CHALMERS

Finding Predictive Patterns in Historical Stock Data Using Self-Organizing Maps and Particle Swarm Optimization

HANS SALOMONSSON

The Division of Physical Resource Theory
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2012
Master of Science thesis number 2012:3

Finding Predictive Patterns in Historical Stock Data Using Self-Organizing Maps and Particle Swarm Optimization

Salomonsson, H.

The Division of Physical Resource Theory
CHALMERS UNIVERSITY OF TECHNOLOGY
412 96 Gothenburg

REPORT NO. 2012:3

Finding Predictive Patterns in Historical Stock Data Using Self-Organizing Maps and Particle Swarm Optimization

HANS SALOMONSSON

The Division of Physical Resource Theory
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2012

Acknowledgments

First I would like to thank my supervisor and examiner Professor Kristian Lindgren for accepting this project as a thesis and giving me the support and space to carry it out.

A warm thanks to my girlfriend, Karin Lindstedt, for great support and for making my life easier during the writing of this report.

I would also like to thank my family for the support during this thesis as well as my entire time at the university.

Sammanfattning

Syftet med detta examensarbete är att hitta en ny metodik för att hitta mönster i candlestick-grafer som kan förutsäga en akties framtida kursrörelse. Metoden ska kunna hitta sådana mönster utan någon fördefiniering av hur dessa ser ut. En algoritm som kombinerar particle swarm optimization och self-organizing maps har implementerats och utvärderats för att lösa detta problem. Icke-transformerade dagliga öppnings, högsta, lägsta och stängningskurser har använts som indata. Denna algoritm hittade mönster som med statistisk signifikans överträffade slumpvis handel. Vidare undersöktes flera intressanta egenskaper hos de funna mönstren. Däribland längden på mönstren, prognostiseringshorisont samt kvantitativt hur lik indata ska vara ett mönster.

Abstract

The purpose of this thesis is to find a new methodology for finding predictive patterns in candlestick charts without any predefining of how these might look. An algorithm combining particle swarm optimization and self-organizing map has been implemented and evaluated. Non-transformed daily open, high, low and close data has been used as input. The algorithm found predictive patterns that statistically significant outperformed random trading. Moreover, interesting properties such as the optimal length of the pattern, target length and similarity of input to found pattern are discussed.

Contents

Nomenclature

d	Similarity threshold for an input pattern to be close enough a pattern.
EMH	Efficient market hypothesis. A theory that claims that asset prices reflects all available information.
l_{grid}	Length of the side of the square map used in the SOM.
l_{target}	How many days into the future the pattern makes its prediction.
l_{window}	Length of a pattern in days.
OHLC	Opening, high, low and closing price of a stock.
PSO	Particle swarm optimization. A stochastic optimization algorithm.
q_{used}	Fraction of neurons/patterns in the SOM used for trading.
r_{target}	Return from the last day in the pattern's closing price and closing price of the day specified be the target length.
ρ	Fraction of trades that were positive.
RWH	Random walk hypothesis. A theory that claims that stock prices move like a random walk.
SOM	Self-organizing map. Neural network technique to represent an high dimensional input space in lower dimensions.
$\mathbf{x}$	Position vector with variables to be optimized using PSO.
ξ	Input pattern to the SOM.

1 Introduction

This section covers some background in finance from an academic perspective. Moreover, the purpose of this thesis and how it is linked to previous studies are explained.

1.1 Background

The efficient market hypothesis, EMH, were for a long time taken to be true [1]. Since then the hypothesis has been refined into three different forms; strong, semi-strong and weak [2]. Today a large portion of the trades world-wide are automated by trading algorithms, but this contradicts even the weak form of the EMH. The weak form of the EMH claims that the price of an asset reflects all publicly available information. Therefore a trading system that uses historical data that is publicly available can not give any prediction on future prices. Another theory proposed by Bachelier [3] and later developed by Cootner [4] and Fama [5] claims that the stock price moves like a random walk. If this is true no prediction on the direction of the price can be made. But still there are many traders looking for patterns in historical stock prices and using these for trading decisions. In other words it seems that there are quite some disagreement between academia and practitioners.

There exist many findings that do contradict the EMH [6][7][8] and also the random walk hypothesis, RWH [9][10]. In addition to this there exists articles on trading systems using neural networks that are more profitable than a buy-and-hold strategy [11][12].

How and why markets are not efficient has received a lot of attention in the field of behavioral finance. They claim that the financial markets are informationally inefficient. Further, they claim that this is partly due to cognitive biases among investors. People are not fully rational, e.g. investors are overconfident, stereotype certain stocks as winners, do not change their perception enough on news, etc. Behavioral finance is not only interesting for explaining how the financial markets work, but also for practitioners to gain insights in how rational deviations can be used to create profitable trading systems [13]. The so-called *chartists* are an example of this.

Chartists, also known as technical analysts, uses recurring patterns in stock charts to make trading decisions. These patterns are claimed to be created due to monetary, political and psychological forces [14]. Due to the subjective nature of these patterns it is difficult to formalize these to be backtested on historical data to examine their effectiveness. However, by using a template grid predefined with a bull flag Leigh et al. [15][16] and Wang et al. [17] found this pattern, frequently used by chartist, to be

systematic significant better than random trading. An illustration of the bull flag can be seen in Figure 1.

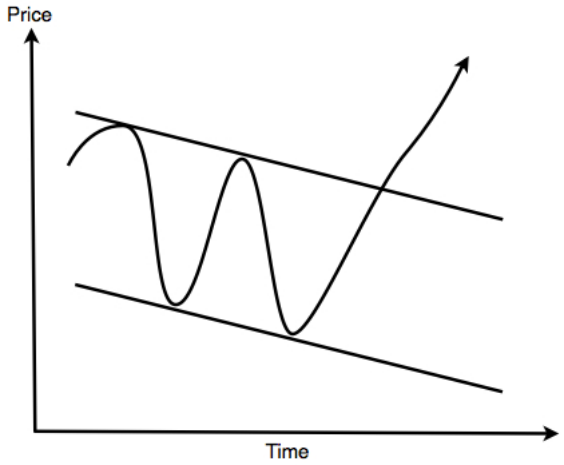

Figure 1: *An illustration of a bull flag. The formation is used by chartists to make trading decisions. It consists of two parallel bearish trend lines. A buy signal is created when the stock price breaks through the upper trend line.*

Leigh et al. and Wang et al. showed that the technical analysis pattern called bull flag was profitable, but what about all the other patterns used by practitioners? One approach to answer this question would be to design template grids for all known patterns and test them. This would imply a lot of work and many difficulties, e.g. different chartists would disagree of the definition of the patterns, how many days the pattern is made up of, the angle of e.g. the flag, etc. This makes the ambiguity of this research high. This thesis suggests an alternative approach to the template grid. The approach, or methodology, is looking for patterns without the need to first describe them, define their formation length or target horizon. In this way not only known patterns will be found, but possibly also new patterns. Instead of only using the close price, which seems to standard in research, the open, high and low prices are also included.

