An Agent-Based Model of Stock Market Prices

Agent-Based Modelling
University of Amsterdam

February 10, 2019

Abstract

Financial markets are of great importance in the world of trade, but due to the number of involved parties and the complexity of interactions, they can not be easily predicted. While no model can fully grasp such a complex system, an agent-based model (ABM) is a good avenue to acquire some insights about the inner workings of the market. Based on the El Farol bar problem, an ABM was build to emulate the development of a stock price over time. After tweaking the system to show dynamics similar to a real market, the effect of different model components was investigated. It was shown that the market becomes more stable when many agents with a large memory were present, an individual trader could profit in such a market by having a shorter memory, and thereby could adapt to current trends. Agents, displaying a random strategy when making there next offer, have been shown to be be relevant to keep the market from stalling. Paradoxically, however, increasing their number over a certain threshold will cover the effects of the non-random agents, and dampen market trends. Overall, while the ABM does not grasp all the aspects and dynamics of a real stock market, it can still be used to gain some insights into the inner workings of one.

1 Introduction

Since ancient times, people gathered at markets to trade goods. Salesmen displayed their goods in their stands and buyers walked around, negotiating and searching for the bargains. Over time the markets evolved, as with the introduction of the computers the speed and magnitude of the trades could increase. The urge of a physical location disappeared and the type of traded goods altered. Originally there were mainly cattle markets and wool markets, while nowadays the most money is spend and earned on the financial market, where money, precious metals, financial securities and derivatives are traded.

One of the most important things of every trade is the price. While this used to be a matter of negotiations between two parties and the current financial market, this has become an incredible complex system. There are no longer two parties involved, but more than thousands and it is updated in matters of microseconds. Neither rational thinking nor economical theories, as the demand-supply model, are capable to explain these fluctuations.

The financial market involves complex interactions between individuals [1]. To be able to model some of that complexity, agent-based modelling (ABM) could be applied which uses a 'bottom-up' model of the market based on the behaviour of agents [1]. The goal is to simplify the market to a model, which is hindered by the fact that financial markets are inherently unpredictable [2].

Because of this complexity of the system, an ABM has several benefits compared to other models [2]. In particular, ABMs can work with emergent phenomena, are flexible and can give a more natural representation of a system. Moreover, in an ABM each individual involved in the financial market can be defined as an agent [1]. Each agent can then be assigned different characteristics, making the population heterogeneous, similar to a real financial market. A further reason to use an ABM in this situation, is the learning of agents. Agents interact with each other, adapt to new information and update the system accordingly [3].

In this project, the El Farol Bar problem [4] is used as an initial inspiration for the stock market. Studying an artificial stock market as a minority game helps researchers to understand the dynamics of the real stock market. The aim of this research is to study which attributes and strategies lead to the best performing agents.

In the next sections, the modelling of the artificial stock market is discussed. A Sobol sensitivity analysis is executed for sensitivity analysis and the model then is validated against real market data. Experiments are carried out to investigate the effect of market memory and strategy evaluation memory on the profit and number of matches an agent achieves.

2 The Model

In this research, the ABM methodology is based on the ODD protocol [5, 6], and programmed in Python. First of all, an overview on the purpose and the main processes of the model is provided. This includes the entities, attributes, and an analysis of the process and scheduling of the simulation. Secondly, the design concepts are introduced, such as the interaction between the agents. Lastly, details are provided with all the necessary information to allow a re-implementation of the model.

2.1 Overview

Purpose

Financial stock markets are influenced by individual decision making [1, 3]. This can cause the system to become very complex as each individual can make different choices and therefore lead to a considerably large state space [1, 3]. Consequently, this study pursues to analyse the best performing agents. Based on this, traders can apply the outstanding strategies in the trading world.

Entities and attributes

The most important entity in this model is the stock market. It keeps track of the sellers and buyers in the system, as well as the current and past stock prices. Besides that, the agent entity encompasses sellers and buyers. Each agent has a Boolean which determines whether the agent is a seller. It also keeps track of its previous decisions and how each of his possible choices would have performed in that round. Agents have been differentiated between random agents and non-random (now called 'smart') agents. Random agents have made their decisions on the last stock market price, smart agents have used their memory and strategies to make a decision. Moreover, the smart agents have strategies - the third entity in this model. Each strategy has given weights to each slope of the stock market as far back as the agent's memory size.

Furthermore, the attributes (i.e. the state variables and parameters) of each entity are shown in table 1-3 and their functions are presented in table 4-6. For that, pseudo-random number generators are used to drive the model.

Process overview and scheduling

Time has been discretised in this model. Each time step is one round in which sellers offer their stock and buyers have the opportunity to buy it. In each round every agent has been called to make a choice. They have determined which strategy they were going to use in that round, based on the matching of the previous rounds and how their available strategies would have performed in these. Subsequently sellers and buyers have been matched up and the average price for which the stock was sold in that round was appended to the stock market history.

agent		
attribute	type	explanation
ID	Integer	Identification number of the agent
is seller	Boolean	True if agent is a seller, False if agent is a buyer
is random	Boolean	True if agent is utilising random strategies, False if agent uses adaptive strategies
sell/buy prices	Array	Two lists which keeps track of an agents offered sell or buy prices
strategies	Strategies	Strategies of an agent, includes the agent's memory
positivity	Float	Random number [0,1] by which the sell/buy prices are offset to give variation to the model

Table 1: Attributes of entities: all attributes of the agents, strategies and the stock market. For each attribute its type and function is given.

strategies		
attribute	type	explanation
number of strategies	Integer	The number of different strategies this agent has
strategies	List	The list of the different strategies which the agent can choose from
memory	Integer	Number which determines how many past prices the agent will use for predicting the next stock price
strategy evaluation memory	Integer	Number which determine how many past prices will use for predicting which strategy would have been better in previous rounds

Table 2: Attributes of entities: all attributes of the agents, strategies and the stock market. For each attribute its type and function is given.

