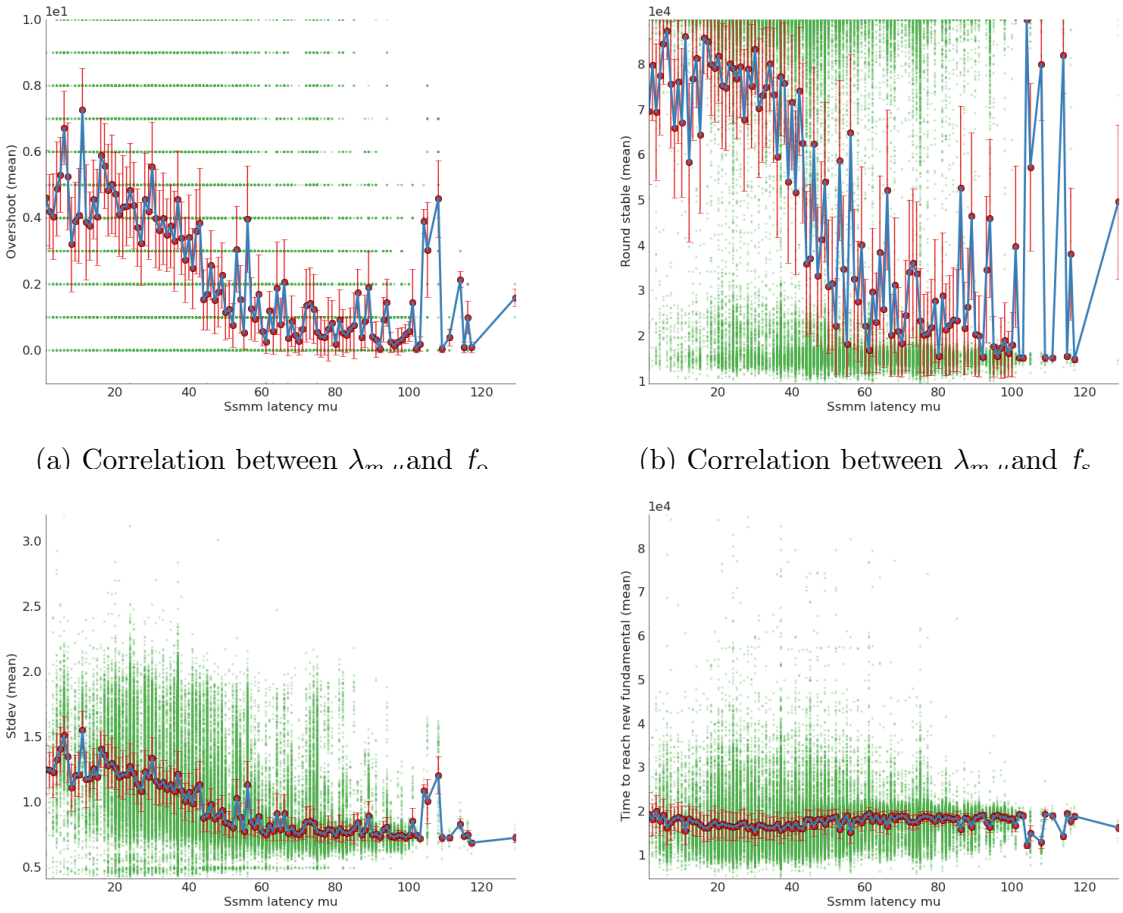


Figure 16: Relation between N_m, $\lambda_{m,\mu}$, and the model fitness when the number of chartists was fixed to $N_c = 150$ agents. Due to missing data, some of the curves are not complete.



(a) Correlation between $\lambda_{m,\mu}$and f_σ

(b) Correlation between $\lambda_{m,\mu}$and f_σ

(c) Correlation between $\lambda_{m,\mu}$and f_σ

(d) Correlation between $\lambda_{m,\mu}$and f_t

Figure 17: Correlation between $\lambda_{m,\mu}$ and fitness values (fixed N_m, variable N_c)

12.2.1 Fixing the number of market makers

In experiment $\mathcal{D}_{11}$, the number of chartists was varied, while the number of market makers were kept at a constant $N_m = 52$ agents. While it was fairly obvious that the market would not be impacted by changing $\lambda_{m,\mu}$when only a few market makers were active, it is less obvious that the same is true for the number of chartists. Yet figure

12.3 Number of chartists

Figure 19 shows the average population wide number of agents $\mathrm{E}_{\mathcal{P}}[N_c]$ plotted against each of the four fitness measures, and the figures are summarized below.

- The more chartists a market has, the faster it responds to the fundamental. This is especially true when comparing markets with less than 100 chartists, and less pronounced when comparing markets with over 100 chartists.

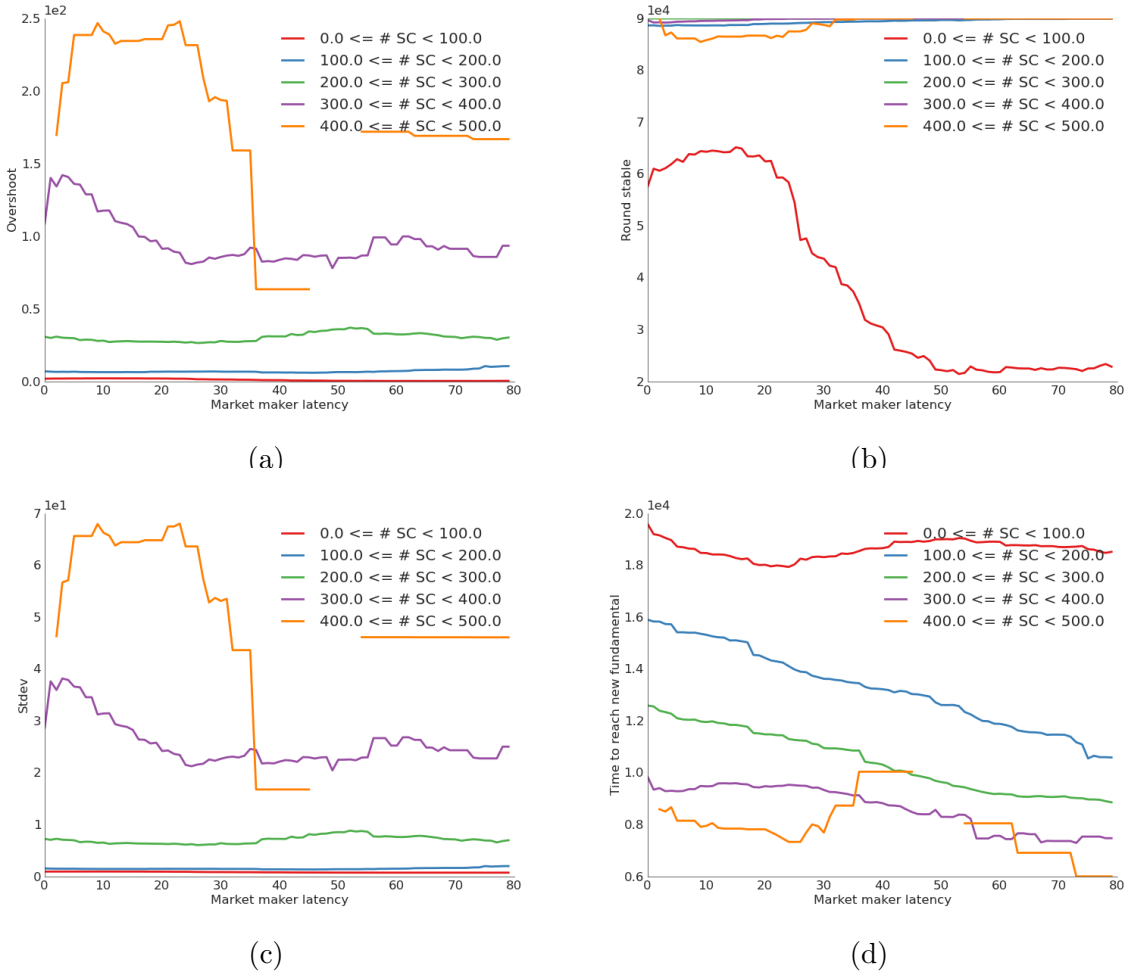
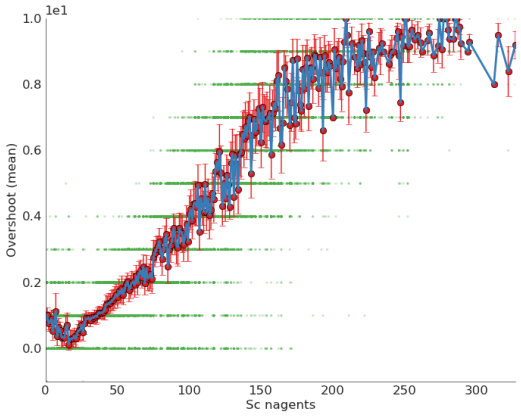
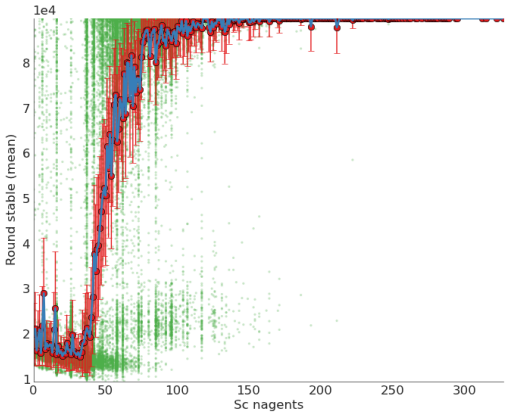


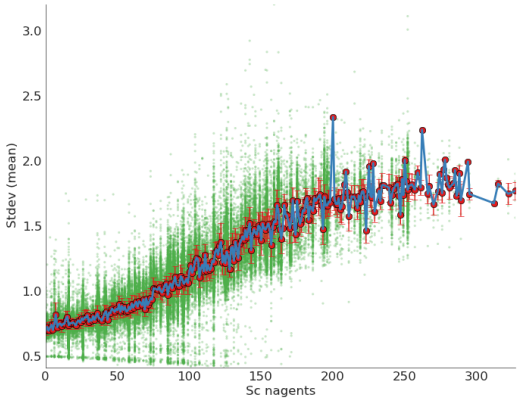
Figure 18: Relation between N_m, $\lambda_{m,\mu}$, and the model fitness when the number of market makers was fixed to $N_m = 52$ agents.



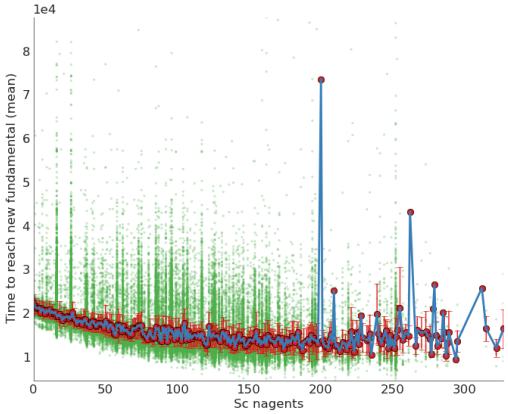
(a) Correlation between $N_c$and f_o



(b) Correlation between $N_c$and f_s



(c) Correlation between $N_c$and f_σ



(d) Correlation between $N_c$and f_t

Figure 19: Correlation between $N_c$and the four fitness measures when $N_m = 52$ (experiment $\mathcal{D}_{11}$)

- The model overshoot is also correlated with the number of chartists in such a way that markets with more chartists have a larger overshoot on average. Whereas the market only seemed to benefit from a decreased response time when the number of chartists were kept below 100, the overshoot continues to grow steadily larger even as the number of chartists is increased beyond 100 agents.

- f_σ is correlated with the number of chartists in the same way as f_o, such that more chartists make the traded prices flicker more.

- Finally, the graph for f_s show that the market rarely becomes stable when it contains more than 50 chartists or so.

The large errorbars around the points with a large value of N_c is caused by data sparsity in this region. The GA was set to search for stable markets, and since markets with a large number of chartists tend to be unstable, such markets were rarely selected for creating offspring.

12.4 Chartist latency

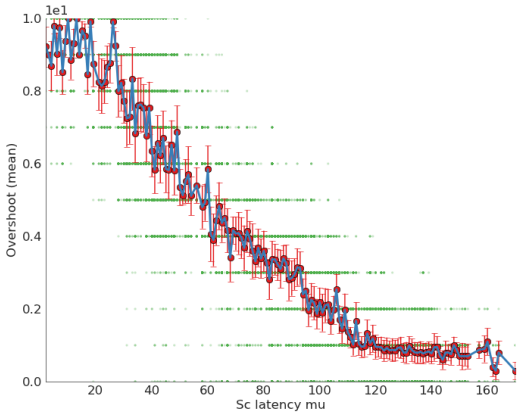
Fixed number of chartists

Figure 20a shows that $\lambda_{c,\mu}$ is negatively correlated with f_o, such that markets with faster chartists are more likely to have a larger overshoot. Next, figure 20 shows that $\lambda_{c,\mu}$ is negatively correlated with f_σ, such that markets with faster chartists are more likely to have flickering trade prices.