1.2 Purpose

The purpose of this thesis is to find a methodology that can be used to find patterns that carries predictive information in historical stock price data. When used the patterns should perform better than random trading during the same period. This would contradict all forms of the EMH because publicly available historical data are used to make successful forecasts. Moreover,

it would also contradict the RWH since these patterns carry predictive information in the part considered to be noise by the RWH. Moreover, if such patterns are found explore variables that could be useful for practitioners. Examples of such variables are how similar a formation has to be to a pattern to be classified as the pattern and in what time scales the most predictive information resides. In addition to this the possibility of finding profitable patterns during the financial crises will also been examined.

The data used has not been transformed in any way, e.g. by computing moving averages, Fourier transform or any other transformation. By doing transformations noise could potentially be filtered out and better signals to use for a trading system might be the result. The reason for not doing any transformations is to restrict the patterns so that they potentially could be spotted by a chartist by pure visual inspection in a candlestick chart. The patterns found with this methodology can potentially be used both by chartist and for automated trading systems.

1.3 Research Questions

1. Can predictive patterns be found using SOM and PSO on non-transformed daily open, high, low and close data?

2. Are the mean of the returns from trading with the found patterns statistically signifiant different from that of random trading during the same period and how big is this difference?

3. Which time frames include the most predictive information?

4. Do nervous market conditions decrease the predictability of patterns?

1.4 Delimitations

The methodology suggested in this thesis is not a trading system. However, the patterns found could be incorporated into a trading system, e.g. by using an ensemble method. Ensemble methods are a group of algorithms that combines weak predictors and combining them to one strong predictor.

No consideration of the cost of trading has been taken into account. By including this, shorter formation would be punished and longer formations favored. Including this cost is important when designing a trading system, but not for the purpose of this thesis.

The focus of this thesis is to investigate if the suggested methodology can be used to find patterns that outperform random trading. If patterns are found the next thing to study is if these patterns resemble patterns used by

chartists. By doing so existing patterns could be verified as predictive and unseen patterns could possibly be incorporated into practitioners toolkit. However, this did not find its place into the time frame of this thesis.

The resolution of the data was daily. Intraday series was too expensive to be used. However, the same analysis could be applied to intraday data.

Computers with four giga bytes of random access memory were used to run the simulations. Running simulations with more data than used in this thesis a computer with more random access memory would be necessary. Alternatively rewrite the algorithm to store temporal data on the hard drive.

1.5 Outline of this Thesis

The rest of this thesis is structured as follows. In Section 2, the necessary theory lying behind our experiments is described. In Section 3, the algorithm that was used for finding predictive patters are presented. Section 4 describes the results generated from this algorithm. In Section 5 these results are discussed. In Section 6 the conclusions made from this work is pointed out. A graph of the Nasdaq Composite Index from 2002 to 2012 can be found in Appendix A.

2 Theory

This section explain the theory needed to understand the rest of this thesis. First the properties of the stock market are discussed. Then the reader will get a refresher in statistics and finally the machine learning algorithms used in this thesis are explained.

2.1 Properties of the Stock Market

How the dynamic of the stock markets work is something that is not fully understood. Below are some different views on the topic.

2.1.1 The Efficient Market Hypothesis

When the term ”efficient market” was first introduced in economics it was referring to markets which ”adjusts rapidly to new information” [18]. A more modern definition is that asset prices in an efficient market ”fully reflect all available information” [1]. This implies that the market is rational and there exists no way to forecast future price movements. There are three different versions of the efficient market hypothesis; the weak, the semi-strong and the strong. The weak form claims that asset prices reflects past information, the semi-strong additionally claims that the asset price is instantly adjusted to new information and the strong version additionally claims that insider information is reflected into the price as well [2]. Many studies confirms the efficient market hypothesis, e.g. a study made by Cowles showed that even market professionals in average did not perform better than the market [19]. A similar study was made by Jensen [20]. He analyzed 114 funds over the period 1955-64 and found that any advantage the funds might have was consumed by the fees and expenses from owning such a fund.

2.1.2 Evidence Contradicting the Efficient Market Hypothesis

Fama wrote ”in short, the evidence in support of the efficient markets model is extensive, and contradictory evidence is sparse” [1]. However, there are several studies that do contradict the EMH. One such study was conducted by Basu. By using the price/earnings ratio on 1400 stocks he found that the stocks with a low ratio outperformed those with a high ratio with 7% per year [6]. Another study made by Ball and Brown showed that there seems to be a drift in the direction of a earnings surprise [7]. Ritter found that there is tendency for new issues to perform negative on a long run. He found this by looking on 1526 initial public offerings over the period 1975 to 1984

[8]. Shiller found that the stock market prices were fluctuating too much to be justified by the change in expected future dividends. He found that the magnitude of the fluctuations were as much as 5 to 13 times as high as they ought to be [21].

2.1.3 The Stock Market as a Random Walk

In Pearson's paper "The problem of the random walk" he discusses what would be an optimal search strategy to find a drunk. If the drunk walked in a unpredictable and random fashion the place he most likely would be is where he started [22]. This analogy has been used to financial time series when the successive returns are serially independent [5]. In the 1950s researchers had for the first time access to computers that could perform analysis on lengthy time series. Kendall found that if the long term trend was removed the residuals seemed to fluctuate randomly and with almost no autocorrelation [23]. Thus was the foundation for the stock market as a random walk established.

To prove that the random walk hypothesis is wrong one needs to find predictability in the part that is only considered to be noise. Several studies have found positive autocorrelation in weekly and monthly returns while the autocorrelation on several years is negative [9]. A study that has had a great impact on practitioners was written by DeBondt and Thaler. They found that the stocks that had underperformed the most during a period of 3-5 years average the highest performance during the following 3-5 year period [10]. This was evidence that the market overreacted by undervalue stocks that were considered to be bad and vice versa for good stocks.

2.1.4 Behavioral Finance

Behavioral finance is the study of how the psychology of the practitioners effect their behavior and subsequent the financial markets. It tries to explain why and how markets might be inefficient. In the EMH the market is supposed to be rational and unbiased forecasts of the market can be performed. Since the forecasts are unbiased every investor would agree on the price and the asset price would change accordingly. Behavioral finance, however, assumes that markets are informationally inefficient and therefore investors potentially would not agree on the current asset price. Partly this is due to people's cognitive biases, see below [13].

Heuristics are rules of thumb and is often used in the decision process. This can lead to biased decisions. An example of such a rule is the $1/N$

rule, which claims that if you are faced with N alternatives to allocate your resources you tend to split them in equal sizes [24].