stock market		
attribute	type	explanation
run time	Integer	The number of rounds in the simulation
number of sellers/buyers	Integer	The number of sellers and buyers in the system
sellers/buyers list	Array	The lists of the sellers and buyers
ratio of random agents	float	The ratio of agents that always choose a random strategy
stock price history	Array	The past and current stock market price
warming-up time	Integer	The number of rounds the warming-up period runs to initialise the model
number of warming-up agents	Integer	The number of agents in the warming-up period
warming-up sellers/buyers	Array	Lists of the warming-up agents

Table 3: Attributes of entities: all attributes of the stock market. For each attribute its type and function is given.

agent	
function	explanation
random choose	Takes the last price on the stock market and offsets it by a random number [0,1]
fixed choose	Evaluates the current trend of the market, based on it's memory. If seller decides the market goes up, it waits selling. If it goes down, it starts the random choose function. For buyer the opposite happens, they wait till the market prices has dropped and start to rise again.
choose strategy	Evaluates the available strategies for the virtual profit they would have made in the past, then picks the one that would have yielded the highest
choose	Based on the stock price history and a list of weights and decides which sell/buy price it will offer for this round
matched	If the agent has found a match, the last price it sold/bought for is added to the agents match price memory and the match count is increased by 1. If not matched the last price stays 0
calculate profit	calculates the difference between the sell/buy price and the actual stock market price
track strategies	Evaluates how close the different strategies of the agent would have performed at predicting the best offer they could have made and stores this value

Table 4: Functions of the entities: all functions of the agents. Each function is given an explanation on how this function is used.

strategies	
function	explanation
create strategies	Takes the number of strategies and memory size and creates a set of strategies
normalise weights	When updating the weights of a strategy, the sum of all weights is set to 1
set weights	Initialises the weights of the memory points in the strategies with random floats
next point	Based on the chosen strategy, calculate the next sell/buy price this agent will offer

Table 5: Functions of the entities: all functions of the strategies. Each function is given an explanation on how this function is used.

stock market	
make sellers/buyers	Create sellers and buyers and add them to their respective lists
update market	Updates the market according to the results of the match up
match	After each agent chooses their strategy, match up the sellers and buyers if the buy price is higher than the sell price
warming-up	Warming-up the system with the warming-up agents during the warming-up time
run simulation	Generate new agents with strategies and run the actual simulation

Table 6: Functions of the entities: all functions of the stock market. Each function is given an explanation on how this function is used.

2.2 Design Concepts

Theoretical Background

The Minority Game

The model is loosely based on the minority game, used in 'The El Farol Problem' [4]. It assumed and modelled inductive reasoning. In complicated problems, humans look for patterns and make hypotheses to which they act. The feedback on the actions taken could then strengthen or weaken these hypotheses, with the aim of finding the best hypotheses for the next situation [4]. In modelling this process, agents are created each with a different set of hypotheses (strategies). When a choice has to be made, the agent acts upon its most plausible hypotheses. As the system changes, the agents will keep track of how well certain hypotheses performed and which ones might be a good choice for the next prediction.

Heterogeneity

The agents in this model are heterogeneous. At first, they can take the form of either a buyer or a seller, which have different interests in their way they determine their buy or sell prices. Secondly, each agent has a different memory. Therefore their strategies, and thus their decisions, can be different.

Individual Decision-Making, learning, sensing and predicting

Each agent has access to a set of strategies, and is tracking how well these strategies would have performed in the gone by rounds. In each round each strategy was evaluated, while evaluations that lie farther back than the strategy evaluation memory reaches, were forgotten. With this updated information, the next round's strategy was chosen based on which strategy would have performed best in the tracked history. While agents adapt to the stock market's past, the adaptation only reaches as far back as the strategy evaluation memory. All agents receive the same information from the market. There is no heterogeneity in the sensing. However, how many past points they can retain (memory) and how far back they evaluate their strategies (strategy evaluation memory) differs between agents. This difference in retained information, in combination with varying strategies and positivity leads to different predictions, even though the information initially given to the agents was identical.

Interaction

The interaction between the agents happens when the agents try to match up - in other words, they try to sell to and buy from each other. The way this is implemented is shown in the

matching algorithm (algorithm 1).

Algorithm 1: The matching algorithm

```
Requires: copy of buyers_list, copy of sellers_list
shuffle lists
for seller in temp_sellers do
    for buyer in temp_buyers do
        Get sell price seller
        Get buy price buyer
        if sell price ≤ buy price then
            Calculate the average of sell and buy price
            Add average to sellers matching price list
            Add average to buyers matching price list
            break
        end
    end
end
```

When every possible match hase been made, the average sell/buy price has been calculated and set as the new stock market price. The agents then have calculated the profit they made in that round. This profit is calculated as the difference between the price they sold/bought for and the stock market price. For example, if a seller sold for a price higher than the stock market price, it gets a positive profit. If a buyer bought for a price higher than the stock market price, it's profit is negative. If an agent did not make a match that round, it made zero profit. For each match made, the agent has updated their match counter.

Stochasticity

The stock market contains random agents. These agents do not choose their new sell/buy price based on strategies, but take the last stock market price and add or subtract a random value to that value. The agents with strategies have a random memory and at initialisation the weights of their strategies is set to a random value. When an agent determines their next offer, their prediction will be offset by their positivity value. This number was randomly generated at the agent's initialisation and will be added to a selling or subtracted from a buying price. In the match-up algorithm (algorithm 1) the agent lists are shuffled before the matching start, thus making sure all the agents have the same probability of finding a match.

Observation

All the observations are shown in section 3 and discussed in section 4.

2.3 Details

Implementation Details

For the complete code see: https://github.com/nathhje/AgentBasedModeling

Initialisation

The warming-up period will contain agents who buy or sell randomly. This will give an initial stock market history. The history of stock market prices then can be used by a new set of agents, which will base their choice on this history of stock market prices.

Input Data

A real stock market was used instead of the warming-up period for some of the experiments [7].