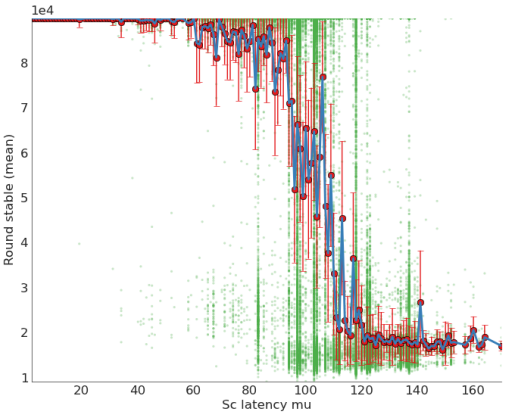
As for the market responsiveness, it is seen that $\lambda_{c,\mu}$ is positively correlated with f_t, such that markets with faster agents is more likely to have a shorter response time to the market. Figure 20d confirms that markets with fast chartists did actually manage to reach the new fundamental price faster than those markets having slow chartists. The average response time of markets in which the chartists had a latency of less than 30 rounds was around 18000 rounds, whereas it was around 25000 rounds with chartists with more than 100 rounds of latency. The market response time is most sensitive in the range $20 < \lambda_{c,\mu} < 60$, and does not change much for larger latencies.

The plots of f_o, f_σ and f_t show that predicting the three fitness measures in markets with slow chartists would be more accurate than for markets with fast chartists, as the correlation of f_o, f_σ and f_t with $\lambda_{c,\mu}$ is stronger for larger values of $\lambda_{c,\mu}$.

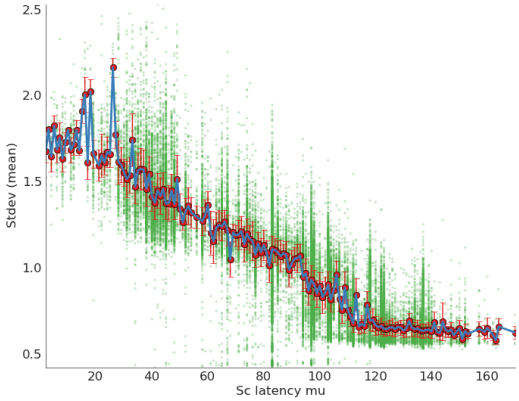
Figure 20b shows that $\lambda_{c,\mu}$ is positively correlated with f_s, but also that the relationship between $\lambda_{c,\mu}$ and f_s seems highly non-linear. The figure illustrates the binary nature of the stability criteria, that is, that a simulation is either stable or not stable. This causes f_s to have a high conditional variance of f_s given $\lambda_{c,\mu}$ in the region $50 < \lambda_{c,\mu} < 120$, meaning that that prediction of f_s from $\lambda_{c,\mu}$ in this region would not be very accurate. What this means is that the stability of a simulation is highly dependent on factors other than $\lambda_{c,\mu}$, when the parameter is within 50 to 120 rounds. When the chartists are faster than 50 rounds, the market is almost always unstable, and when the chartists are slower than 120 rounds the market is almost always stable.



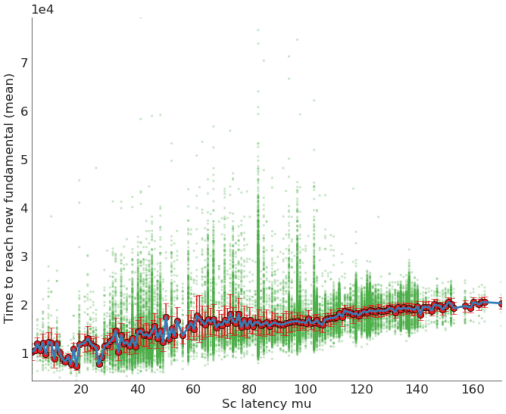
(a) Correlation between $\lambda_{c,\mu}$and f_o



(b) Correlation between $\lambda_{c,\mu}$and f_s



(c) Correlation between $\lambda_{c,\mu}$and f_σ



(d) Correlation between $\lambda_{c,\mu}$and f_t

Figure 20: Correlation between chartist latency and fitness values (fixed N_c, variable N_m)

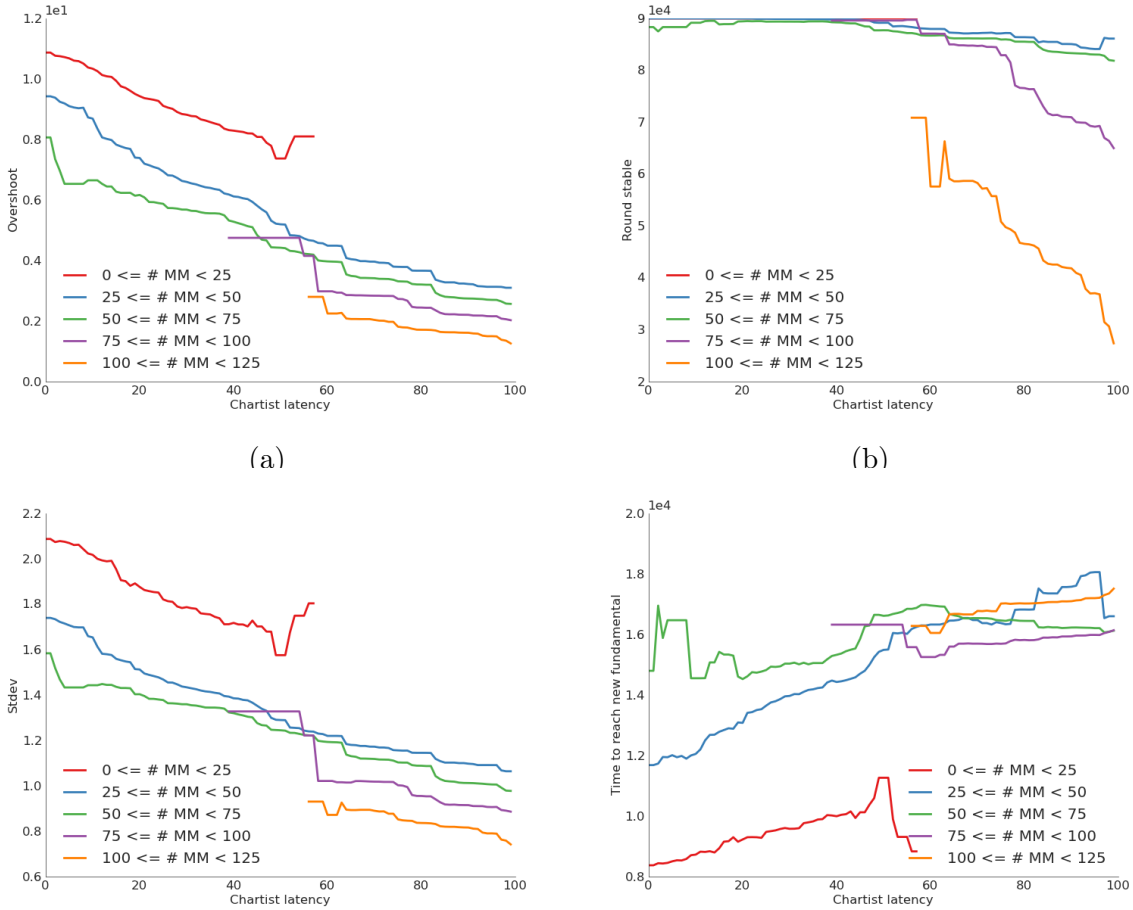
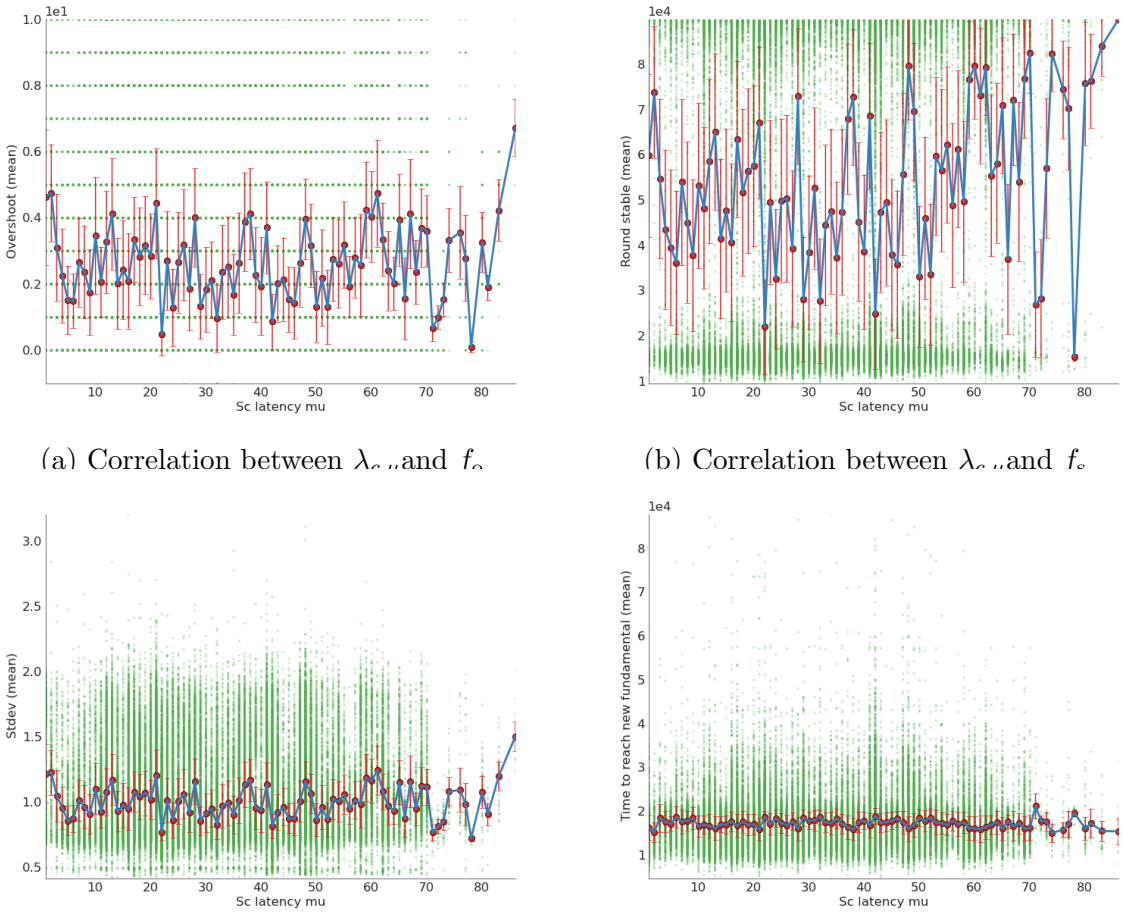


Figure 21: Relation between N_m, $\lambda_{c,\mu}$, and the model fitness when the number of chartists was fixed to $N_c = 150 agents$. Due to missing data, some of the curves are not complete.



(a) Correlation between $\lambda_{c,\mu}$and f_{o}

(b) Correlation between $\lambda_{c,\mu}$and f_{s}

(c) Correlation between $\lambda_{c,\mu}$and f_{σ}

(d) Correlation between $\lambda_{c,\mu}$and f_{t}

Figure 22: Correlation between chartist latency and fitness values (fixed N_m, variable N_c)

Fixed number of market makers

Figures 22 seems to indicate that no correlations exist between the speed of the chartist agents, and the model fitness measures. However, since figure 20 does point towards the existence of such correlations, something else must be obscuring the scatter plots in 22. The reason is found to be that the number of chartists was not kept constant in experiment $\mathcal{D}_{11}$. It turns out that $\lambda_{c,\mu}$ is in fact correlated with f_{o}, $f_{\sigma}$and f_{t}, but that the correlation depends heavily on the number of chartists in the market.

12.5 Chartist to market maker ratio

The above observations about how the number of agents influence the stability and speed of the market pointed out that more market makers made the market slow but stable, while more chartists made the market fast, but unstable. By merging $\boldsymbol{P}\mathcal{D}_{10}$ and $\boldsymbol{P}\mathcal{D}_{11}$, we can calculate the ratio, ρ_{A}, between the number of chartists and the number of market

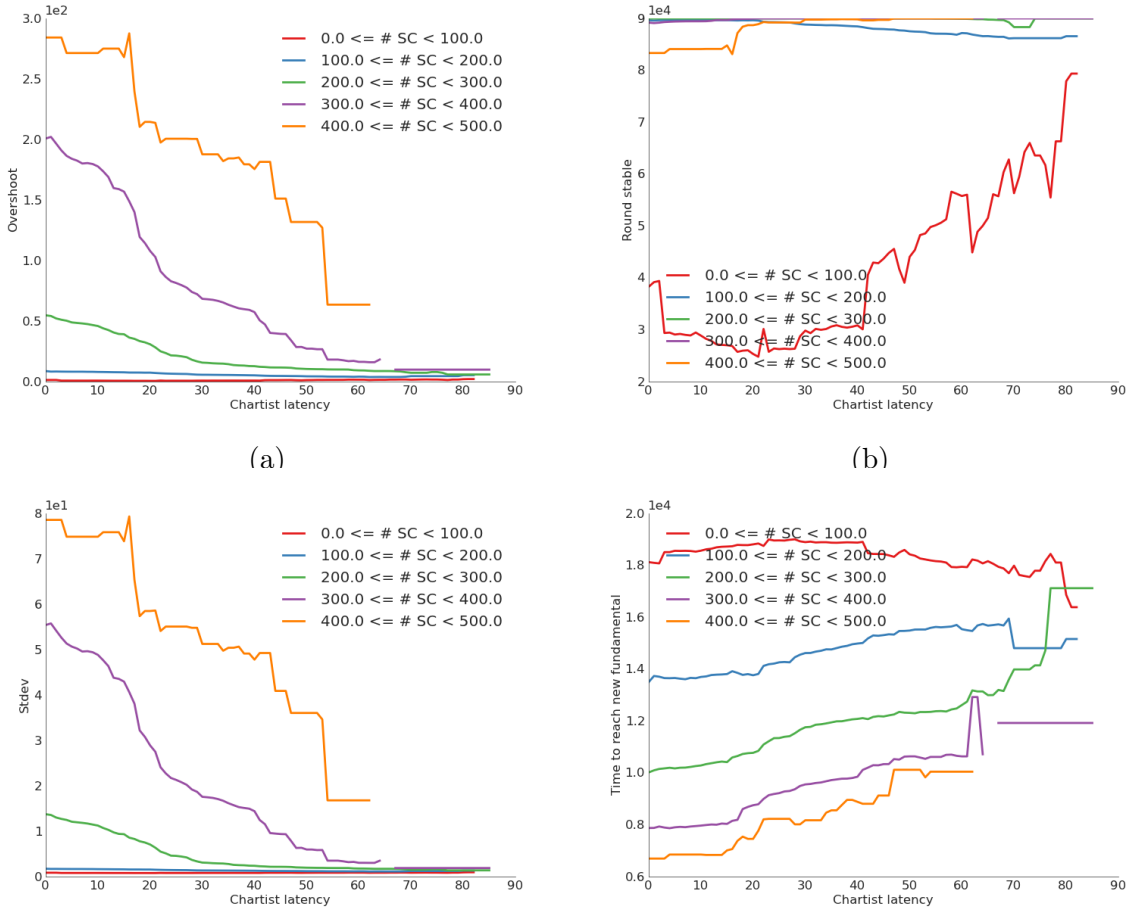


Figure 23: Relation between N_c, $\lambda_{c,\mu}$, and the model fitness when the number of market makers was fixed to $N_m = 52$ agents.