Overconfidence means that people overestimate their abilities. This can be expressed by too little diversification in portfolios [13]. It can also be manifested in trading behavior. People that are more confident tend to trade more. A study by Barber and Odean showed that the more people traded the worse they performed on average [25].

Mental accounting is the act of making fictional accounts that really ought to be combined. E.g. having one account for entertainment and one for restaurant visits might lead to suboptimal decisions.

Representativeness is a common decision making bias that makes people sort things into stereotypes and base their decision on that. E.g. a well performing stock might be thought of as a winner and is expected to continue to be a winner.

Conservatism, also known as anchoring, is the inability to change to new conditions. Even though new information is provided, people anchor on their old belief and do not adjust accordingly to the new information. The consequence of this is that good news is often followed by more good news, because people have not adjusted their beliefs enough on the last piece of news.

Disposition effect describes the fact that traders are inclined not to realizing their losses. The losses are often hold until it reaches the buying price [13].

2.1.5 Trading with Patterns

Trading with patterns is often referred to as technical analysis, or chartism. By finding recurrent patterns in graphs that have predictive value and trading with these the technical analyst hopes to make profits [26]. The idea is that stock prices "move in trends which are determined by the changing attitudes of investors toward a variety of economic, monetary, political and psychological forces" [14].

Using technical analysis on US dollar exchange rate has shown to create significant excess returns, which as been shown by Levich and Thomas [27], Osler and Chang [28] and Neely, Weller and Dittmar [29]. However, there are studies that show that technical analysis does not perform better than a simple buy and hold strategy, e.g. a study made by Allen and Karjalainen [30].

2.2 Statistics

Statistics deal with understanding data. Below is an introduction to methods used in this thisis.

2.2.1 Probability Density Function

A probability density function can be used to describe the probability distribution of a random variable. If an interval is more likely to contain the value of the random variable its probability density function will be higher [31]. An example of a probability density function of a normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$ can be seen in Figure 2. The density has its highest point at zero, which indicates that for this random variable values around zero are the most likely. If the underlying probability density

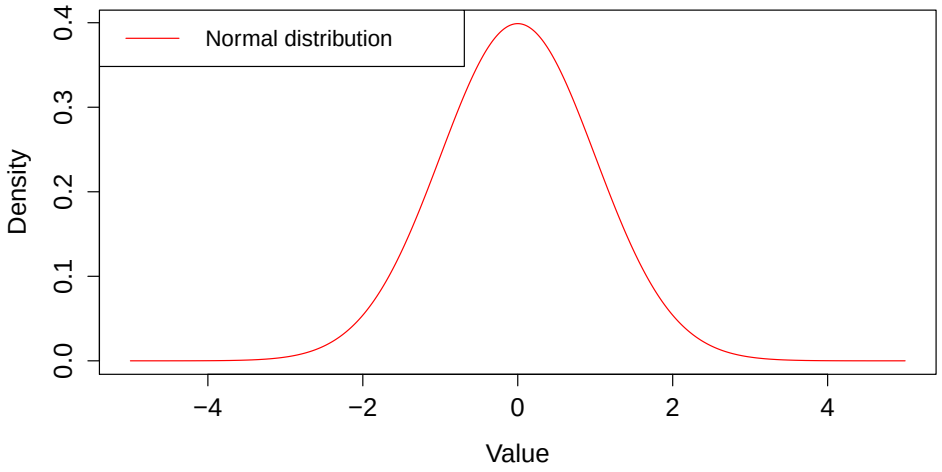

Figure 2: *The density function from a normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$.*

function is not known it can be estimated using a kernel density estimation [32][33]. Such an estimation can be seen in Figure 3.

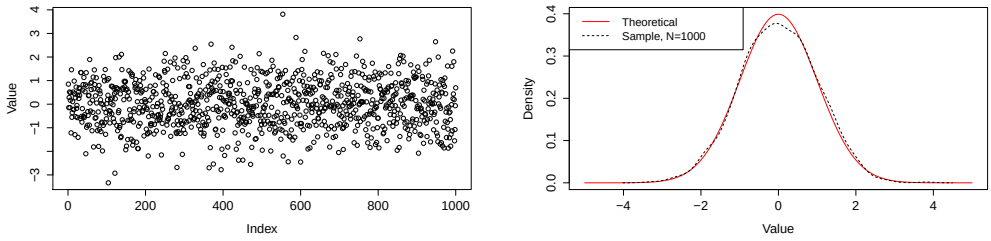

Figure 3: *1000 data points randomly generated from a $N(0,1)$ distribution. The right figure shows the sample's estimated probability density function compared to a theoretical $N(0,1)$ distribution.*

2.2.2 Quantile Quantile Plot

A quantile quantile plot, Q-Q plot, is a statistical visualization method that plots the quantiles of one distribution against the quantiles of another distribution [34]. If the two samples have the same distribution they will approximately follow a straight line, which can be seen in the Q-Q plot in Figure 4.

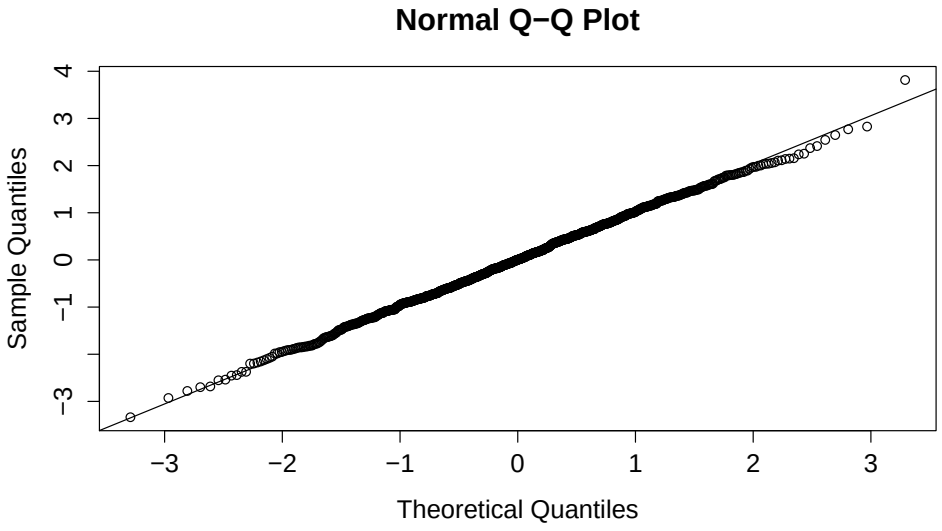

Figure 4: *A Q-Q plot comparing the theoretical quantiles with the quantiles of random generated normal data with mean $\mu = 0$ and standard deviation $\sigma = 1$.*

2.2.3 Test of Hypothesis

A statistical hypothesis is a claim about either one or several populations or the distribution of these. The null hypothesis, often denoted H_0, is the initial stand point and is assumed to be true. The alternative hypothesis, H_a, is the complement to H_0 and therefore contradicts the null hypothesis. The hypothesis test will test wether there exists statistically significant evidence that the H_0 should be rejected and the alternative hypothesis should be accepted instead [35].