3 Results

3.1 Sensitivity Analysis

To determine what input factors were most relevant in the model and had the most influence on the outcome, a Sobol sensitivity analysis was performed. This was done using the SALib library from Python to create the input samples and analyse the data. Five main input factors were determined and analysed for their influence on three output variables.

The first input factor was the percentage of agents in the model that made random choices. Of this percentage, 90% were the actual random agents, and 10% were steering agents. This factor was varied between 0 and 1. The second factor was the number of strategies each agent had to choose from, which was varied from 2 to 10. The third factor was the memory length of the agents. In other words, the number of weights a strategy was made of. This was varied from 2 to 50. The fourth factor was the strategy evaluation memory, so the number of rounds an agent looked back to evaluate what strategy was the most successful. This factor was varied from 1 to 10. The last input factor was the number of agents in the system. This number was specified to be the number of one type of agent, so buyers or sellers. This factor was varied from 20 to 80, meaning the total number of agents was between 40 and 160.

For every factor 1000 sample points were taken and a total of 12000 input samples were created. The model was run for all 12000 input samples with a warming-up period of 100 and a total runtime of 1000. For each run three output values were saved.

The first output value was the average profit of a smart agent. In figure 1, a bar chart of the outcome of the analysis is shown. All first and second order effects were determined, along with an estimate of the total order effects of each input factor. From the figure it becomes clear that the percentage of random agents is the biggest influence on the average profit. The other first and second order effects are negligible compared to the percentage of random agents. From the total order effects, however, it can be seen that the number of strategies, memory and number of agents may all have some influence as well. Only the strategy evaluation memory seems to have no effect.

The second output value was the average number of matches made by a smart agent. The results from the analysis can be seen in figure 2. Again, the first, second and total order effects are plotted. Here the percentage of random agents appears to be the biggest influence as well, though here higher order effects have the same influence as the percentage by itself, as can be concluded from comparing the "random total" bar with the "random" bar. Mainly the memory and the number of agents seem to have influence on the number of matches as well and the second order effects of "random and memory", "random and agents" and even "memory and agents" reflect this. The number of strategies has a lot less influence and the strategy evaluation memory appears to have none.

The final output value was the stability of the stock market. The measure used for this was the variance in the market, determined by computing the mean stock price and at every turn calculating how far away from this mean the stock price was. The result of the analysis on the stability can be seen in figure 3. From this figure it can be seen that there are no first or second order effects among the five input variables, though it can be concluded from the total order effects that there are higher order effects at work for all input factors except for the strategy evaluation memory. And judging by the error bars of the "strategies total" and "memory total", these effects could be significantly higher than the bar chart indicates. Though the possibility that the market is too volatile and random for the stability measure to make sense, should also be considered. Despite using a random seed, the evolution of the stock market can differ greatly because of random factors, because each set of input factors requires a different amount of random number to be drawn. To accurately analyse the effects that influence the stability of the market, it may therefore be necessary to approach this stability or the analysis of this stability in a different way.

On the whole it can be concluded that the percentage of random agents in the market has the biggest influence on the average gains of the smart agents and the strategy evaluation

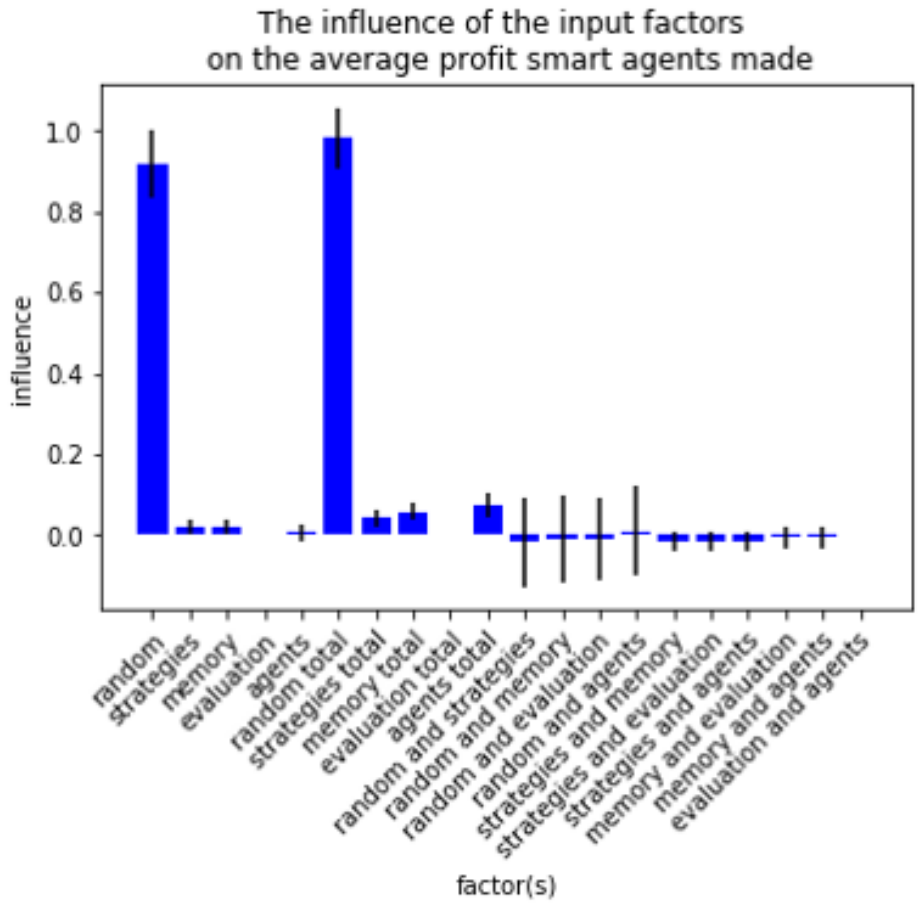

Figure 1: The outcome of a Sobol sensitivity analysis of the model for the average profit of a smart agent. For every input factor the first order effects and the total order effects are depicted. For every combination of two input factors, the second order effects are depicted. The input factors are labelled as random for the percentage of random agents, strategies for the number of strategies each agent had, memory for the memory length each agent based its strategy on, evaluation for the strategy evaluation memory length each agent used to choose what strategy to use each round, and agents for the number of agents in the system.