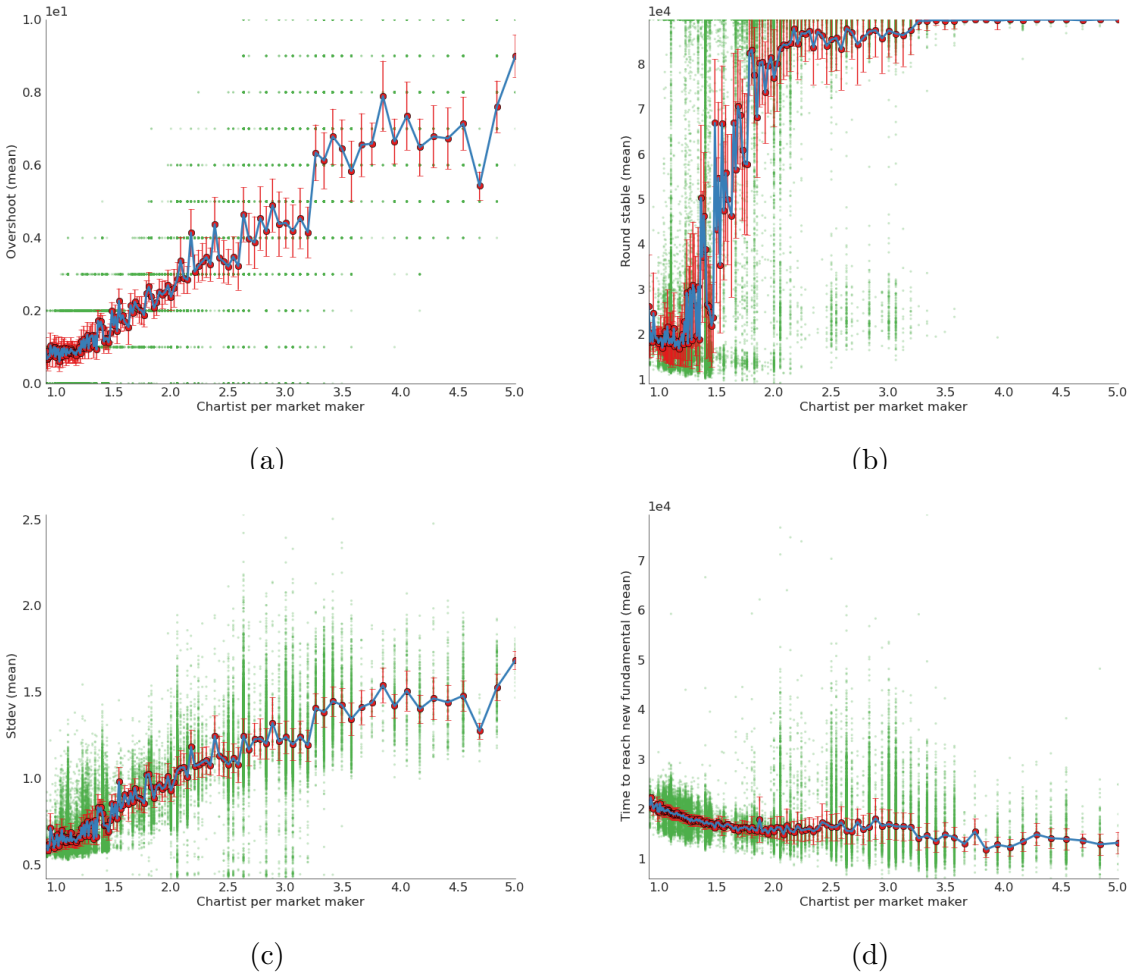
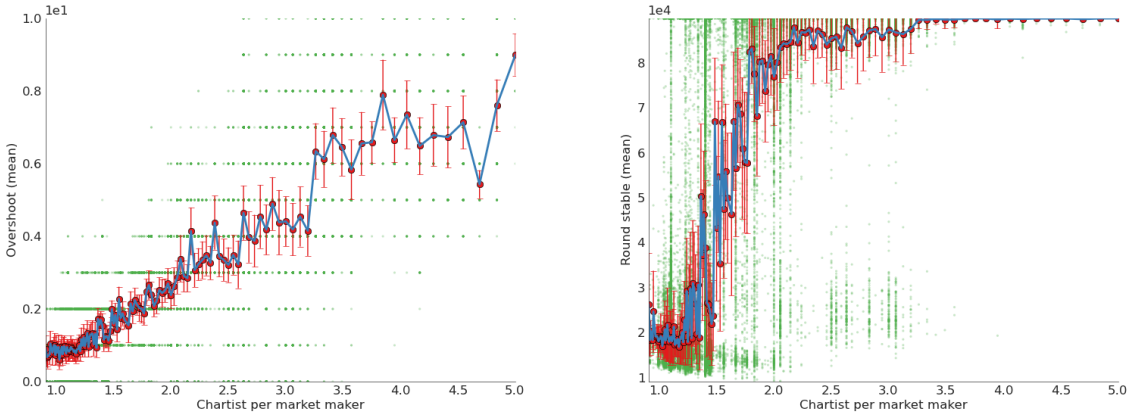


Figure 24: Correlations between ρ_{A} and the fitness values when $N_c = 150$

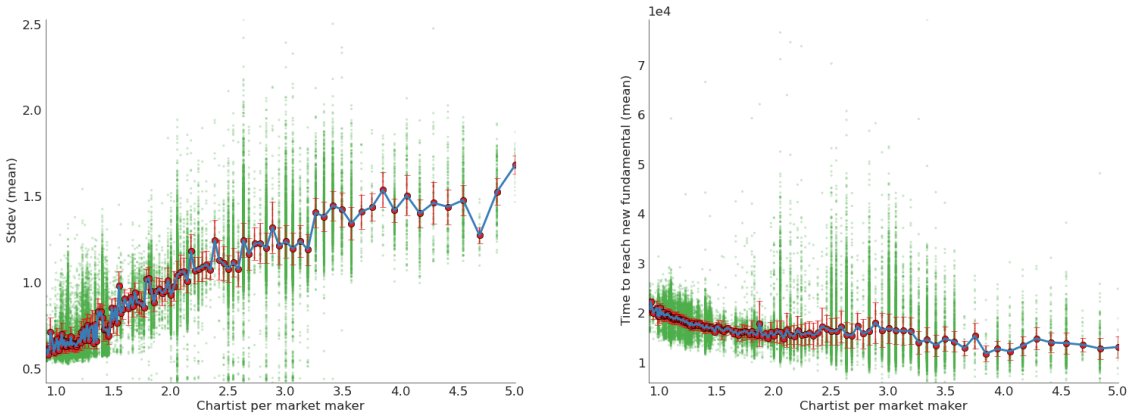
makers, and see how the fitness values correlate. Figure 24 and 25 show the resulting scatter plots for $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$.

13 Grouping models by behavior

This section is be concerned with trying to tie various patterns of model behavior to different regions in the parameter space. The quickest way to get an idea of how the data generated by the simulations is distributed is to make scatter plots. Scatter plots are probably among the most rudimentary of techniques for data analysis, yet they can be incredibly informative, especially when the data that is visualized is low-dimensional. The data of the model fitness is four-dimensional, requiring twelve plots to visualize all combinations. However, since $f_{\mathrm{o}}$is discrete with a small range of values, it is not suitable for a scatter plot. Furthermore, some scatter plots are not useful for interpretation if they do not show any structure in the data. Figure 26 shows three scatter plots which were

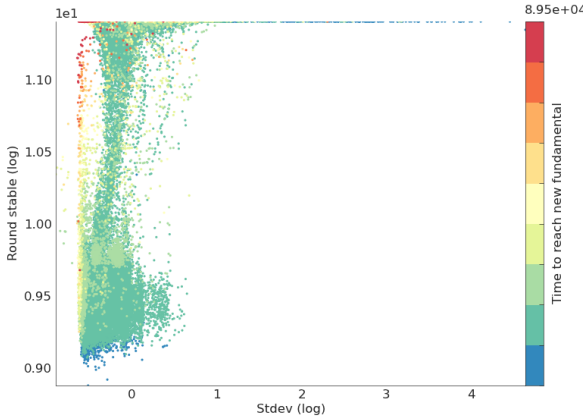


(a) (b)

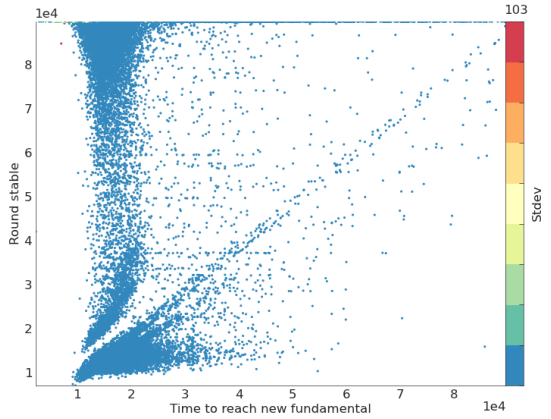


(c) (d)

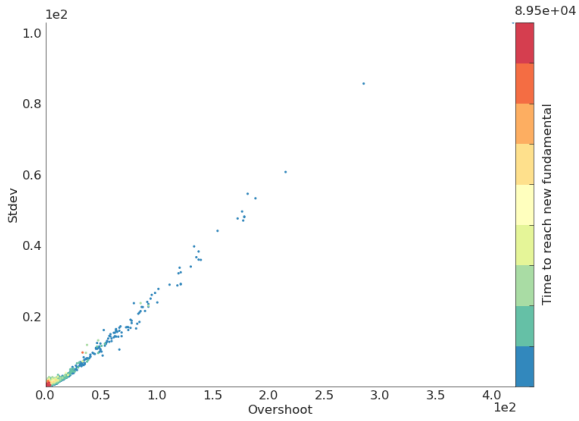
Figure 25: Correlations between ρ_{A} and the fitness values when $N_m = 52$



(a) $\log f_\sigma$ vs. $\log f_\mathrm{s}$ vs. f_t



(b) $f_\mathrm{s}$vs. $f_\mathrm{t}$vs. f_σ



(c) $\log f_\mathrm{o}$ vs . $\log f_\sigma$ vs. f_t

Figure 26: Scatter plots of fitness measures in experiment $\mathcal{D}_9$.

found to best illustrate the structure of the dataset from $\mathcal{D}_9$. Note also that coloring each point corresponding to its value in one dimension makes it possible to show how the data is distributed in three dimensions. The scatter plots do seem to reveal some structure, the presence of large values in the $f_\sigma$feature obscures the nature of this structure, in spite of the logarithmic scaling. The plot showing $\log f_\sigma$ vs. $\log f_\mathrm{s}$ is squeezed to the left, and the color grading on the scatter plot for $\log f_\mathrm{o}$ vs . $\log f_\sigma$ reveals no variety in the $f_\sigma$feature. In an attempt to get some more information out of the scatter plot, data points with an overshoot of over 100 % of the shock to the fundamental (corresponding to $f_\mathrm{o} > 10$) are removed. The resulting scatter plots for the reduced data set are shown on figures 27 and 28.

First of all, it is seen that while the data is distributed similarly in the three data sets, there are some differences. The data from $\mathcal{D}_9$ seems to have many "lonely" data points, which are not part of any cluster, whereas the data from $\mathcal{D}_{10}$ somehow seems to be the cleanest of the three. In all three data sets, there are clusters of data. The clusters do

not necessarily mean anything in themselves. They might simply be due to the way that data points are mutated and crossed by the GA. However, by considering which regions of the fitness space that each cluster covers, it is possible to add meaning to the clusters in terms of model behavior.

13.1 Manually grouping simulations by behavior

Table 9 contains an overview of the named criteria used for roughly grouping simulations into different types of behavior. The following text contains the reasoning for why each of these groups are interesting.