The two sample t-test, also known as Welch's t-test [36], can be used for comparing e.g. the means of two populations if the populations are normally distributed. If the sample size is large the central limit theorem says that the sample mean will follow a normal distribution. However, if the sample size is small the sample mean will follow Student's t-distribution. The t-distribution is a symmetric and bell shaped continuous distribution. Let μ_i be the real mean, n_i number of observations, $\bar{x}_i$ sample mean and s_i^2 the sample variance of the i^{th} population. The t statistic in Welch's t-test is then calculated as in Equation (1). This t value can be looked up in a t distribution table to get the probability that the null hypothesis is true [35]. This value is often referred to as the p-value.

$$t = \frac{\bar{x}_1 - \bar{x}_2 - (\mu_1 - \mu_2)}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \tag{1}$$

H. B. Mann and D. R. Whitney introduced a similar test to see whether one random variable is stochastically larger than another. This test does not require a normal distribution of the random variable as the Welch's t-test does. This test is called the Mann-Whitney-test after the authors [37].

2.3 Machine Learning

In machine learning a machine learns behaviors by exposing it to data. Below are two such algorithms explained. Moreover, the problem of generalization is discussed.

2.3.1 Particle Swarm Optimization

Many species have a tendency to form swarms. Swarming offers advantages to individuals like less chance of being targeted from a predator, easier to defend against predators, food gathering and more efficient reproduction. The reason for swarming to inspire an optimization algorithm is due to the

fact that many eyes have a greater chance of discovering food than just one pair [38].

Particle swarm optimization is an optimization algorithm that is inspired by swarms. Let x_{ij} and v_{ij} be the position and the velocity of particle i in the dimension j out of n dimensions, $\mathbf{x}_i^{pb}$ is the best position for particle i and $\mathbf{x}^{sb}$ is the best position of all particles, r and q are random numbers in $[0, 1]$ and $f(\mathbf{x})$ is the objective function to be maximized. The best position refers to the $\mathbf{x}$ that yields the highest, or lowest if minimizing, value of $f(\mathbf{x})$. The PSO algorithm then looks as follows

1. Initialize random positions and velocities of the particles p_i:
   - $x_{ij} = x_{j,min} + r(x_{j,max} - x_{j,min}), \qquad i = 1, ..., N; j = 1, ..., n$
   - $v_{ij} = \frac{1}{\Delta t}(-\frac{x_{j,max} - x_{j,min}}{2} + q(x_{j,max} - x_{j,min})), \qquad i = 1, ..., N, j = 1, ..., n$
2. Evaluate each particle in the swarm, i.e. compute $f(\mathbf{x}_i)$ for $i = 1, ..., N$.
3. Update the best position of each particle, and the global best position. Thus, for all particles p_i, $i = 1, ..., N$:
   - if $f(\mathbf{x}_i) > f(\mathbf{x}_i^{pb})$ then $\mathbf{x}_i^{pb} \leftarrow \mathbf{x}_i$.
   - if $f(\mathbf{x}_i) > f(\mathbf{x}^{sb})$ then $\mathbf{x}^{sb} \leftarrow \mathbf{x}_i$.
4. Update particle velocities and positions:
   - $v_{ij} \leftarrow v_{ij} + c_1 q(\frac{x_{ij}^{pb} - x_{ij}}{\Delta t}) + c_2 r(\frac{x_j^{sb} - x_{ij}}{\Delta t}), \qquad i = 1, ..., N, j = 1, ..., n$
   - Restrict velocities such that $|v_{ij}| < v_{j,max} = \frac{x_{j,max} - x_{j,min}}{\Delta t}, j = 1, ..., n$.
   - $x_{ij} \leftarrow x_{ij} + v_{ij}\Delta t, \qquad i = 1, ..., N, j = 1, ..., n.$
5. Return to step 2, unless the termination criterion has been reached [38].

The constants c_1 and c_2 are positive constants often set to two, N is the number of particles and j the number of variables that is to be optimized. Δt is often set to 1 for simplicity [38]. $f(\mathbf{x}_i^{pb})$ and $f(\mathbf{x}^{sb})$ are initialized as very low numbers if the objective function should be maximized. To decrease the risk of getting stuck at a local optima a craziness operator can be used. This is applied with a given probability and changes the velocity of particle i in direction j according to

$$v_{ij} = -v_{j,max} + 2rv_{j,max}, \qquad j = 1, ..., n, \tag{2}$$

where r is a random number in $[0, 1]$ [38].

2.3.2 Self-Organizing Maps

Self-organizing maps is an unsupervised neural network learning technique. Its self-organizing process is based on the notion of competition. All neurons compete and only one neuron is the winner, which is simply called the winning neuron. The synaptic weights are then updated in the surroundings of the winning neuron in the map. The self-organizing map is subject to three different processes in its formation; competition, cooperation and synaptic adaptation.

Competition
Let $\xi = [x_1, x_2, ..., x_m]^T$ be an input pattern in a m-dimensional space, $\mathbf{w}_j = [w_1, w_2, ..., w_m]^T$ the synaptic weight vector with the same dimensions as the input space and $j = 1, 2, ..., l$, where l is the number of neurons in the map. The winning neuron is then found by the neuron, whose euclidean distance to the input pattern is the smallest. This formalizes into Equation (3).

$$i(\xi) = \underset{j}{\operatorname{argmin}} \|\xi - \mathbf{w}_j\|, \qquad j \in \Omega \tag{3}$$

where Ω is the lattice of neurons [39].

Cooperation
The winning neuron is the center in a topological neighborhood, $h_{j,i(\xi)}$ of cooperating neurons. The gaussian function in Equation (4) can be chosen as such a topological neighborhood with its apex at the winning neuron and then the further away some neuron is from the winning neuron the less is the cooperation.

$$h_{j,i(\xi)} = e^{-\frac{d_{j,i}^2}{2\sigma^2}}, \qquad j \in \Omega \tag{4}$$

where $d_{j,i}$ is the distance from the winning neuron, i, to neuron j in the lattice and σ determines the stretch of the function. This stretch is often chosen such that the topological neighborhood shrinks with time. This is achieved by choosing σ to dependent on the discrete time, n, and choosing the function to exponentially decay with time as

$$\sigma(n) = \sigma_0 e^{-\frac{n}{\tau_1}} \qquad n = 0, 1, 2, ..., \tag{5}$$

where σ_0 is the initial value of σ and τ_1 is a time constant to be chosen. Since $\sigma(n)$ now depends on the time n, so will also the topological neighborhood function, $h_{j,i(\xi)}(n)$ [39].