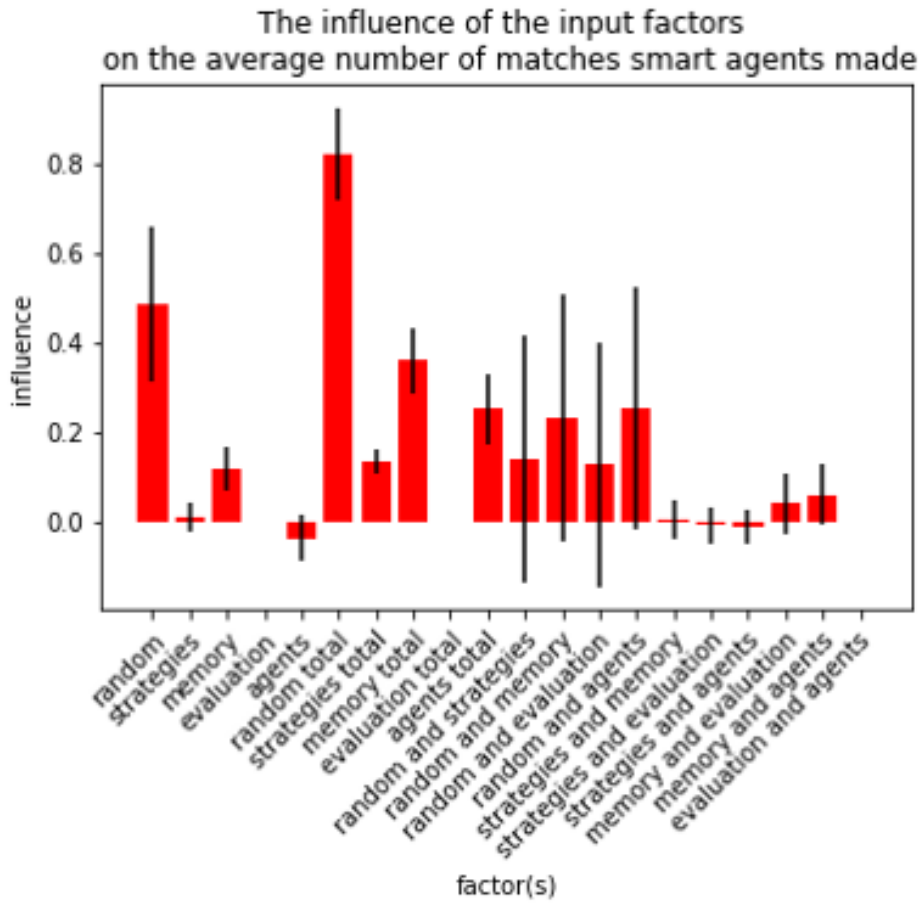

Figure 2: The outcome of a Sobol sensitivity analysis of the model for the average number of matches a smart agent made. For every input factor the first order effects and the total order effects are depicted. For every combination of two input factors, the second order effects are depicted. The input factors are labelled as random for the percentage of random agents, strategies for the number of strategies each agent had, memory for the memory length each agent based its strategy on, evaluation for the strategy evaluation memory length each agent used to choose what strategy to use each round, and agents for the number of agents in the system.

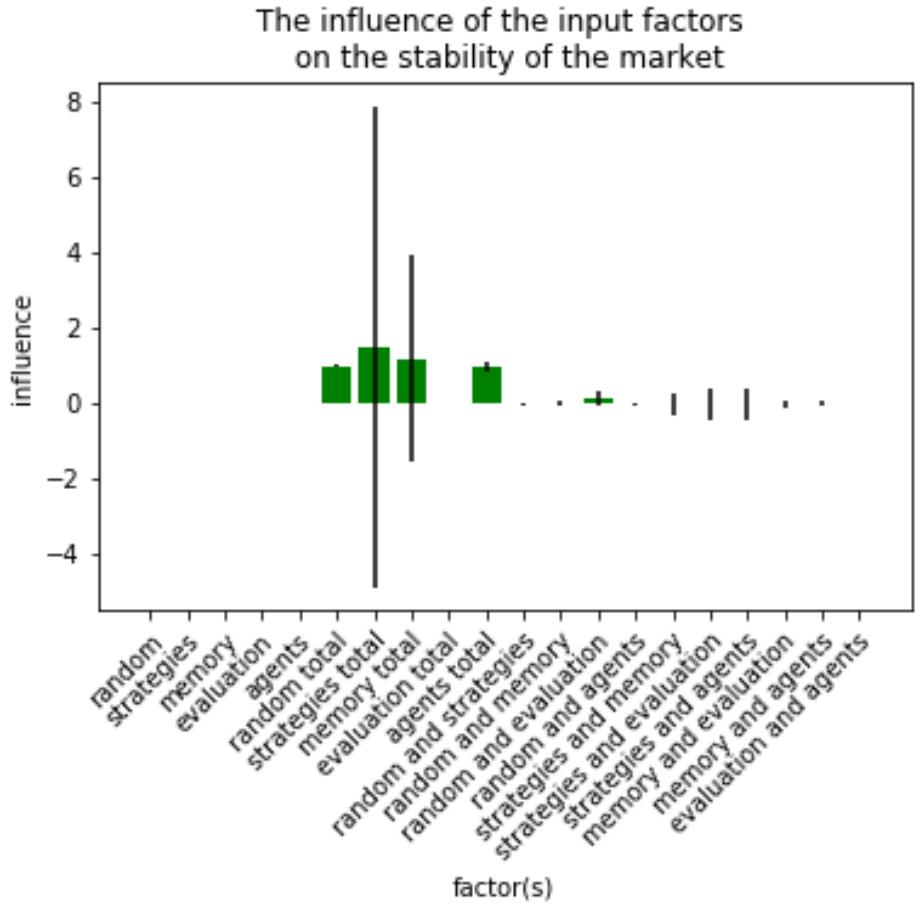

Figure 3: The outcome of a Sobol sensitivity analysis of the model for the stability of the market. For every input factor the first order effects and the total order effects are depicted. For every combination of two input factors, the second order effects are depicted. The input factors are labelled as random for the percentage of random agents, strategies for the number of strategies each agent had, memory for the memory length each agent based its strategy on, evaluation for the strategy evaluation memory length each agent used to choose what strategy to use each round, and agents for the number of agents in the system.