In figure 27, the black dashed lines at $f_\mathrm{t} = f_\mathrm{s}$ divide each plot into region A, (upper left triangle) and region B (lower right triangle). Region A contains the fitness-points of the simulations which are counted as stable *after* they reach the new fundamental, and region B contain those that become stable before.

The description below provides a brief summary of which simulations belong in the two regions.

$f_\mathbf{s} < f_\mathbf{t}$ This happens when the traded price never leaves the stability margin after reaching the new fundamental price. Note however that this case does not necessarily mean that the prices do not flicker.

$f_\mathbf{s} > f_\mathbf{t}$ This happens when the traded price leaves the stability margin once or more after reaching the new fundamental. The traded price can be close to the fundamental, but flickers in and out of the stability margin as on figure 5. The figure shows one example where the trade price fairly stable and with no overshoot, leading to good (low) f_σ and f_o fitness values to be assigned to the parameters. However, even though the traded prices are mostly within the stability margin, occasional flickers out of the margin causes the simulation to score a bad (high) f_s fitness. Note also that f_t is undefined in this case.

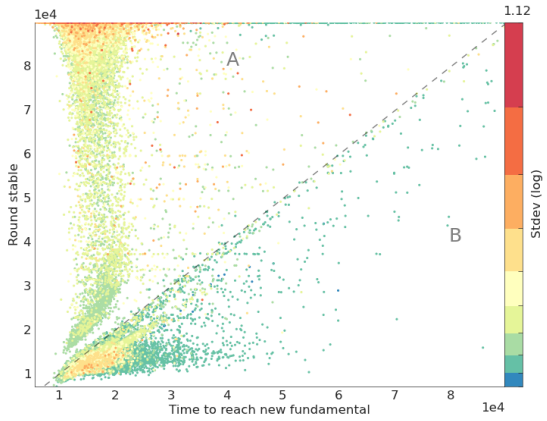
$f_\mathbf{s} = f_\mathbf{t}$ This happens if a trade is executed at price $m_\mathrm{stable} - p_\mathrm{fas} < p_\mathrm{match} < m_\mathrm{stable} + p_\mathrm{fas}$, and another trade is executed at price $p_\mathrm{match} = p_\mathrm{fas}$ in the same round.

Fast and stable simulations with flickering prices

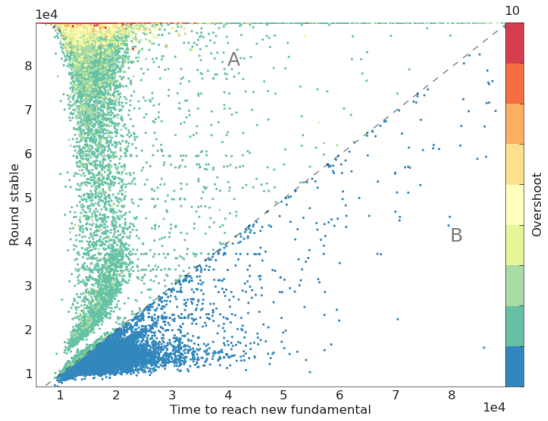
The points in the lower left corner in are those which quickly reached the new fundamental price, and quickly because stable, leading those simulations to be assigned low f_t and f_s fitness-values. These points are extracted by using filter $\mathcal{F}_1$ (see table 9)

Slow or fast and stable simulations with non-flickering prices

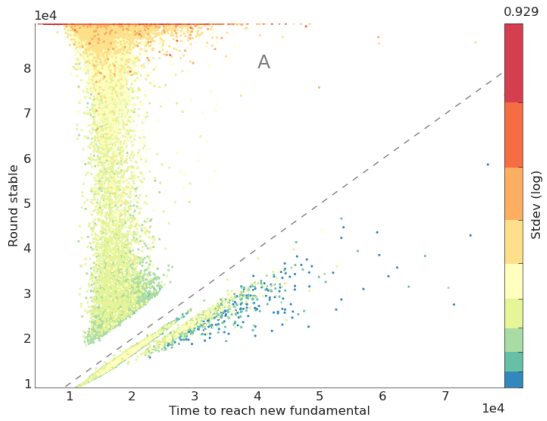
All three data sets have data points which are close to the diagonal. However, only $\mathcal{D}_9$ has data points which are close to the diagonal in the upper right corner of the figure. These points are interesting because they belong to simulations which became stable as soon as they reached the new fundamental price. Hence, these simulations should have prices that do not flicker, and therefore yield a small f_σ-fitness. This is confirmed by looking at the



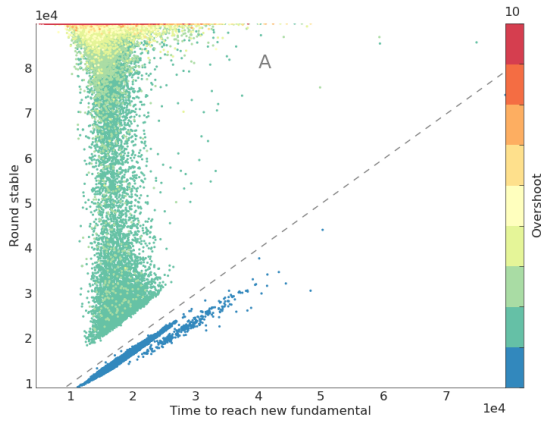
(a) Experiment $\mathcal{D}_9$



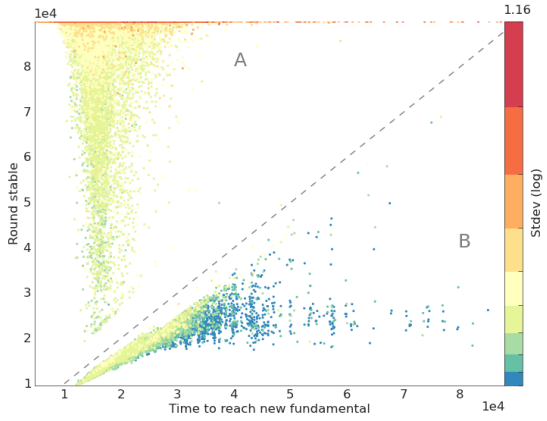
(b) Experiment $\mathcal{D}_9$



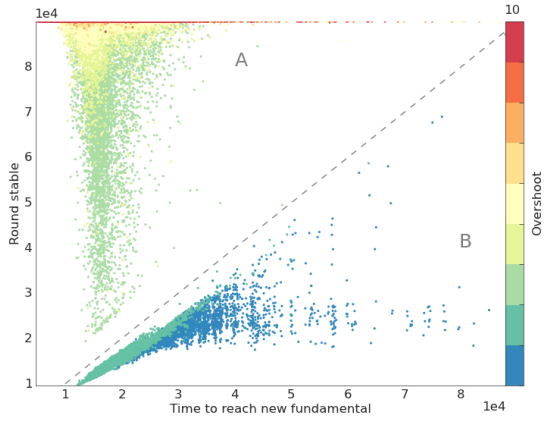
(c) Experiment $\mathcal{D}_{10}$



(d) Experiment $\mathcal{D}_{10}$

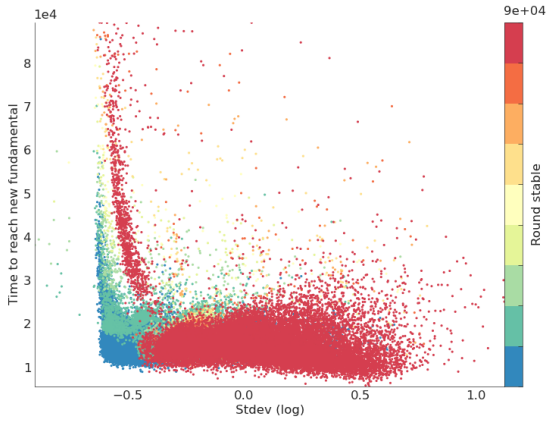


(e) Experiment $\mathcal{D}_{11}$

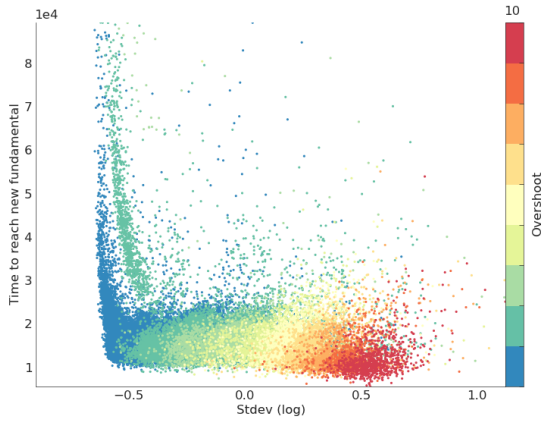


(f) Experiment $\mathcal{D}_{11}$

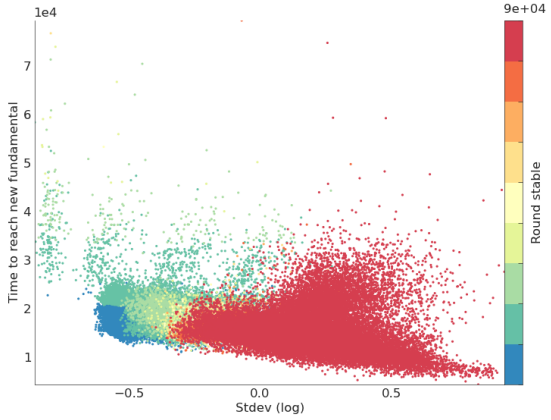
Figure 27: Scatter plot of f_s against f_t with coloring showing $\log f_\sigma$ and f_o



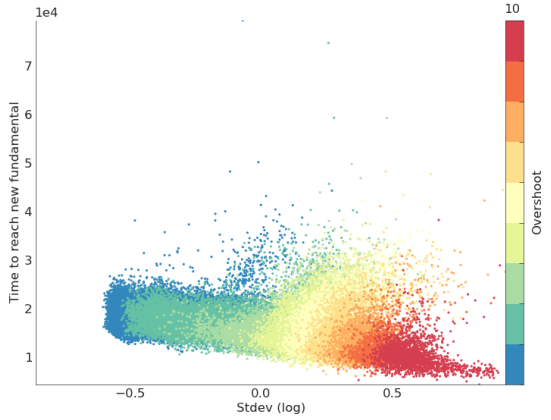
(a) Experiment $\mathcal{D}_9$



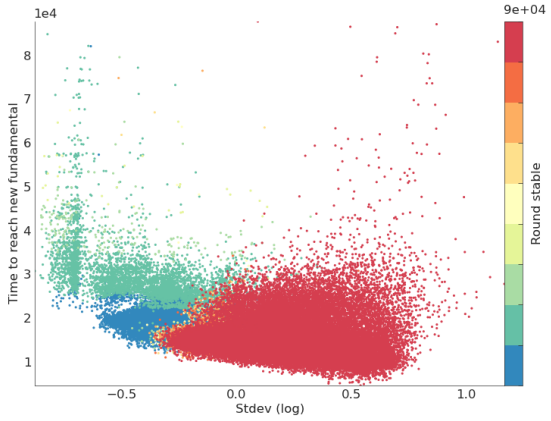
(b) Experiment $\mathcal{D}_9$



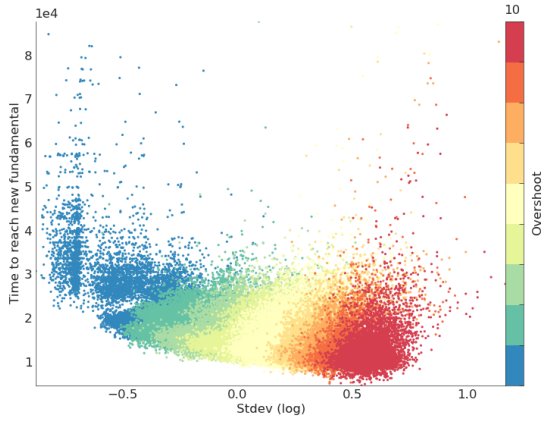
(c) Experiment $\mathcal{D}_{10}$



(d) Experiment $\mathcal{D}_{10}$



(e) Experiment $\mathcal{D}_{11}$



(f) Experiment $\mathcal{D}_{11}$

Figure 28: Scatter plot of $\log f_\sigma$ against f_t with coloring showing f_s and f_o

left scatter plot of $\mathcal{D}_9$, as all the points close to the dotted line has a green/blue color. The right plot of $\mathcal{D}_9$ shows that these simulations did not have any overshoot. It is interesting that both slow simulations which take a long time to reach the new fundamental, as well as simulations who manage to be fast, have no overshoot, and this observation begs the question of whether or not these simulations have some common parameters that make them behave in such a way. These points are extracted by applying filters $\mathcal{F}_2$ and $\mathcal{F}_3$ (see table 9) to the data matrix $\boldsymbol{F}\mathcal{D}_9$ which selects points that lie within a distance of 400 rounds of the diagonal.

Stable before reaching the new fundamental

Most of the simulations falling in region B, meaning that they became stable before reaching the new fundamental price, had no overshoot. However, when the model was allowed to have a large number of chartists, but in $\mathcal{D}_{11}$, a group of simulations did have a small overshoot. These two groups of points were extracted by applying filters $\mathcal{F}_4$ and $\mathcal{F}_5$ (see table 9).

Simulations with overshoot

Filter $\mathcal{F}_6$ selects all the simulation which had an overshoot, as it is interesting to see if the parameters that caused the market to have an overshoot can be somehow differentiated from parameters which cause the market to have no overshoot.