Synaptic Adaptation

Now when there is a winner and the topology neighborhood is determined it is time define how the adaptation to the input pattern should work. The adaptation for the synaptic weights are updated as

$$\Delta \mathbf{w}_j = \eta h_{j,i(\xi)}(\xi - \mathbf{w}_j) \tag{6}$$

where i is the winning neuron, j the excited neuron and η is the learning parameter. The updated synaptic-weight vector at time $n+1$ is then given by

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta h_{j,i(\xi)}(n)(\xi(n) - \mathbf{w}_j(n)) \tag{7}$$

Also the learning parameter η can be chosen to decrease as the algorithm progresses. A suitable such function is

$$\eta(n) = \eta_0 e^{-\frac{n}{\tau_2}} \tag{8}$$

where η_0 is the start value for η and τ_2 is another time parameter that needs to be chosen.

During the training of a SOM it experiencing two different phases. The first phase is the ordering phase, where the lattice is ordering itself. The second phase is the convergence phase in which the SOM converges to the input patterns so that continuing training will almost not change the map [39]. The two phases for a one dimensional map can be seen in Figure 5.

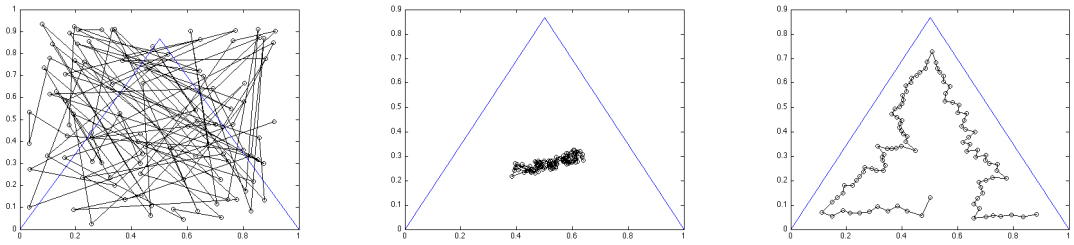

Figure 5: *A SOM going through the ordering phase in the middle figure and then the convergence phase in the right figure. The neurons were initiated randomly in the unit square. The training points resides uniformly inside the triangle.*

2.3.3 Generalization

A common problem in machine learning is the problem of generalization, that is how well a model performs on unseen data. A frequent method for

investigating the accuracy of unseen data is to divide the original data into two sets; a training set and a test set. The model is then trained on the training set and its generalization is then estimated by applying the model to the test set . Typically both the performance on the training and test set increases in the beginning of the learning procedure. However, after a while the performance on the test set start to decrease. This is called overfitting and an example of this can be seen in Figure 6 [40]. There are several reasons

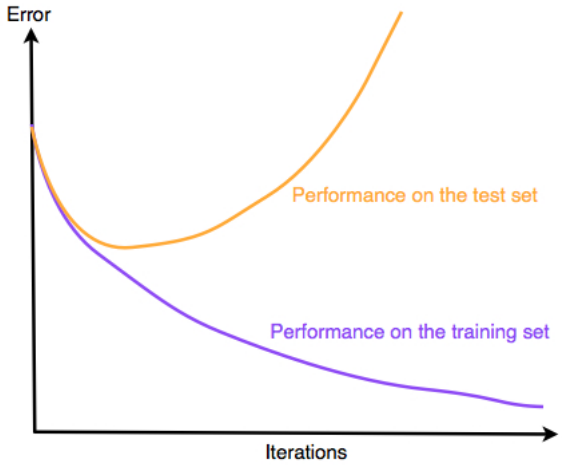

Figure 6: *An illustration of overfitting. The error on the training set continues to decline, while the error on the test set start to increase.*

why the performance on the test set might not be as good as on the training set. Some common reasons are listed below.

Presence of Noise in Data. The learning algorithm will try to fit the training data as good as possible. However, if there is noise in the data the algorithm will find a model that best describes this noisy data and not the actual process.

Lack of Representative Samples. The data used for learning is just a sample of the original population. Therefore there is a risk that the data does not contain representative cases. In this case the learning algorithm does not have any awareness of the missing cases.

Multiple Comparison Procedure. When the learning algorithm uses a methodology of evaluating and selecting different well performing models or parameters there is a chance that these were purely coincidences. This risk increases with the number of different models or parameters that are evaluated at some point in time [40].

2.4 Visualization of Stock Market Data

A common technique used for visualizing stock data is a candlestick chart. It summarizes the price movements during a time period and shows the highest, lowest, opening and closing price of the period. If the opening price is lower than the closing price the color of the bar is typically set to green. In the reverse situation the body is usually colored red. An example with two daily bars can be seen in Figure 7 [41].

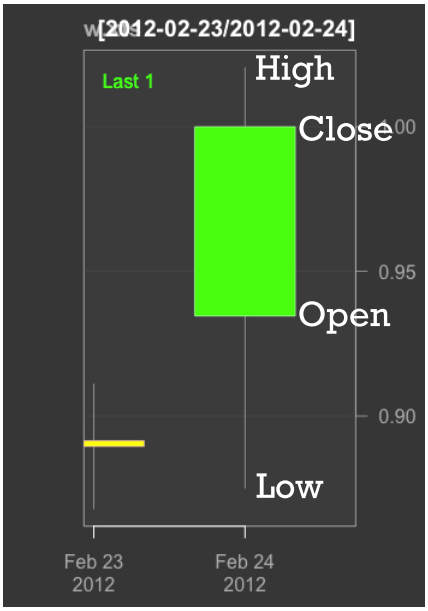

Figure 7: *Candlestick chart displaying two bars with daily data. The green color indicates that the closing price is higher than the opening price. The yellow color represents the other way around.*

3 Method

In the last section the theory needed to understand this thesis was explained. How this theory was implemented is discussed in this section.

3.1 Terminology

This section will clarify terms that are specific for this thesis. The first thing that needs to be clarified is the definition of a pattern. A pattern consists of one or many days of open, high, low and closing prices for every day. The length of a pattern is called *window length*, l_{window}, which refers to the number of days that the pattern is made up by. A pattern also has a *target length*, l_{target}, which is the number of days in the future the pattern is making a prediction about. A pattern can thus be plotted as candlestick chart with the length l_{window}. An example of a pattern can be seen in Figure 8.

Since a pattern can be plotted as a candlestick chart it can also be compared

Figure 8: *An example of a pattern with* $l_{window} = 5$ *and thus* $4 \cdot 5 = 20$ *dimensions.*

with other candlestick charts of the same length as the pattern measured in days. Using a sliding window and computing the difference between this input data and the pattern it can be decided if the data resembles the pattern. The *similarity threshold*, d, is the maximum distance from an input vector, ξ, to the j:th pattern, $\mathbf{w}_j$, for ξ to be classified as pattern j. The distance from the input to pattern j is computed as

$$d_j = \sqrt{\sum_{i=1}^{n}(w_{j,i} - \xi_i)^2} \tag{9}$$

where n is the number of dimensions. If $d_j < d$ the input window is classified as pattern j and a buying trade is done and held during the target length, that is the stock is bought at this day's closing price and sold in l_{target} days for the closing price of that day. The trade is then logged. This could possibly mean that several neurons is found that are close enough to the input pattern for that day.