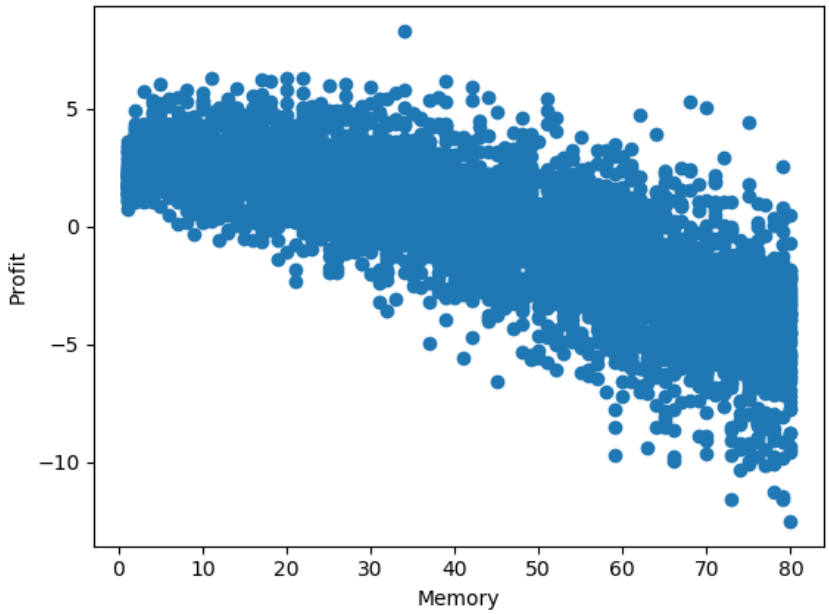

Figure 4: The memory of 10000 agents, plotted against their profit made in 110 time steps. It can be seen from this plot that the agents with a lesser memory range happen to have higher profits on average.

memory has no influence on any of the outputs, making it a poor variable to use in any experiments, which was omitted for this reason.

3.2 Experiments

The main reason why this model was build is to check on how much information a trader on the financial market should base his decision. Experimenting on how the length of this memory influence the market and what other effects it has is another reason to do so. The results of the experimentations will be illustrated and discussed in this section. The relation between the memory and profit of the agents was evaluated first. Therefor 100 simulations were run, each of 100 time steps with 100 agents. Every agent has a memory in the range between 1 and 80, which then were compared with the profit the agent has made in the simulation. This is shown in figure 4. A trend can be seen from this experiments: the longer the memory of the agent is, the lower the profit of the agent is expected to be. Where the agent with a memory of 1 on average makes 2.3 profit, the expected profit of an agent with a memory length of 80 is -1.9. Also the latter has an greater variance.

It's significantly with $\alpha = 0.01$ to say that a memory of 1 gives better profit than a memory

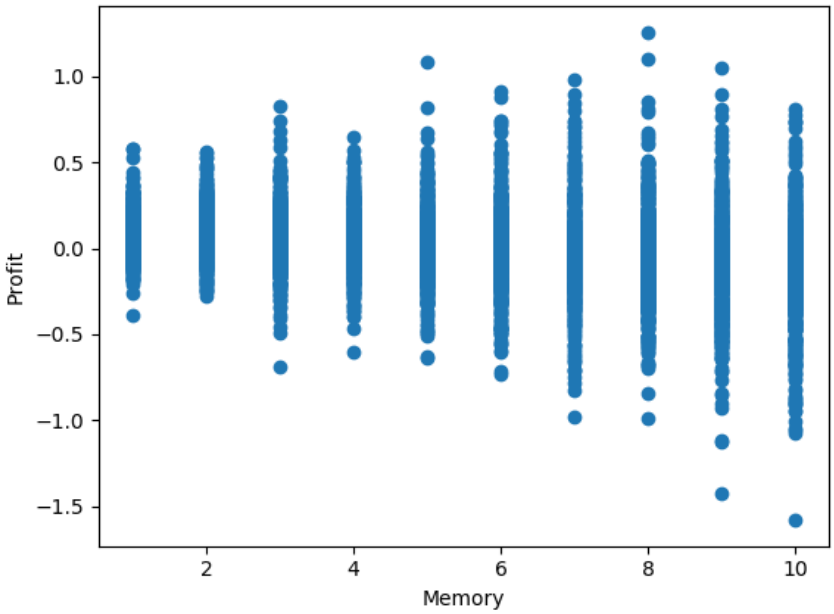

Figure 5: The memory of 10000 agents, plotted against their profit made in 110 time steps. Memory of all agents were restricted to values of ten or less.

of 80. Even though a memory of 1 does not necessary gives the highest profit – there is one outlier with a memory of 32 - the best strategy can be chosen based on only the last change. The possible explanation is that the number of strategies are greater with a higher memory, causing more variance.

The question arose whether these changes when everyone adapts this advantage and only bases their strategies on less history point. In figure 5 the same experiment is shown, but now all smart agents have a memory between 0 and 10. It can be noticed that there is an incredible drop in expected profit for every memory, as the profits are all not significantly different from 0. The variance is slightly growing with the increase of memory. It can concluded that creating a strategy with less historic data is an good decision, unless everyone start doing it, in agreement with the conclusion of the El Farol model.

This lead to the question how the market develops when all smart agents have a similar memory length. To test this, markets were initialised with all agents having one set memory, and the variance of the market was observed. Shorter memory caused the markets to show a higher variance. Starting with memory 20, the variance dropped, and levelled out around memory 40. At this point, increasing all the agents memory further, did not seem to have an effect on market stability anymore.