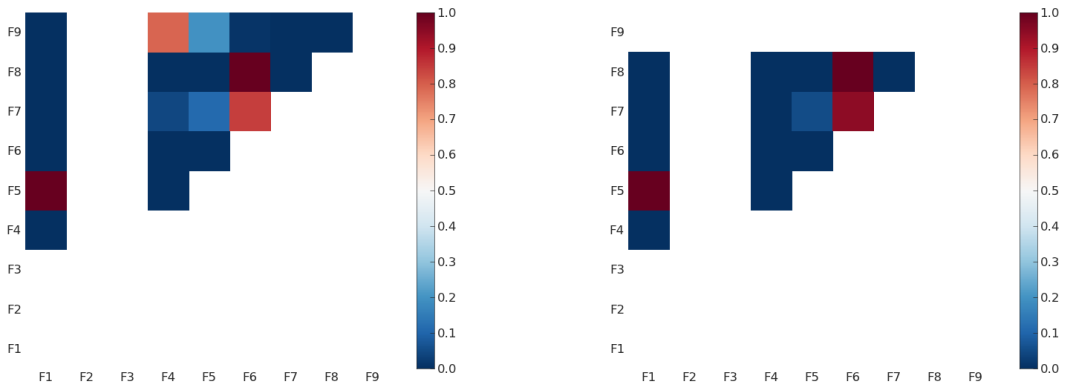
Fast simulations

All three experiments produces a group of simulations which had a quick response to the shock, but took longer to become stable. The simulations are in the column-shaped cluster in figure 27 and the all have relatively low f_{t}-fitness of less than 25000 rounds or so. These data points were extracted using filters $\mathcal{F}_7$ and $\mathcal{F}_8$ (see table 9).

Unstable simulations with non-flickering prices

The final group of simulation that are singles out in this section are those that had very smooth price curves (that is, a small value for f_σ), yet did not manage to become stable. These simulations were selected by filter $\mathcal{F}_9$.

The criteria in table 9 do not prevent a simulation to be selected by different filters. The groups of points are therefore likely to have a non-empty intersection. Using the filters is an the attempt to separate the simulations by their behavior. Hence, the filters should in general not select the same data points. The Jaccard index $J(A, B)$ is calculated between sets A and B and used to determine the overlap between the sets. Figure 29 shows the Jaccard index between the sets created by applying the filters to each of the three data sets. Since the distance matrix is symmetrical, only the upper left part has been plotted. In case that the filter produced an empty set, the Jaccard index is undefined, resulting in white squares.



(a) Experiment $\mathcal{D}_{10}$ (b) Experiment $\mathcal{D}_{10}$

Figure 29: Jaccard index between the sets of data points extracted by each filter.

$$J(A, B) = \frac{|A \cap B|}{\min|A|, |B|} \tag{19}$$

Tables 10 and 11 show the fitness and parameter arithmetic means for each of the nine groups selected by the filters for $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$. Since the average parameters of the filters when applied to $\mathcal{D}_9$ did not differ significantly, nothing of interest could be derived from the data, and the table for $\mathcal{D}_9$ has therefore been moved to the appendix for reference.

When the number of chartists was fixed as in experiment $\mathcal{D}_{10}$, the simulations with overshoot (picked out by filter F6) has an average overshot of $\mathrm{E}_{\mathcal{F}6}[f_\mathrm{o}] = 4.1$ ticks. These markets had comparatively few, but fast market makers ($\mathrm{E}_{\mathcal{F}6}[N_m] = 67.1$ and $\mathrm{E}_{\mathcal{F}6}[\lambda_{m,\mu}]$), and fast chartists ($\mathrm{E}_{\mathcal{F}6}[\lambda_{c,\mu}] = 78.3$). Figure 29 shows that F7 and F8 mostly contain points that are also contained in F6. Both F7 and F8 a lower average overshoot, and this is caused by the markets to have slower chartists, and fewer, slower market makers. When the number of chartists were varied as in experiment $\mathcal{D}_{11}$ (and with a constant of $N_m = 52$ market makers), the average latency of the chartists $\mathrm{E}_{\mathcal{F}6}[\lambda_{c,\mu}]$ did not differ from the other groups. Instead, the biggest difference was that markets with overshoot on average had a high number of chartists. As for the market maker latency, it was smaller than any of the other groups. On the the other hand, markets with no overshoot on average had a large number of marker makers when N_c is fixed to $N_c = 150$, although the market makers were not particularly fast. Markets with no overshoot also had the lowest average number of chartists among the nine groups.

13.2 Clustering with mixture of Gaussians

In this section, the focus is shifted from looking at population wide statistics to analysis sub-groups within each population. Whereas the previous sections showed that there do indeed exist statistical relationships between the latency of the agents and the behavior

ID	Target simulations	Filter criteria
$\mathcal{F}_1$	Fast and stable (but maybe flickering)	$f_\mathrm{t} < 12000$, $f_\mathrm{s} < 12000$, $\log f_\sigma > 0$
$\mathcal{F}_2$	Slow, stable and not flickering (diagonal)	$\lvert f_\mathrm{t} - f_\mathrm{s}\rvert < 400$
$\mathcal{F}_3$	Fast and stable and not flickering (diagonal)	$\lvert f_\mathrm{t} - f_\mathrm{s}\rvert < 400$
$\mathcal{F}_4$	Stable before reaching fundamental, no overshoot	$f_\mathrm{s} < f_\mathrm{t}$, $f_\mathrm{o} = 0$
$\mathcal{F}_5$	Stable before reaching fundamental, with overshoot	$f_\mathrm{s} < f_\mathrm{t}$, $f_\mathrm{o} > 0$
$\mathcal{F}_6$	Has overshoot	$f_\mathrm{s} > f_\mathrm{t}$, $f_\mathrm{o} > 0$
$\mathcal{F}_7$	Fast response, quick to stabilize	$1000 < f_\mathrm{t} < 25000$, $20000 < f_\mathrm{s} < 40000$
$\mathcal{F}_8$	Fast response, slow to stabilize	$1000 < f_\mathrm{t} < 25000$, $40000 < f_\mathrm{s} < 75000$
$\mathcal{F}_9$	Smooth prices with a small overshoot, yet unstable	$e^{-0.5} - 0.1 < \log f_\sigma < e^{-0.5} + 0.1$

Table 9: Filter IDs and fitness-regions

	$\mathcal{F}_1$	$\mathcal{F}_2$	$\mathcal{F}_3$	$\mathcal{F}_4$	$\mathcal{F}_5$	$\mathcal{F}_6$	$\mathcal{F}_7$	$\mathcal{F}_8$	$\mathcal{F}_9$
$\lambda_{c,\mu}$	N/A	N/A	N/A	123.8	121.5	78.3	116.9	104.2	95.0
$\lambda_{c,\sigma}$	N/A	N/A	N/A	6.5	6.5	9.0	7.1	8.7	8.7
$\lambda_{m,\mu}$	N/A	N/A	N/A	73.9	75.8	39.2	73.3	59.6	35.9
$\lambda_{m,\sigma}$	N/A	N/A	N/A	5.6	6.0	9.6	6.6	8.7	9.2
N_m	N/A	N/A	N/A	126.5	125.6	67.1	121.2	98.0	81.2
f_o	N/A	N/A	N/A	0.0	1.0	4.1	1.8	2.1	0.2
f_s	N/A	N/A	N/A	17025.9	15920.1	81539.2	27387.5	59444.5	24392.3
f_σ	N/A	N/A	N/A	0.6	0.7	1.2	0.7	0.9	0.6
f_t	N/A	N/A	N/A	20405.2	18724.9	15504.3	18725.1	16840.5	31017.6
Count	0	0	0	8486	25574	47493	5379	2528	399

Table 10: Means of $\mathcal{F}_1$ through $\mathcal{F}_9$ for $\mathcal{D}_{10}$

	$\mathcal{F}_1$	$\mathcal{F}_2$	$\mathcal{F}_3$	$\mathcal{F}_4$	$\mathcal{F}_5$	$\mathcal{F}_6$	$\mathcal{F}_7$	$\mathcal{F}_8$	$\mathcal{F}_9$
$\lambda_{c,\mu}$	N/A	N/A	N/A	30.4	32.3	33.3	34.3	35.1	35.8
$\lambda_{c,\sigma}$	N/A	N/A	N/A	9.3	9.9	4.6	7.2	6.6	9.0
N_c	N/A	N/A	N/A	19.0	26.9	154.0	46.3	54.6	41.1
$\lambda_{m,\mu}$	N/A	N/A	N/A	67.2	55.0	38.9	50.0	47.3	46.0
$\lambda_{m,\sigma}$	N/A	N/A	N/A	8.4	11.6	22.9	17.5	19.5	14.7
f_o	N/A	N/A	N/A	0.0	1.0	14.6	2.0	2.6	0.0
f_s	N/A	N/A	N/A	15678.2	15075.3	87971.4	31954.9	63613.9	21851.4
f_σ	N/A	N/A	N/A	0.7	0.8	3.6	0.9	1.0	0.6
f_t	N/A	N/A	N/A	19965.3	18765.4	13685.0	17265.5	16449.5	31048.9
Count	0	0	0	71329	31377	84597	349	3014	1706

Table 11: Means of $\mathcal{F}_1$ through $\mathcal{F}_9$ for $\mathcal{D}_{11}$

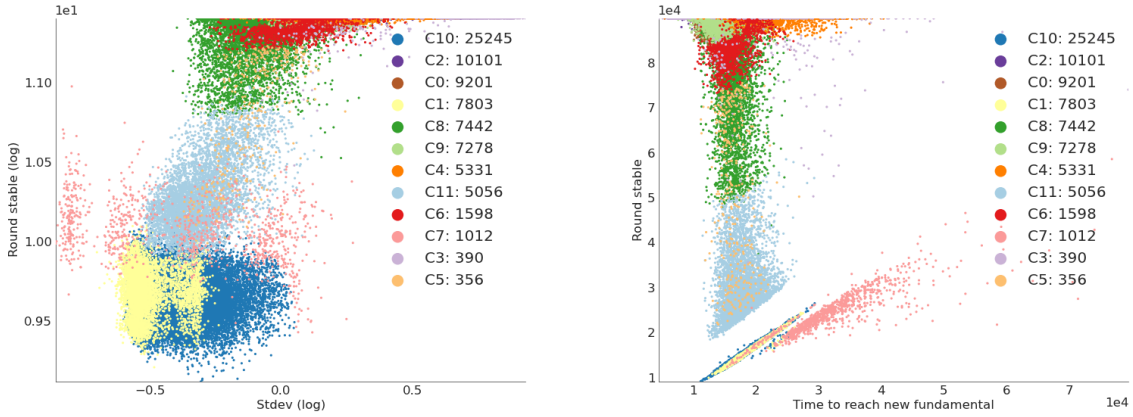
of the model, each discovered correlation was calculated over the entire population. Although the correlations reveal overall tendencies of the model behavior when changing a single parameter, little can be said about how the various parameters interact to determine model behavior. For instance, even though prediction of, say a negative correlation between f_t was $\lambda_{c,\mu}$ was found, there might be configurations of the model in which faster chartists were actually beneficial to the market.

One way to approach this problem is to see whether or not there exists relationships between partitions in the parameter space to partitions in the fitness space. As in section 7.5, the first step is to partition space, since each partition can be interpreted in terms of the model behavior. For instance, a partition covering the lower left half of the 2-dimensional space in figure 27 would encompass all the simulations which had a fast response time and became stable quickly (no matter if they had prices that flickered within the stability margin or not).

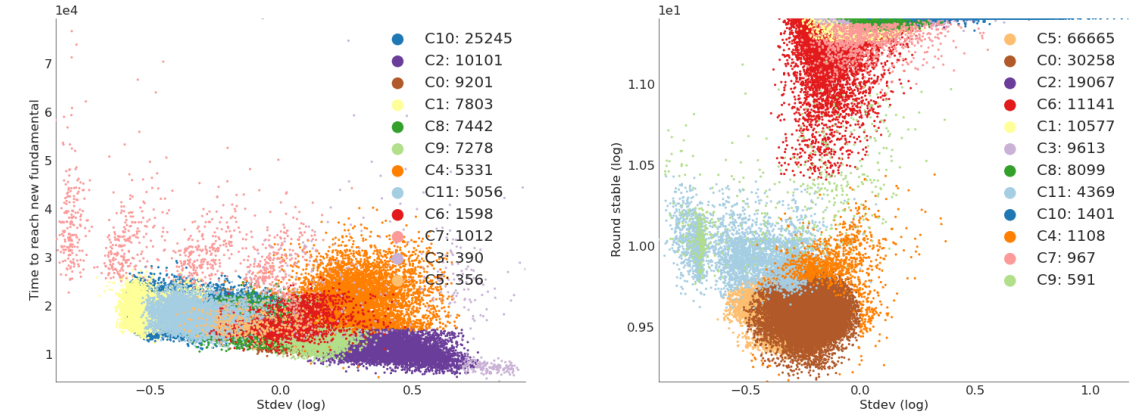
In order to investigate this, a Gaussian mixture model (GMM) was used to find clusters in the fitness space. All four fitness measures were used for the clustering. After discarding simulations with undefined fitness values and removing outliers, the data set $\mathcal{D}_{10}$ contained 80813 data points, whereas $\mathcal{D}_{11}$ contained 187310 data points. The large number of data points and the low dimensionality made it possible to allow a each Gaussian component to have a full covariance matrix, giving the model a high level of flexibility. A mixture model with 12 components was calculated for each of $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$. Figure 30 shows scatter plots of the . Scatter plots of $\mathcal{D}_{11}$ are quite similar to those of $\mathcal{D}_{10}$ and have therefore been omitted. Tables 12 and 13 show the mean values of the fitness and parameters calculated over each cluster in $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$.