Under the assumption that data just before the stock price increased significantly would be good data to train the SOM on, the variable *target size*, r_{target}, was introduced. This was accomplished by filtering the training sets to just include such time windows where the stock price after l_{target} days had increased above the threshold given by the target size. The target size, window length and target length is illustrated in Figure 9.

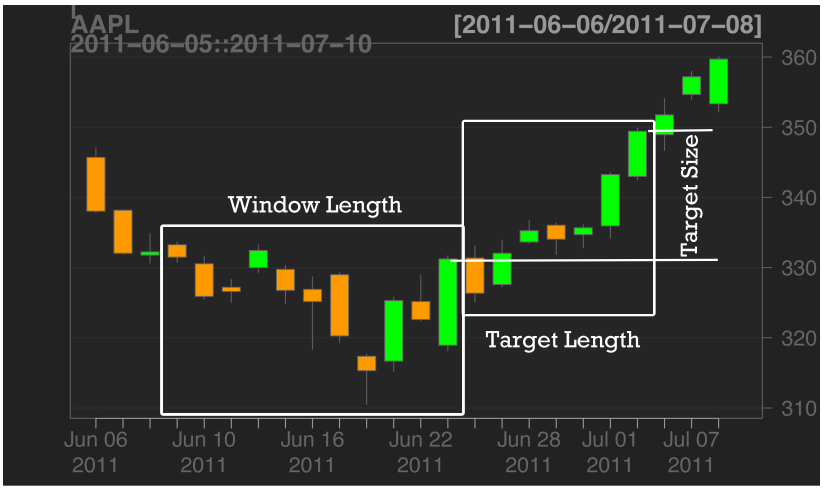

Figure 9: *A daily candlestick chart of Apple's stock price illustrating window length, target length and target size.*

The self-organizing map used is a square 2D map. The side length of this map measured in number of neurons is referred to as the *grid size*, l_{grid}. The total number of neurons is thus the grid size squared.

Every neuron in the SOM corresponds to a pattern. When the performance of the SOM is evaluated it is not desirable to use all patterns. Instead only a fraction, q_{used}, is used. These are chosen to be the $\texttt{round}(q_{used} l_{grid}^2)$ patterns that performs best on the training set.

Return was for simplicity defined as $r = \frac{P_{t+l_{target}}}{P_t}$, where P_t is the price of the stock at time t. This measures the return during the l_{target} period. To make it easy to compare results from different target lengths the return was

throughout this thesis recalculated to the daily return, r_d, as

$$r_d = r^{\frac{1}{l_{target}}}. \tag{10}$$

3.2 Data

Data refers to the financial data needed to perform the analysis. This subsection describes from where the data was gathered, how it was cleaned and how it was normalized.

3.2.1 Resources

Intraday financial data is expensive, but there are several services on the internet that provides historical daily stock market data for free. The data used in this thesis contains the open, high, low and close prices of every day and stock. This thesis used data from 2003 to 2011 of the 2796 stocks listed on the Nasdaq Exchange in North America. The data was downloaded from Yahoo!Finance.

3.2.2 Data Cleaning

Data from Yahoo!Finance is not always correct and is not compensating for splits. A split is a decision that increase the number of stocks outstanding. Practically this means that the price of the stock make a great change in price during the night. It is not desirable to use this data points to test the performance of a pattern therefore data previous to a point where the fraction between two days closing price was larger than two or smaller than 0.5 was disregarded.

3.2.3 Data Normalization

To be able to compare patterns with one similarity threshold the data has to be scaled in a good way. The scaling used in this thesis is done by dividing opening, highest, lowest and closing price for every day with the last day's closing price. An example can be seen in Figure 10.

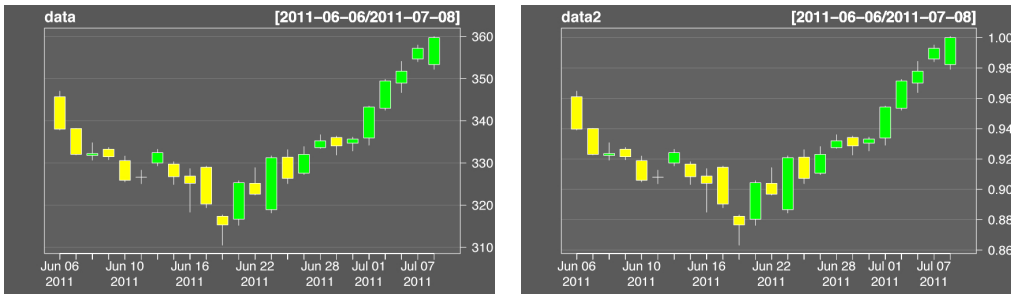

Figure 10: *The stock price of Apple before and after a normalization.*

3.2.4 Training and Test Set

One way of splitting the data into a training and a test set would be to use a portion of the stocks for training and the rest for testing. However, the correlation between different stocks can be quite high and doing this split would mean a risk that the model performs better on the test set than it should have done on unseen data. With this in mind it is better to split the data with respect to time. In this thesis the data is divided up in three equal sized periods where the first two are training sets and the last is the test set. The SOM is trained on the first and the optimization is carried out on the second training set.

3.3 Infrastructure

To implement the code solving the research questions some design considerations had to be made. Below follow a discussion of what was used and why.

3.3.1 Overview

Using a document with all the symbols listed on the Nasdaq Exchange, the stock data was downloaded using libcurl. Libcurl is a client side URL transfer library for C++ [42]. The data was then permanently stored in a local database. MongoDB is an open source NoSQL database implemented in C++ and was used to store the data [43]. The main reason to store the data locally was to avoid the network delay when running the simulations. As the algorithms, implemented in C++, progressed it logged its results in comma separated files, CSV-files. CSV-files are text files of data formatted so that attributes are separated by a comma and each row corresponds to an object.

These files were then imported in R for analysis. The infrastructure can be seen in Figure 11.

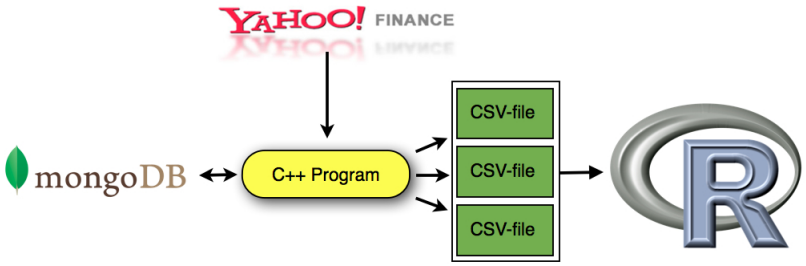

Figure 11: *The infrastructure used in this thesis. The computationally intensive tasks were carried out in a C++ program. The analyses of the results were then done in R due to its many convenient built in functions and plotting capabilities.*

3.3.2 C++

C++ is an object oriented programming language that extends C. It is portable and very performant [44] and was used in this thesis for the computationally intensive tasks. However, it is fairly low level and inconvenient to analyze and plot the results.