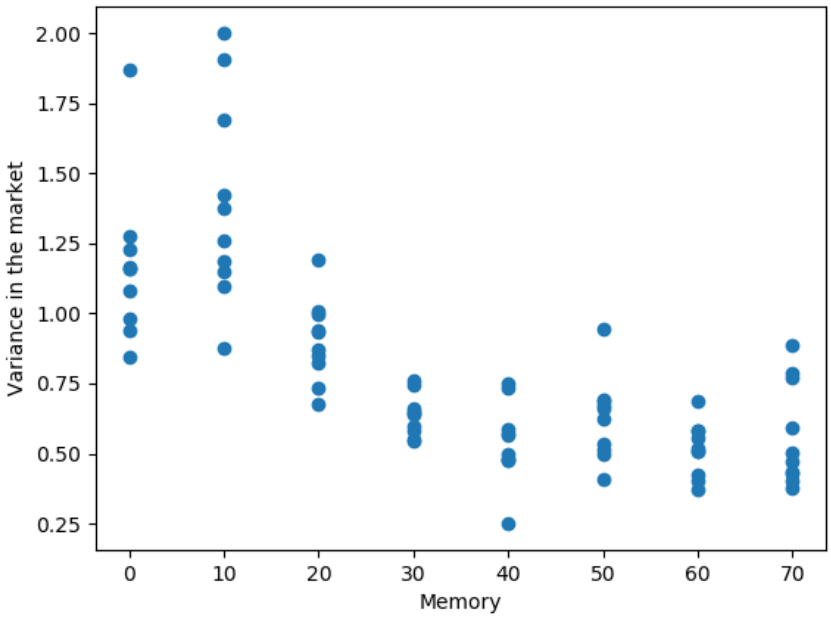

Figure 6: The effect of memory on the variance of the market. Markets were initialised with agents that all shared one memory size and run for ten iterations. This was done for memory sizes ranging from 0 to 70. Mean values of the markets variance were plotted against the memory.

3.3 Verification

The model is created in such a way that its output appears similar to an actual financial market. It exhibits random changes and periodic effects, both caused by the choices of the agents in the model. To verify the model, however, the effect of different input data on the output is investigated.

Since the model is based on pseudo-random effects, each run will lead to varying results. However, certain trends should become apparent, despite these random effects. With certain input data, the model, if properly designed, should exhibit certain market effects. Observing these predicted effects should allow us to verify the model. As a measure of these trends the average growth and the variance of the market for models run on different data have been compared.

To evaluate the influence of the input data on the prices as the model runs, the input and output attributes have been compared. Specifically the slopes between different time points, for both the input and output data, are looked at. Similarly, the variance in the stock prices can be compared for these two data sets.

For a linear, or close to linear, input the model will adopt this behaviour rather clearly (See figure 7). The growth per time step was generated stochastically. The changes of the price of the input are normally distributed with a mean of 0.16 and a variance of 0.0001. It can be seen that the growth in the input is still visible over 500 runs. As the changes of the input are defined, the standard deviation of the growth is near 0.01, and it can be seen that the output remains slightly higher, especially when the number of random agents increases.

The higher standard deviation in the output can be explained by the randomly drawn weights agents use in their strategies. Due to these random weights, there is a greater change of oscillation. Since the prices keep growing, no equilibrium is reached and thus the random effects will persist (See figure 7). The standard deviation grows with the number of random agents, as their behaviour is not based on the history of the stock price.

In contrast, the average growth decreases as the ratio of random agents rises. While the random agents can cause the prices to both grow or shrink, their contribution to a rising price is lower than the effect of the smart agents.

While the market should in theory be symmetrical, it can be observed that the price growth, after the smart agents are introduced, always outpaces the growth in the warming-up period. No exceptions to this have been observed. This could potentially be caused by memory length of the agents, which is shorter than the entire warming-up period and thus giving them the perception of a higher average growth. This will be further discussed in section 4.

With increased randomness in the warming-up period as the result are visible in figure

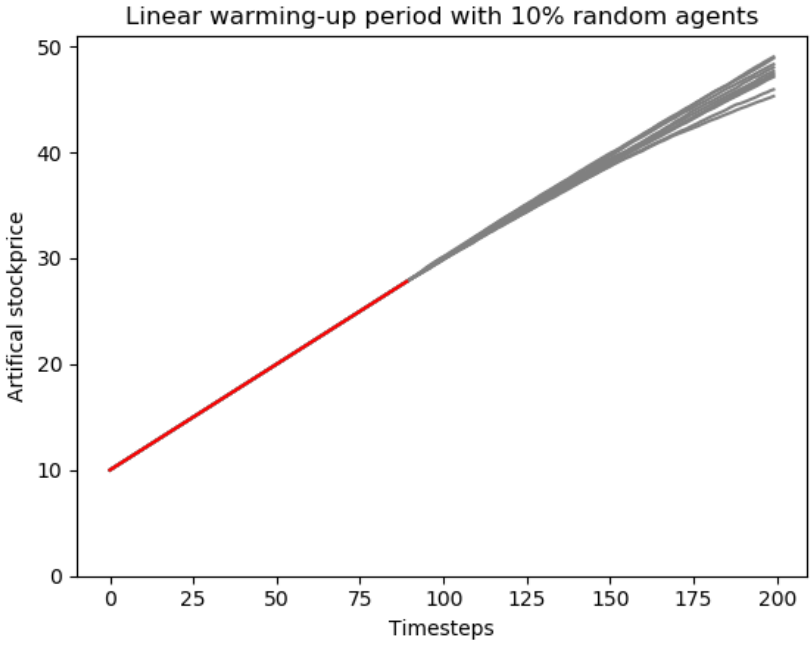

Figure 7: Ten predictions of the model. Every trader starts with a linear stock price history, which has a continuous growth. The model has ten random agents and 90 agents with three strategies. At the beginning, due to the character of the strategies, each strategy will predict the same point, the linear extrapolation of the warming-up period, and thus the market will tend to follow this, only brought off course through the random agents.