Table 12 and 13 shows the mean of the fitness values and parameters over each cluster. The tables are sorted by the average value of f_o. $\mathcal{O}$ is used to denote the set of outliers, that is, markets with $f_\mathrm{o} > 10$

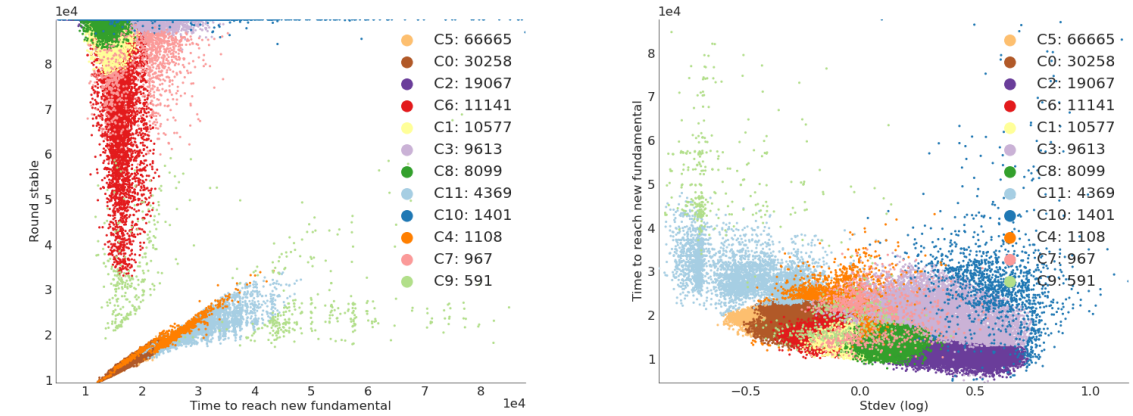
A general note of the results di



(a) $\mathcal{D}_{10}$ (b) $\mathcal{D}_{10}$



(c) $\mathcal{D}_{10}$ (d) $\mathcal{D}_{11}$



(e) $\mathcal{D}_{11}$ (f) $\mathcal{D}_{11}$

Figure 30: GMM cluster assignments in $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$. Note that there is no correspondence between the displayed colors in $\mathcal{D}_{10}$ and $\mathcal{D}_{11}$, as colors were assigned randomly by the algorithm.

	f_o	f_s	f_σ	f_t	$\lambda_{c,\mu}$	$\lambda_{c,\sigma}$	$\lambda_{m,\mu}$	$\lambda_{m,\sigma}$	N_m	Count
$\mathcal{C}_1$	0.0	16301.2	0.6	19217.4	127.2	6.2	78.4	5.2	132.2	7803
$\mathcal{C}_7$	0.3	24383.1	0.7	32005.5	87.0	9.5	25.8	10.5	66.7	1012
$\mathcal{C}_{10}$	1.0	15832.8	0.7	18602.8	121.9	6.5	76.3	6.0	126.3	25245
$\mathcal{C}_8$	2.0	81599.7	0.9	16333.6	101.8	8.7	56.4	9.2	91.3	7442
$\mathcal{C}_{11}$	2.0	30515.0	0.7	17822.9	114.2	7.3	71.7	7.0	117.5	5056
$\mathcal{C}_0$	3.0	89015.0	1.1	14687.5	89.0	9.2	39.5	10.0	66.8	9201
$\mathcal{C}_5$	3.0	51306.4	0.9	16508.7	102.2	9.0	59.2	8.8	96.0	356
$\mathcal{C}_6$	3.0	83914.2	1.1	16820.9	92.3	9.2	40.6	10.0	72.4	1598
$\mathcal{C}_9$	4.0	89496.0	1.2	14459.8	77.7	9.6	29.9	10.4	56.7	7278
$\mathcal{C}_4$	5.0	89896.6	1.4	22034.2	62.4	9.4	21.5	10.3	51.6	5331
$\mathcal{C}_3$	6.7	86440.9	1.8	18692.7	54.9	9.2	26.2	10.5	38.0	390
$\mathcal{C}_2$	7.9	89996.1	1.6	11574.7	36.4	9.2	26.4	9.6	36.4	10101
$\mathcal{O}$	11.5	89998.8	2.2	8835.1	17.8	9.3	24.3	8.5	19.4	740

Table 12: Cluster means ($\mathcal{D}_{10}$)

XXX NOT FINISHED. WRITE ABOUT THE CLUSTERS THAT HAVE SOME INTERPRETABLE VALUE. IT DOES NOT HAVE TO BE ALL OF THEM. $\mathrm{Var}_{\mathcal{C}8}[f_s]$ and $\mathrm{Var}_{\mathcal{C}11}[f_s]$ and $\mathrm{Var}_{\mathcal{C}5}[f_s]$ are large. The points in this cluster have parameters which

Chartist latency and market response time

As was noted earlier, the evolution of $\lambda_{c,\mu}$ and f_t indicates that slow chartists made the market slow, and fast chartists made the market fast. C1 is the cluster with the $\mathrm{E}_{\mathcal{C}1}[f_t]$

As table 12 shows, $\mathcal{D}_{10}$ only contained 740 cases of a market with an overshoot of more than ten ticks. The most noticeable thing about the parameters of the markets with a large overshoot is that the average latency of both types of fast traders are very small. Compared to the fast in markets with no overshoot assigned to $\mathcal{C}_1$, the chartists in $\mathcal{O}$were almost seven times faster, while the market makers were a bit over three times faster. Furtherore, the markets in $\mathcal{O}$ took only 8835 rounds on average, while the markets in $\mathcal{C}_1$ took more than twice. Hence, this is a further illustration of the speed/stability trade-off.

Comparing $\mathcal{C}_0$ and $\mathcal{C}_5$, it is seen that the largest difference is that $\mathrm{E}_{\mathcal{C}5}[f_s]$ is around 38000 rounds smaller than $\mathrm{E}_{\mathcal{C}0}[f_s]$. Collaborating this with the facts that $\mathrm{E}_{\mathcal{C}5}[f_\sigma] < \mathrm{E}_{\mathcal{C}0}[f_\sigma]$ and $\mathrm{E}_{\mathcal{C}5}[f_o] = \mathrm{E}_{\mathcal{C}0}[f_o]$, it becomes clear than $\mathcal{C}_0$ contains markets which did non become stable due to large price flickering. Comparing the average parameters of the two clusters, it is seen that the two main differences are the speed and number of the market makers. $\mathcal{C}_5$ has around 30% more market makers than $\mathcal{C}_0$, while the market makers in $\mathcal{C}_0$ are around 33% faster than the market makers in $\mathcal{C}_5$. This suggests that a large number of slower market makers does a better job of reducing price flickering than a smaller number of fast market makers. The same phenomenon can be observed by comparing clusters $\mathcal{C}_8$ and $\mathcal{C}_{11}$.

Clusters $\mathcal{C}_1$ and $\mathcal{C}_7$ both contain markets with never left the stability margin after entering the first time. In other words, stable markets with little or no overshoot, and very little price flickering. It is interesting to compare the response time of the two clusters, and the markets in $\mathcal{C}_7$ took 40% longer to reach the new fundamental price than the markets in $\mathcal{C}_1$. The cause for this appears to be that the markets in $\mathcal{C}_7$ has market makers that were three times faster than the market makers in $\mathcal{C}_1$. Furthermore, the markets in $\mathcal{C}_1$ had around twice as many market makers than the markets in $\mathcal{C}_7$. These results suggest that the presence of a relatively small group of fast market makers actually made the market respond *slower* to the fundamental shock. The markets in $\mathcal{C}_7$ had just as market makers as the markets in $\mathcal{C}_5$, and the chartists were just as fast in $\mathcal{C}_7$ as in $\mathcal{C}_5$. The reason for the markets in $\mathcal{C}_0$ to respond almost twice as fast as the markets in $\mathcal{C}_7$ is therefore that the market makers in $\mathcal{C}_0$ are around 34% slower than the market makers in . This result is consistent with the finding that faster market makers tend to slow down the response of the market.

A group of markets which have not yet been discussed are those in which the trade price never reaches the new fundamental. Since f_t is undefined in this case, such data points were removed in order to be able to calculate statistics over f_t. However, such markets are interesting because the stock is essentially traded at more than what it is true worth (according to the fundamentalists), and hence they are markets in which the stock is overvalued. An interesting question is which parameters cause such a thing to occur in the market. As was previously shown, market makers have a tendency of slowing down the response to the shock in the fundamental price, which is the same as saying that the market makers increase the duration of the temporary overvaluation. It seems reasonable to assume that what causes the market to remain frozen in a state of overvaluation is the same that causes temporary overvaluation, mainly the presence of a large number of market makers, or the presence of fast market makers. Figure 31 verifies this hypothesis. The figure shows that market in which permanent[8] overvaluation occurs, N_m is correlated with $\lambda_{m,\mu}$. Hence if a market has a large number of market makers, overvaluation can occur even with relatively slow agents. On the other hand, when then market makers are fast, it only takes a few agents to cause prolonged overvaluation.

Table 13 shows the the data from the experiment in which the number of chartists were varied while the number of market makers was kept constant. The clustering was not able to distinguish behavior in terms of the chartist latency, as the average of $\lambda_{c,\mu}$ is more or less the same in all the clusters. Instead, the parameter with the most noticeable impact on the overshoot is N_c. A big difference from the results in table 12 is that the outliers in $\mathcal{D}_{11}$ have a large average overshoot. Some of the markets in this groups manage to recover from the overshoot, while some of them stabilize at a level below the fundamental. Markets that do not return to the fundamental or stabilize at a lower price continue to plummet, and such markets are said to have crashes. Since the duration of the simulation is only a few minutes of real-time, the observed crashes are flash crashes.

Although table 13 seems to indicate that the chartist latency has little influence on when the market will crash, this is not quite true, and simply a result of limits in trying

[8]Permanent for the duration of the simulation, that is

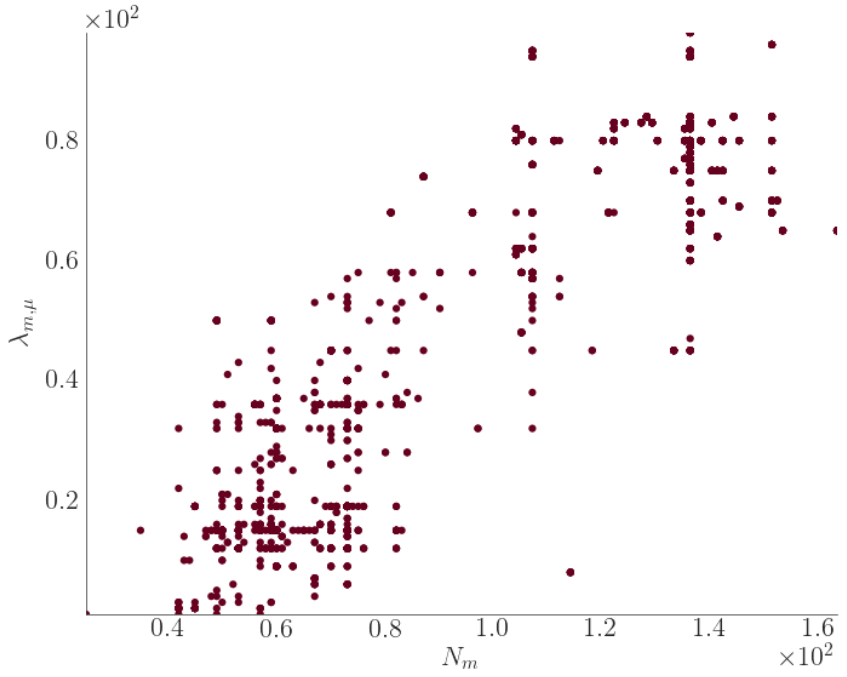


Figure 31: Scatter plot of N_m and $\lambda_{m,\mu}$ for markets with lasting overvaluation. $\text{Corr}(N_m, \lambda_{m,\mu}) = 0.897$

.

	f_o	f_s	f_σ	f_t	$\lambda_{c,\mu}$	$\lambda_{c,\sigma}$	N_c	$\lambda_{m,\mu}$	$\lambda_{m,\sigma}$	Count
$\mathcal{C}_5$	0.0	15270.3	0.7	19209.4	30.0	9.3	17.5	68.7	7.9	66665
$\mathcal{C}_{11}$	0.0	21182	0.6	29334.9	35.5	8.9	40.2	46.6	14.5	4369
$\mathcal{C}_0$	1.0	14897	0.8	18523.0	32.2	10.0	25.9	55.6	11.3	30258
$\mathcal{C}_4$	1.0	19666	0.9	25087.1	35.3	7.3	55.5	39.0	18.9	1108
$\mathcal{C}_9$	1.0	30399	0.7	35789.7	34.4	7.8	45.6	43.6	17.9	591
$\mathcal{C}_6$	2.0	79198	0.9	15620.1	37.5	6.1	63.5	47.3	20.6	11141
$\mathcal{C}_7$	2.6	78771	1.0	20524.6	37.7	5.6	63.8	38.5	22.6	967
$\mathcal{C}_1$	3.0	88210	1.0	14225.8	37.9	4.6	88.0	44.2	23.3	10577
$\mathcal{C}_8$	4.0	89633	1.1	13452.7	33.6	4.4	103.9	44.4	22.7	8099
$\mathcal{C}_3$	5.7	89842	1.4	20087.3	34.6	4.5	136.1	29.6	23.9	9613
$\mathcal{C}_{10}$	7.0	89935	1.8	27587.6	32.8	4.2	165.8	25.9	24.3	1401
$\mathcal{C}_2$	7.0	89982	1.5	11691.4	32.3	4.4	154.4	37.6	22.7	19067
$\mathcal{O}$	40.5	89951	9.9	10434.7	29.2	4.0	255.5	36.4	23.5	23454

Table 13: Cluster means ($\mathcal{D}_{11}$)

do group models by behavior using a clustering algorithm. Figure 23 showed that the chartist latency does indeed have a large role to play for the market stability, but only when the number of chartists is more than 200 or so.