3.3.3 R-Project

R-Project is a free open source environment for statistical computing and graphics. It is similar to the S language developed at Bell Laboratories [45]. In excess of the built in functions there is approximately 2000 community supported packages at the Comprehensive R Archive Network, CRAN [46]. R was chosen as environment to analyze the results of the simulations due to its ease of use and great visualization capabilities.

3.4 The Algorithm

The algorithm refers to the code that aims to find predictive buying patters in the data. The algorithm used a combination of particle swarm optimization and self-organizing maps, as explained below.

3.4.1 Overview

The central part of this thesis was to find patterns in historical OHLC data that carries predictive information. This should be done without any predefinition on how the patterns might look. To accomplish this a self-organizing

map trained on historical data was wrapped in a particle swarm optimization algorithm. The SOM's neurons each corresponded to a pattern. The benefit of using a SOM instead of a random point is that there is no need to define the area in space that the patterns should randomly be generated in. Another benefit is that areas that are denser with points will also be examined closer by the SOM, due to the fact that more neurons will be present in that area. In addition to this it is also convenient to make 2D plots of the map used in the SOM to visually inspect if there are any clusters of neurons that are predictive.

By using only a SOM we need to explicitly set several parameters, e.g. window length, target length, etc. However, if the SOM is wrapped in an optimization algorithm that optimizes over these variables and using the patterns performance as its objective function, we will have an algorithm that is searching for the most predictive patterns without predefining how the patterns look or any other variables that are of interest. See Table 1 for which variables that were optimized.

The reason to use PSO as the optimization algorithm is that it is a stochastic optimization algorithm. Using a gradient optimization algorithm would likely lead to a non optimal optima. Genetic algorithm is another stochastic optimization algorithm, but since PSO has been shown to find its solution faster and with the equivalent precision as the genetic algorithm it was chosen [47].

As mentioned earlier the data was divided up in three equally sized parts. The SOM was trained on the first of the two training sets. These patterns were then used to trade on the second training set. By using a sliding window with length equal to l_{window} on every stock used under the whole period buying signals were generated if the euclidean distance from the data in this window to any of the patterns defined by any of the SOM's neurons were smaller than the similarity threshold, d. The returns from a fraction of the best patterns, decided by q_{used}, was logged. If the performance was better than any other previous subset of patters the patterns were also used on the test set and the performance was logged.

This logic was implemented in C++ and used approximately 2000 lines of code. Simplified pseudocode of the algorithm follows below.

```
particles = initPSO()
trainSet1 = getTrainingSet1()
trainSet2 = getTrainingSet2()
testSet = getTestSet()
while (1) do
    for (p in particles) do
        som = trainSOM(p, trainSet1)
        testPerformance(p,som,trainSet2)
        if (Performance is best so far) then
            logParticleValues(p)
            logPatternsUsed(p,som)
            testAndLogPerformance(p,som,testSet)
        end if
    end for
    updatePSO(particles)
end while
```

3.4.2 Implementation of SOM

The SOM algorithm was described in section 2.3.2. However, there are design issues that have to be considered when used on the problem at hand. The first problem that needs to be solved is how the input should be translated into an input pattern ξ. This was done by simply assigning every day's open,high,low and close price to one dimension each, as follows

$$\xi = [O_1, H_1, L_1, C_1, ..., O_{l_{window}}, H_{l_{window}}, L_{l_{window}}, C_{l_{window}}]^T \tag{11}$$

where O_i, H_i, L_i and C_i corresponds to the open, high, low, and close prices at day i and l_{window} is the length of the pattern. The number of dimension of the input is thus four times the window length. Haykin has made suggestions on suitable parameter values for the SOM algorithm [39]. He suggests that $\eta_0 = 0.1$ and $\tau_2 = 1000$ in Equation (8) and that τ_1 in Equation (5) can be set to $\tau_1 = \frac{1000}{log(\sigma_0)}$. Haykin also recommends that the learning parameter $\eta(n)$ should not decrease below 0.01. Another recommendation made by Haykin is to use 500 times the number of neurons iterations as stopping criteria. These recommendations were used in the implementation. Moreover, σ_0 in Equation (5) was chosen so that

$$\sigma_0 = \sqrt{\frac{l_{grid}^2}{2log(0.4)}},$$

which was derived from Equation (5) and Equation (4) setting $h_{j,i(\xi)} = 0.4$, $n = 0$ and $d_{j,i} = l_{grid}$. Also $\sigma(n)$ was lower bounded so that it would never

be smaller than 0.01, so that the neighborhood function would not approach zero as $n \to \infty$.

3.4.3 Implementation of PSO

The variables that were interesting to optimize were window length, target length, target size, fraction of the neurons to use, similarity threshold and grid size. These were inserted in a vector for each particle as can be seen in Equation (12)

$$\mathbf{x}^{[i]} = [l_{window}^{[i]}, l_{target}^{[i]}, r_{target}^{[i]}, q_{used}^{[i]}, d^{[i]}, l_{grid}^{[i]}]^T \qquad i = 1, ..., N \tag{12}$$

where N is the number of particles. $N = 20$ and $N = 40$ were tested with no significant difference in the result. Therefore $N = 30$ was used for the remainder of the simulations. The constants c_1 and c_2 was set to 1.5 both as recommended by Hassan et al. [47]. To decrease the chance of getting stuck in a local minimum the craziness operator, defined in Equation (2), was implemented with a probability of 1%.
The maximum and minimum values of the different variables can be seen in Table 1. l_{window},l_{target} and l_{grid} are integers and were therefore rounded after they were updated. The PSO algorithm does put restrictions on the velocities, but not the positions. Therefore restrictions on l_{window},l_{target} and l_{grid} to be greater than zero was implemented. However, upwards and for the rest of the variables in Table 1 there are no restrictions on the values they could take during the optimization process. The restrictions in the table are used when the positions of the particles are initialized. The PSO uses an

Table 1: *The minimum and maximum values of the variables to be optimized. The optimized variables in the order of appearance below were window length, target length, target size, fraction of the neurons to use, similarity threshold, and grid size of the SOM.*