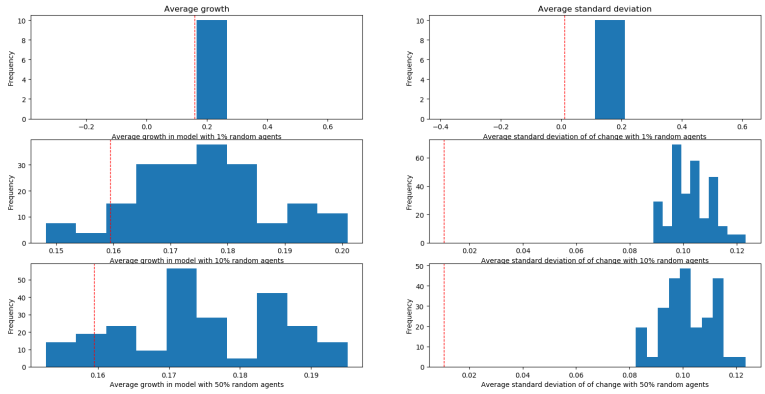

Figure 8: The effect of the linear input and the output of the model can be seen. With a very small percentage of random agents, the model's output has the same average and variance (the input has been marked down with the red dotted line). As the number of agents in the model increases, the variance of the average growth increases as well and moves slowly towards zero. The standard deviation of the growth also drops as the number of random agents increases.

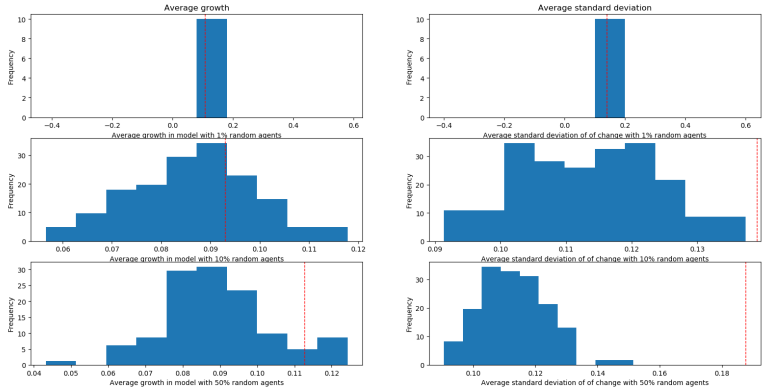

Figure 9: The same analysis as before, but now with the random agents calling the shots the first 90 time steps. This causes the warming-up period to have an average growth of near zero. However it can be seen that with little random agents the output has the same properties as the warming-up period. With an increased amount of random agents, it can be seen that both the average as the standard deviation drops. All models have been ran with 100 agents over 100 time steps.

9, the observed effects are less clearly passed on as the agents process the history for their predictions. A highly volatile market will cause different strategies to produce strongly varying predictions. The changes can be described as pulled from a normal distribution. However, the standard derivation of the output clusters 0.2, and is thereby significantly smaller than the standard deviation of 0.8 in the warming-up period. This number is stable even when the percentage of random agents is increased.

It also showed that the average growth is negative in all cases. Again, this is effected by the limited memory of the agents, and the fact that they can only perceive the most recent entries. As the average growth of the last data entries is negative, the growth continues to develop with a negative trend.

Lastly, for the average growth, no clear effect of an increased number of random agents is observed. It is possible though, that more random agents tend to bring the average growth of the market closer to 0 9. This, however, contradicts the assumptions made about the random agents in 2.2, that random agents would introduce variation into the market and agents with strategies would stabilise it. Here however, it becomes apparent, that with an increase in random agents, changes in the market decrease.

3.4 Validation

This model was never meant to predict the financial market, merely to recreate the interactions of buyers and sellers on a market and their common effect on the market prices. So the

intention never was to recreate real market data and use this model to predict the course of the market. Thus this is not the proper way to validate the model. However, as said in section 3.3, preferably the market would be as realistic as possible. So real financial market that is fed to the model and this will make it possible to compare the real course with the modelled course, then similarities should be expected.

In order to do so, in the data are the opening prices of NASDAQ in 2016 downloaded[7]. The American stock market is a convenient market as it has been reasonably stable in 2016. The opening values will be added as the first data points of the model, after that the agents will consider these values and start trading. See figure 10.

In this explicit case it is visible that there are still many differences between the model and the real data from the original system. The volatility of the real market data isn't properly captured by the model. This might be, however, solvable by adjusting the parameters of the model. It is not possible to state that this validates the model, but as said in the beginning of this paragraph, the model was never meant to predict the financial market, just to see how it responses to different trade strategies.

4 Discussion

4.1 Sensitivity analysis

A Sobol sensitivity analysis was performed to determine which factors have the highest influence on the models outputs. As output factors, the profit of a smart agent and their number of matches as well as the stability of the whole market were analysed.

The factor with the biggest influence on an agents profit, was the number of random agents occupying the same market. This is not unexpected since the represents a zero-sum-game (one man's profit is another man's loss). Random agents showed a stable average loss, meaning that a higher number of random agents, would yield more profit for the smart agents. Additionally, with fewer smart agents present, a bigger part of that profit would reach each one on average. The number of strategies, memory length and number of agents also showed some effects, but these were small compared to the effect of the random agents. Since these variables can determine how an agent performs in a market, i.e. by making better predictions, or by competing with fewer agents, tit is not surprising, that they show some relevance. Memory evaluation memory, however, appeared to have no effect at all. For the number of matches, a similar pattern emerges, with the number of random agents having the biggest influence, for similar reasons as mentioned above, and memory, number of strategies and number of agents, having some, but less influence. These however might arise from the second order effects with memory. Market stability seems to be influence most heavily by the random