14 Ratios

The previous sections showed that the number of fast traders and their latencies were important for deciding how the market is going to respond to the fundamental shock. However, in terms of modeling, the number of agents is not particularly interesting as the number of agents in real markets is surely much larger. The same argument can be somewhat applied to the agent latency, as delays in real markets . To re-iterate: the main purpose of this model is to allow for agents of different types to have quantifiable speed differences. The model

The findings in the previous

Crashes never occur in markets with only fast market makers or fast chartists, no matter how many agents are active and how fast they are. Even markets with

15 Summary of results

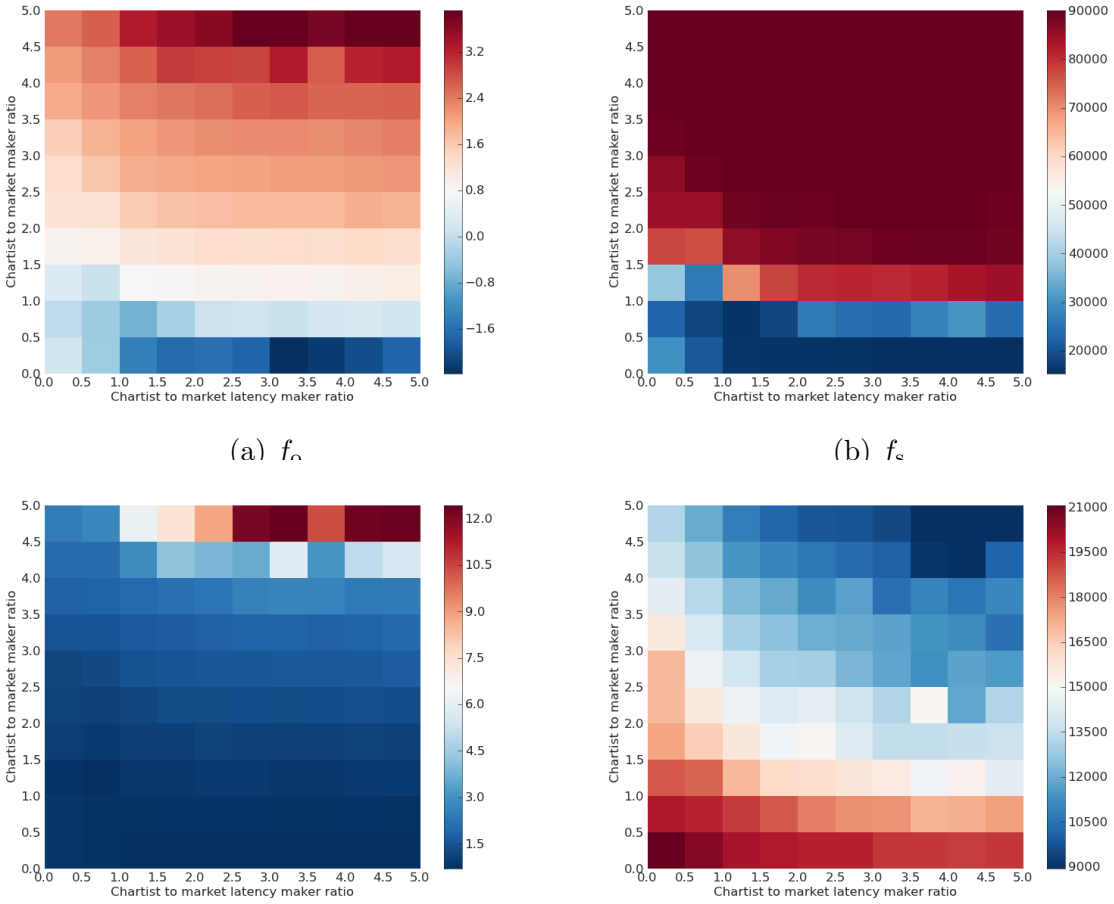
The results showed that both types of HFT agents influence the market in positive and negative ways. This section contains a summary of the results.

Agent roles Each of the three agent types can be said to play a particular role in the market. The fundamentalists are responsible for driving the market back towards the fundamental price. The market makers stay constantly in the market to fill orders submitted at sub-optimal prices. Finally, the chartists adds another driving force, the strength of which does not depend on the value of the fundamental, and the direction of which may or may not be in the direction of the fundamental.

Lower agentlatency increases the impact of agent strategy The latency was found to have a big impact on the market in such a way that fast agents have a larger influence, according to their role as described above, than do slow traders. The agency latency was most important in markets containing a large number of fast traders, as the cumulative impact of a small group of fast traders could not be measures irregardless of how fast these agents are.

Speed/stability trade-off A trade-off between the stability of the market in terms of overshoot and price flickering versus the speed with reach the market could respond to the shock in the fundamental was found. Thus, the fastest markets were also those with the largest overshoot, and vice versa.

Market makers add market stability Market makers have a stabilizing effect on the markets and make the market more robust to the impact of the chartists. Thus markets with market makers tend to have less flickering prices than markets without market makers.



(a) f_{α} (b) f_{ε}

(c) f_{σ} (d) f_{t}

Figure 32: Image plots for model fitness as a function of ρ_{A} and ρ_{λ}

Market makers slow market response Market makers increase the time it takes for the market to reach the true fundamental price after the shock.

Chartists cause price flickering Price flickering increases with the number of chartists in the market.

Chartists increase market response High speed chartists influence the market so as to decrease the duration of the period of time in which the stock price stock price is overvalued by reducing the time required for the price to return to the new fundamental price after the negative shock. This was seen both in table 13 and in table

Ratio of agent types influences the market More important than the number of fast traders in the market was the ratio between the number of chartists and the number of market makers. The market makers proved to be quite efficient in counteracting the both the positive and negative impact of the chartists. The more chartists per market maker were present in the market, the faster the market responded to the fundamental, but at the cost of an increased risk of a market crash.

Ratio of agent latencies influences the market The ratio between the latency of the chartists and the latency of the market makers had a similar impact as the ratio between the number of agents. Markets in which the chartists were several times faster than the market makers were fast but unstable. When market makers were fast, the likelihood of misvaluation increased.

Misvaluation of the stock price The market was shown to respond slowly to the shock in the fundamental price when the market contained market makers. This overvaluation could be permanent in the influence of the market makers was big enough. Figure 31 showed how the interplay of N_m and $\lambda_{m,\mu}$ caused the stock price to become overvalued. Undervaluation also occurred, but less frequently.

Occurrence of flash crashes Flash crashes were seen to happen when the ratio of the number of chartists to the number of market makers, ρ_{A} was large when the ratio between the latencies ρ_{λ} was also large. Specifically, it was found that flash-crashes could occur in markets in which the number of chartists was four times or greater that of the number of market makers. Furthermore, crashes occurred four times more often when the chartists were faster than the market makers. In markets where these conditions were not present, crashes were very rare.

Part V
Discussion

16 Co-location and communication delays

Information technology has made it possible to trade on markets anywhere on the globe with a latency that is so low that it is not registered by humans. According to [26], trading across the transatlantic ocean can currently be accomplished with a delay of around 60ms, which is around 10 times faster than the shortest time that a trained person needs to react to an event that requires cognitive effort to be resolved.

Some algorithmic traders use high performance computing techniques such as dedicated FPGA processing [41] to analyze market data in a few dozen microseconds. Co-location, in which trading firms pay premium rates for a location close to the market has become a more or less (as it were: [42]) accepted standard requirement for firms that want to do high frequency trading [32, 11, 50]. Other algorithmic traders rely less on speed and use regular communications channels. Hence, in the real world, algorithmic agents trade with delays stemming from network latency and computation time, varying from moderately fast (a few hundred milliseconds) to extremely fast (a few milliseconds, or less).

A latency of 60ms, which as already mentioned is the current cutting edge trading speed over the Atlantic Ocean [60, 45], corresponds to 60 rounds in the model. A round trip from New York to Chicago takes 8.5 milliseconds[1], corresponding to 8 or 9 rounds in the simulation. Hence, the delays of the fast traders in the model approximate the delays of actual algorithmic traders. We therefore believe that the findings showing that the right combination of fast traders can cause the market to crash is rather interesting. Furthermore, we believe that this work contributes to the ongoing discussion for and against high frequency trading and the accompanying technological arms-race.

17 Market crashes

The results showed that the model is capable of reproducing fast market crashes, also known as flash-crashes. This section offers an explanation of what causes such crashes to occur.

It is somewhat intuitive that HFT chartists should be suspected of having an influence on the market such that the market become more likely to crash. The results did indeed confirm this, as it was shown that a negative correlation exists between the number of chartists active in the market, and the size of the overshoot.

It is conceivable that the market makers also contribute to the market crashing, since the market makers also ignore the true fundamental price. Indeed, the results showed that the market does not crash from having fast chartists alone. The results also showed that the market will not crash from having only fast market makers either. Instead, both types of fast traders were required to make the market crash. The following text will

provide an explanation as to why this is so. It was found that markets containing no market makers will almost never crash. Without the presence of market makers, even a market saturated with chartists will eventually return to the fundamental price, as the force of the initial downtrend created by the fundamentalists dissipates.

Case with no fast traders

The shock to the fundamental creates a drive in the market for falling prices, due to the presence of the fundamentalists. The fundamentalists have a large delay, and the downwards drive is therefore initially small, as most of the fundamentalists fail to observe that the shock has happened. As the fundamentalists begin to observe the shock, they start submitting sell orders at lower prices, as they believe that the stock is no longer worth the price at which they were previously willing to sell. Hence, the number of sell orders starts to increase.

As for the buy side of the order book, the number of new buy orders starts to fall, as the fundamentalists start to register the shock. The buy orders that were previously submitted by slow traders at prices slightly below the old fundamental are not canceled, as the model assumes that the fundamentalists are too slow to register the change in the fundamental. Furthermore, in order to simulate an order book with a long trade history, the order book was initialized with a large number of market orders with a normal price distribution centered around the initial fundamental. These buy orders provide matches for the increasing number of sell orders, and the traded price begins to drop. If the market has no fast traders, the traded price will eventually reach the new fundamental, and stay within the stability margin. Thus, in the rather simple case where the market only contains fundamentalists, crashes do not occur.

Case with chartists but no market makers

When adding chartists, the market starts to behave in a different manner. The chartists do not use any information about the true true fundamental price, but are instead only concerned with the actual traded price. After the shock, the fundamentalists start submitting bids to sell at lower prices, Depending on the parameters of the chartists, some chartists will interpret this as a downtrend, while others will not. The chartists that detect a downtrend will start submitting bids to sell at a lower price, as they believe the price will continue to drop. The chartists that did not detect a trend will remain inactive. The sell orders submitted by the chartists are matched by previously existing buy market orders at lower prices. Hence, the chartists add to the force that drives the traded price down by submitting sell orders at lower prices. However, since the only active traders in the market are fundamentalists and chartists, the supply of buy orders at prices lower than the new fundamental are limited. The chartists that detected a downtrend will exclusively place sell orders, and the fundamentalists will rarely submit buy orders at prices much lower than the fundamental. When the supply of buy orders at prices below the new fundamental dries out, the execution price will not drop further. The chartists that detected a downtrend will continue to submit sell orders for as long as they believe that the trend continues, but the only new buy orders are submitted by the fundamentalists.

As some of the fundamentalist buy orders are placed a few ticks below the fundamental price, the execution price will flicker, but always in a region close the true fundamental price.

Case with chartists and market makers

When the market also contains market makers, the situation is quite different. Like the chartists, the market makers ignore the fundamental price. Instead they submit buy and sell orders just above and below the best buy and sell prices existing in the order book at the time that the market maker requested the market information. The market maker strategy is such that it will always try to follow a narrowing spread, in order to stay competitive. On the other hand, if the market maker discovers that the spread is widening, the agent will attempt to avoid buying/selling at a higher/lower price than necessary. The agent therefore tries to follow the widening spread by submitting buy/sell orders at lower/higher prices.