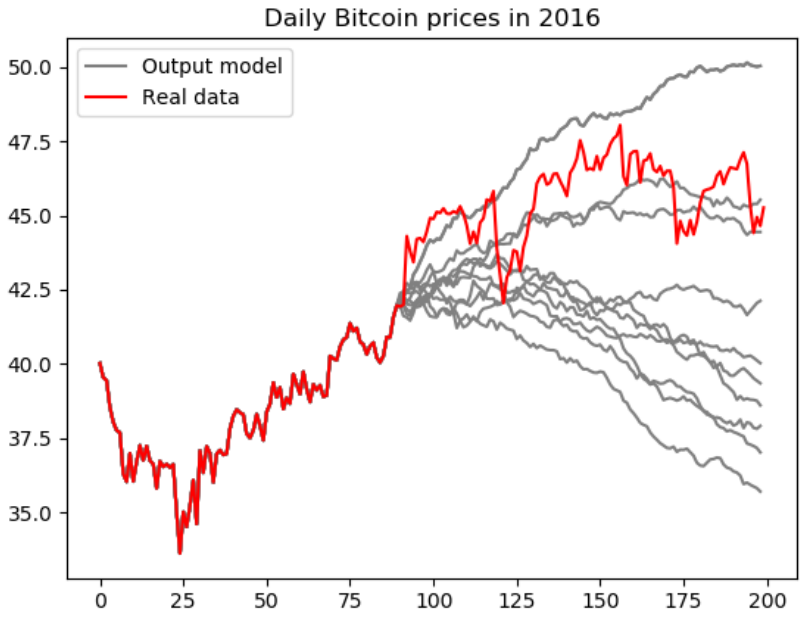

Figure 10: The line graphs of the actual price of the NASDAQ in 2016 (red) and ten different outputs created by the model. It shows that the volatility of the model isn't close to the real values. However, the real market data is in between the price ranges predicted by the model.

agents, strategies and memory. Due to the height of the error bars however the latter two are probably not the most reliable influences. Again, the number of random agents seems to have the biggest influence. Overall it can be concluded, that the number of random agents is the biggest driving force in the market. The number of strategies, memory size and total number of agents seem to have some effect. And the strategy evaluation memory appears to have almost no effect on the model.

4.2 Experiments

Experiments have been done on the ABM in order to answer the main hypothesis: how the length of the memory of the agents influences the profit and the course of the market. To do so, certain starting information have been created, in one case completely random. When answering the question how the starting data influences the output of the model, not one one set of random input data should be given and evaluated. In this specific case is already said that the ending of the warming-up period can be crucial to the rest of the models output. This as not been properly tested due to lack of time.

The variance is slightly growing with the increase of memory, so lower memory does seem to be better. However, as stated in the El Farol model and shown in the experiments, if everyone uses the same strategy, the strategy would not work optimally.

For a market with agents of all equal memory, the profit did not change for different values. The market stability however was heavily influenced by the memory. Longer memory led to a more stable market, since all agent remembered a longer past and can therefore to more similar conclusions.

5 Conclusion

The financial market is unpredictable and complex. Every trader keeps making decisions, interact with other traders and all those decisions continuously change the market prices. The decisions are mostly found on one or more strategies: every trader evaluates the current market and then bases their actions on their observations and own beliefs. How far should a trader look back before making a decision? Should the trader check the price history of last week, last month, last decade? And how does this changes the stability of the financial market? The hypotheses is that more information can causes better predictions, but there will be a limit to the amount of processed intel as to the use in the prediction.

To answer this question, an agent based model is build, since there are many different agent (traders, both buyers and sellers) which are a heterogeneous group with a complex interaction with each other and the environment (the financial market). Here every agent can

have a different memory (amount of data in history reviewed) and strategies (an individual set of weights, relating to the personal opinion of importance of the previous prices in the market). To create a financial market with properties similar to the observed system, three different agent possibilities are introduces: a random choice, a fixed choice based on trends in the market and a choice based the set of strategies. The end result is a model with many different parameters.

To test the model, a Sobol analysis is done. The most important conclusion of the analysis is that the way the best strategy is picked is not of real importance. To verify the model, it is checked how well the model can retain certain properties and how well the occurring randomness of the model can be explained. Validation is also done on the model, however the model was never meant to analyse real world data.

Eventually, less memory is profitable for the traders. However this does not seems to be the case if every agent uses this technique. Also when every agent bases their decisions on a limited amount of data, the financial market grows rapidly unstable.

References

[1] B. Lebaron, "Building the santa fe artificial stock market." *Brandeis University*, June, 2002. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.130.6439

[2] J. Wiesinger, D. Sornette, and J. Satinover, "Reverse Engineering Financial Markets with Majority and Minority Games Using Genetic Algorithms," *Computational Economics*, vol. 41, no. 4, pp. 475–492, 2013.

[3] B. LeBaron, W. Arthur, and R. Palmer, "Time series properties of an artificial stock market," *Journal of Economic Dynamics and Control*, vol. 23, no. 9-10, pp. 1487–1516, 1999. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0165188998000815

[4] W. B. Arthur, "Inductive Reasoning and Bounded Rationality (The El Farol Problem)," *Amer. Econ. Review (Papers and Proceedings)*, vol. 84, p. 406, 1994.

[5] B. Müller, F. Bohn, G. Dreßler, J. Groeneveld, C. Klassert, R. Martin, M. Schlüter, J. Schulze, H. Weise, and N. Schwarz, "Describing human decisions in agent-based models - ODD+D, an extension of the ODD protocol," *Environmental Modelling and Software*, vol. 48, pp. 37–48, 2013.

[6] V. Grimm, U. Berger, D. L. Deangelis, J. G. Polhill, J. Giske, and S. F. Railsback, "The ODD protocol : A review and first update," *Ecological Modelling*, vol. 221, no. 23, pp. 2760–2768, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.ecolmodel.2010.08.019

[7] B. Marjanovic, "Huge stock market dataset - historical daily prices and volumes of all u.s. stocks and etfs," 2017. [Online]. Available: https://www.kaggle.com/borismarjanovic/price-volume-data-for-all-us-stocks-etfs/home