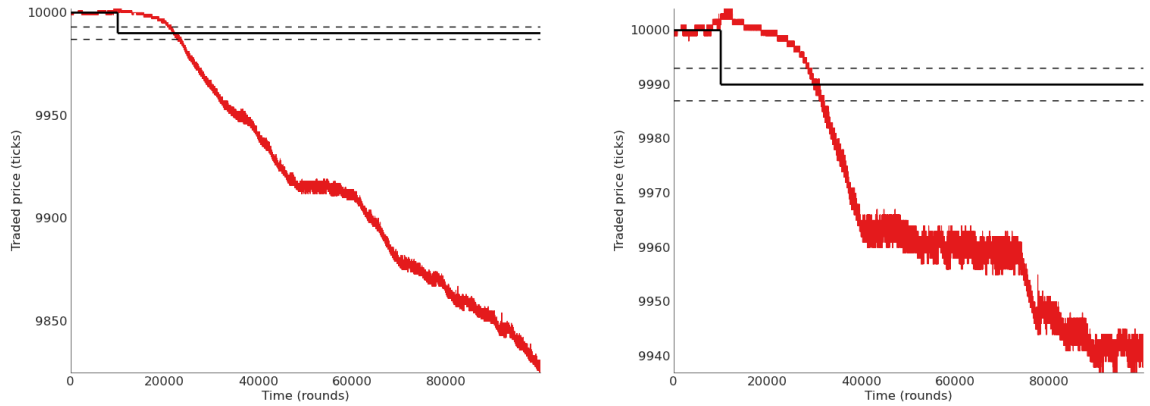
When the sell price starts to drop after the shock due to the activity of the fundamentalists and the chartists, the market maker will try to stay competitive on the sell side be decreasing its own sell price. If the decrease in the sell price is large enough to make the spread smaller than what the agent is prepared to risk, the market maker submits a new sell order with as low a price than its strategy allows.

On the other side of the order book, the best buy price starts to drop as the sell orders submitted by the chartists start to eat away at the existing buy orders. If the market maker orders are among the orders that match the chartist orders, the market makers request the latest market information and use it to submit new buy orders. If the market maker orders were not matched by chartist orders the market makers will cancel their existing buy order and submit a new one at a lower price in order to stay a competitive buyer. In any case, the market maker will eventually start submitting buy orders at a lower price than before. Hence, the market makers provide the market with a new supply of buy orders, the prices of which can be arbitrarily low. As these buy orders are filled by sell orders, initially by both fundamentalists and chartists but eventually solely by chartists, the traded price will drop, and the chartists will continue detecting a trend and continue to drive the market down into a crash.

18 Fast market makers causing overvaluation of the stock

In the previous part it was shown that overvaluation, that is, a discrepancy between the fundamental price and the traded price of the stock, occurs in some markets.

As was shown in figure XXX, both an increased number of market makers and market makers with lower latencies tended to push the market towards responding slowly to the change in the fundamental price. A slower response means that a discrepancy between the traded price and the true fundamental price persists for longer. In the case of a negative



(a) (b)

Figure 33: Two examples of market crashes

shock ($\eta < 0$), the fundamental price will be lower than the traded price, which is the same as saying that the stock is overvalued.

While this does not pose a problem in the current market model, it is easy to see how such an inefficiency can be exploited. In the current model, the fast traders do not know the fundamental price, and therefore have no concept of the true value of the stock. Instead the fast agents trade only by observing the order book, whereas it is the slow traders who know the fundamental. However, if fast traders were to obtain information about the fundamental. such a trader would be able to use this knowledge to deliberately trade at below optimal prices, in expectation that the traded price will eventually return to the fundamental price. In the case of a sudden negative shock in the fundamental, such fast fundamentalists would be able to be reasonably safe by short-selling, as aggregated action of all the active fundamentalists will most likely drive the traded price down towards the new and lower fundamental. In the case of a sudden positive shock to the fundamental, the fast fundamentalists would be able to profit by buying up large volumes of stock which they would then sell after the traded price has descended fully or partly to the higher fundamental price. The profit generated by such a strategy would be at the expense of the slow fundamentalist, and is exactly how the fast and informed (e.i., informed of the fundamental price) agents in [] managed to turn a profit. This work has shown that the presence of fast market makers make the scenario assumed in that work more likely to happen.

We find it reasonable to speculate that the increased response time, or prolonged overvaluation of the stock price, is not only due to the nature of the market making strategy, but due to the the market making strategy being used by traders *who are very fast*. When the market makers are very fast, their behavior will be more alike, since they act on almost the same information, and hence their aggregate behavior will combine into a consistent force that has a strong influence on the market. In the field of multi-agent simulation, this phenomenon is known as strategy crowding [38], and basically means

that agents using similar strategies and/or observe the same environment tend to act in similar ways. In this model. agents generally *do not* observe the same information at the same time, and strategy crowding therefore seems less likely to happen. However, when the agents become fast, the will increasingly start to observe the same information, or at least information which is only very slightly delayed and therefore very similar. This is similar to the argument raised in [], but whereas the model in that work assumed identical strategies, this work has shown that

18.1 Frequency of crashes

Crashes did not occur often particularly. Even during experiment $\mathcal{D}_{11}$, in which the genetic algorithm ended up prioritizing the market response times, and generate a large number of genes with many chartists and very fast market makers, the market had an overshoot of over 25 tick in just less than 0.2% of the cases[9]. In experiment $\mathcal{D}_{10}$, not a single case of markets with an overshoot of over 17 ticks was generated.

[9]The simulation was run around $4 \cdot 10^5$ times, and 7989 of these had $f_o > 25$

19 Additional tables

20 Dataset 1

Write your Appendix content here.

References

[1] Jerry Adler. Raging bulls: How wall street got addicted to light-speed trading. `http://www.wired.com/business/2012/08/ff_wallstreet_trading/all/`. Accessed: 2014-01-17.

[2] Jerry Adler. Raging bulls: How wall street got addicted to light-speed trading. *Wired Magazine*, 2012.

[3] Franklin Allen and Risto Karjalainen. Using genetic algorithms to find technical trading rules. *Journal of financial Economics*, 51(2):245–271, 1999.

[4] Yakov Amihud and Haim Mendelson. Dealership market: Market-making with inventory. *Journal of Financial Economics*, 8(1):31–53, 1980.

[5] Volkan Arslan, Patrick Eugster, Piotr Nienaltowski, and Sebastien Vaucouleur. Scoop–concurrency made easy. In *Dependable Systems: Software, Computing, Networks*, pages 82–102. Springer, 2006.

[6] Bruno Biais and Paul Woolley. High frequency trading. *Manuscript, Toulouse University, IDEI*, 2011.

[7] Jeff A Bilmes et al. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *International Computer Science Institute*, 4(510):126, 1998.

[8] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

[9] Tobias Blickle and Lothar Thiele. A comparison of selection schemes used in genetic algorithms, 1995.

[10] Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1–8, 2011.

[11] Jonathan Brogaard, Björn Hagströmer, Lars L Norden, and Ryan Riordan. Trading fast and slow: Colocation and market quality. *Manuscript*, 8:25, 2013.

[12] Alain Chaboud, Erik Hjalmarsson, Clara Vega, and Ben Chiquoine. Rise of the machines: Algorithmic trading in the foreign exchange market. *FRB International Finance Discussion Paper*, (980), 2009.

[13] Carl Chiarella, Giulia Iori, and Josep Perelló. The impact of heterogeneous trading rules on the limit order book and order flows. *Journal of Economic Dynamics and Control*, 33(3):525–537, 2009.

[14] Silvano Cincotti, Sergio M Focardi, Linda Ponta, Marco Raberto, and Enrico Scalas. The waiting-time distribution of trading activity in a double auction artificial financial market. In *The Complex Networks of Economic Interactions*, pages 239–247. Springer, 2006.

[15] Paul Davidsson. *Multi agent based simulation: beyond social simulation*. Springer, 2001.

[16] Lawrence Davis. Handbook of genetic algorithms. 1991.

[17] Ann Devitt and Khurshid Ahmad. Sentiment polarity identification in financial news: A cohesion-based approach. In *ACL*, 2007.

[18] Alexis Drogoul, Diane Vanbergue, and Thomas Meurisse. Multi-agent based simulation: Where are the agents? In *Multi-agent-based simulation II*, pages 1–15. Springer, 2003.

[19] David Easley and Jon Kleinberg. *Networks, crowds, and markets*, volume 8. Cambridge Univ Press, 2010.

[20] J Doyne Farmer and Spyros Skouras. An ecological perspective on the future of computer trading. *Quantitative Finance*, 13(3):325–346, 2013.

[21] Thierry Foucault, Johan Hombert, and Ioanid Rosu. News trading and speed. *Available at SSRN 2188822*, 2012.

[22] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007.

[23] Nigel Gilbert. Agent-based social simulation: dealing with complexity. *The Complex Systems Network of Excellence*, 9(25):1–14, 2004.

[24] Namrata Godbole, Manja Srinivasaiah, and Steven Skiena. Large-scale sentiment analysis for news and blogs. *ICWSM*, 7, 2007.

[25] David E Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. *Urbana*, 51:61801–2996, 1991.

[26] Anton Golub, John Keane, and Ser-Huang Poon. High frequency trading and mini flash crashes. *Available at SSRN 2182097*, 2012.

[27] Markus Gsell. Assessing the impact of algorithmic trading on markets: A simulation approach. Technical report, CFS working paper, 2008.

[28] Peter Hoffmann. A dynamic limit order market with fast and slow traders. *Available at SSRN 1969392*, 2012.

[29] Cars H Hommes. Heterogeneous agent models in economics and finance. *Handbook of computational economics*, 2:1109–1186, 2006.

[30] Alan J Izenman. *Modern multivariate statistical techniques: regression, classification, and manifold learning*. Springer, 2008.

[31] Kiyoshi Izumi, Fujio Toriumi, and Hiroki Matsui. Evaluation of automated-trading strategies using an artificial market. *Neurocomputing*, 72(16):3469–3476, 2009.

[32] Luke Jeffs. Co-location is the key to faster trading speeds. `http://www.efinancialnews.com/story/2010-06-28/co-location?ea9c8a2de0ee111045601ab04d673622`. Accessed: 2014-01-17.

[33] Neil Johnson. Financial black swans driven by ultrafast machine ecology. *Available at SSRN*, 2012.

[34] Eric Jones, Travis Oliphant, and Pearu Peterson. Scipy: Open source scientific tools for python. *http://www. scipy. org/*, 2001.

[35] Moshe Koppel and Itai Shtrimberg. Good news or bad news? let the market decide. In *Computing attitude and affect in text: Theory and applications*, pages 297–301. Springer, 2006.

[36] Blake LeBaron. Agent-based computational finance: Suggested readings and early research. *Journal of Economic Dynamics and Control*, 24(5):679–702, 2000.

[37] Blake LeBaron. A builders guide to agent-based financial markets. *Quantitative Finance*, 1(2):254–261, 2001.

[38] Charles M Macal and Michael J North. Tutorial on agent-based modeling and simulation. In *Proceedings of the 37th conference on Winter simulation*, pages 2–15. Winter Simulation Conference, 2005.

[39] Michael J McGowan. Rise of computerized high frequency trading: Use and controversy, the. *Duke L. & Tech. Rev.*, page i, 2010.

[40] Thomas McInish and James Upson. Strategic liquidity supply in a market with fast and slow traders. *Available at SSRN 1924991*, 2012.

[41] Gareth W Morris, David B Thomas, and Wayne Luk. Fpga accelerated low-latency market data feed processing. In *High Performance Interconnects, 2009. HOTI 2009. 17th IEEE Symposium on*, pages 83–89. IEEE, 2009.

[42] Jon Najarian. High-frequency trading is making a joke of the markets. `http://finance.yahoo.com/blogs/the-exchange/high-frequency-trading-making-joke-markets-124446937.html`. Accessed: 2014-01-19.

[43] Travis E Oliphant. *A Guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[44] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[45] Matthew Philips. High-speed trading: My laser is faster than your laser. `http://www.businessweek.com/articles/2012-04-23/high-speed-trading-my-laser-is-faster-than-your-laser`. Accessed: 2014-01-17.

[46] J. Pitman. *Probability*. Springer Texts in Statistics. Springer, 1993.

[47] Jean-Yves Potvin, Patrick Soriano, and Maxime Vallée. Generating trading rules on the stock markets with genetic programming. *Computers & Operations Research*, 31(7):1033–1047, 2004.

[48] Marco Raberto, Silvano Cincotti, Sergio M Focardi, and Michele Marchesi. Agent-based simulation of a financial market. *Physica A: Statistical Mechanics and its Applications*, 299(1):319–327, 2001.

[49] De Rainville, Félix-Antoine Fortin, Marc-André Gardner, Marc Parizeau, Christian Gagné, et al. Deap: A python framework for evolutionary algorithms. In *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference companion*, pages 85–92. ACM, 2012.

[50] Geoffrey Rogow. Colocation: The root of all high-frequency trading evil? `http://blogs.wsj.com/marketbeat/2012/09/20/collocation-the-root-of-all-high-frequency-trading-evil/`. Accessed: 2014-01-19.

[51] Neil J Salkind. *Encyclopedia of research design*, volume 1. Sage Publications, Incorporated, 2010.

[52] D Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178. ACM, 2010.

[53] Jonathon Shlens. A tutorial on principal component analysis. *Systems Neurobiology Laboratory, University of California at San Diego*, 2005.

[54] Steven Skiena. *The Algorithm Design Manual: Text*, volume 1. Springer, 1998.

[55] Leigh Tesfatsion. Agent-based computational economics: Growing economies from the bottom up. *Artificial life*, 8(1):55–82, 2002.

[56] Leigh Tesfatsion. Agent-based computational economics: A constructive approach to economic theory. *Handbook of computational economics*, 2:831–880, 2006.

[57] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.

[58] Chi Wang, Kiyoshi Izumi, Takanobu Mizuta, and Shinobu Yoshimura. Investigating the impact of trading frequencies of market makers: a multi-agent simulation approach. *SICE Journal of Control Measurement, and System Integration*, 4(1):001–005, 2013.

[59] Darrell Whitley. A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85, 1994.

[60] Christopher Williams. The $300m cable that will save traders milliseconds. `http://www.telegraph.co.uk/technology/news/8753784/The-300m-cable-that-will-save-traders-milliseconds.html`. Accessed: 2014-01-17.