	Min	**Max**
l_{window}	1	12
l_{target}	1	12
r_{target}	-0.2	0.5
q_{used}	0.01	1
d	0.02	0.2
l_{grid}	1	15

objective function, $f(\mathbf{x})$, to evaluate the performance of a position. Three

different objective functions were used:

$$f_{mean}(\mathbf{x}) = \text{Average Return} \quad (13)$$
$$f_{frac}(\mathbf{x}) = \text{Fraction Positive Trades} \quad (14)$$
$$f_{comb}(\mathbf{x}) = \text{Average Return} \cdot \text{Fraction Positive Trades} \quad (15)$$

where Fraction Positive Trades are the fraction of trades that generated a return larger than one. For statistical reasons $f(\mathbf{x})$ was set to zero if the total number of trades on the training set performed by a SOM was lower than 2000. The effect of this was that these SOMs were discarded. By doing so the risk of finding patterns that only worked by chance decreased. This is due to the fact that if, e.g. three trades were done with excellent average return it might have been by chance. If instead 100 trades were done with the same average return the likelihood that the average return is closer to the real one of the patterns are larger. The number 2000 was chosen somewhat arbitrary based on the average number of trades per neuron it would yield. The largest SOM has a $l_{grid} = 15$, which is equivalent to totally 225 neurons in the SOM. The average number of trades per neuron would therefore be at least $\frac{2000}{225} \approx 8.9$. Potentially excellent patterns with low frequency were discarded using this approach, because the total number of trades could be lower than 2000. However, the benefit of more trades, that could be used for more reliable statistical analyses, were preferred for the purpose of this thesis.

3.5 Random Trading

Random trading was performed on the same time period as the test set. This algorithm was implemented as follows

1. A random stock was chosen with uniformly probability.
2. A date was randomly selected in the period.
3. A trade was then performed holding the stock l_{target} number of days.
4. Return to step 1.

This was done for 20000 iterations and the return at each iteration was logged.

3.6 Analysis of Output

The algorithm implemented in C++ and described in section 3.4.1 logged its process and data in csv-files. These were imported into a R environment

using the R command *read.table*. To find patterns that were statistically significant both the average return and number of trades were important to analyze. A great amount of time were therefore spent examining different patterns from different simulations. R has built in functions for Q-Q plots and t-tests that were used. To be able to plot candlestick charts the package *quantmod* [48] was used.

3.7 Simulations

There are several variables relating to the simulation that can be varied, e.g. number of used stocks, number of particles in the PSO, the objective function in the PSO and of course the time period. The first simulations were mostly done to see if the written program was working. The period from the 13th of January 2009 to the 13th of January 2011 was used, the number of stocks was varied between 10 and 2900 and the number of particles from 20 to 40. This phase showed that using 250 stocks and 30 particles generated good results, so these values were used throughout the rest of the simulations.

Once it was verified that the program worked the following simulations were carried out.

- **Different Objective Functions.** The time period was set to 13th of January 2009 to the 13th of January 2011. For each of the three different objective functions used by the SOM four different simulations were carried out.

- **Different Time Periods.** To see if the algorithm performed better in some periods than other four different periods were tested; Jan 13th 2003 to January 12th 2005, January 13th 2005 to January 12th 2007, January 13th 2007 to January 12th 2009 and January 13th 2009 to January 12th 2011. Fraction positive trades were used as objective function. Four different simulations were performed during each period.

- **Predictability over Time.** To investigate whether the predictability of patterns generally decreased with time a set of patterns were generated during 2003 and were then tested on data from January 13th 2004 to January 12th 2005, January 13th 2005 to January 12th 2007, January 13th 2007 to January 12th 2009 and January 13th 2009 to January 12th 2011.

4 Results

This section outlines the results generated during the simulations. First the performance of some found patterns are examined. After that the results from using different objective functions are shown. Then the optimized variables are explored. Finally the generality of the patterns found are examined.

4.1 A Typical Simulation

The progress of a typical simulation can be seen in Figure 12. The performance on the test set indicates that several of the patterns found on the test data carries predictive information and therefore performs better than random trading. Up until iteration seven the performance on the test set

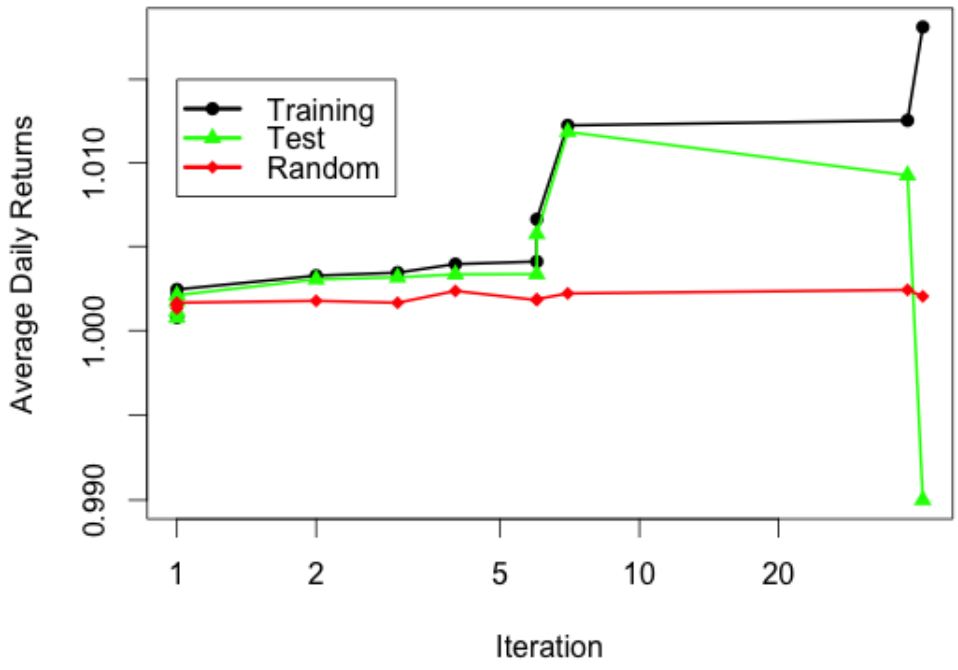

Figure 12: *The black line represents the performance of the best known subset of patterns found in the training data at that time. The green line is these patterns used on a test set. The red is random trading during the same time interval. The simulation is running with 250 stocks and 20 PSO-particles.*

follows the one of the training set. After this point the performance on the test set declines while the algorithm continues to find better patterns on the training set. What is seen is an example of overfitting.

The different patterns performance from one SOM in a simulation can be examined studying a feature map of the SOM in Figure 13. The feature map shows the performance of each neuron in the map. NA means that the neuron was not used. For this subset of patterns the predictive information seems to be gathered in one edge of the total data set.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	1.0095399
1	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
2	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	1.0095210
3	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
4	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
5	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
6	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
7	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
8	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
9	NA	NA	NA	NA	1.003386	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
10	1.005769	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
11	1.003053	1.004904	1.014631	1.0098401	1.024266	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
12	0.989398	1.019665	1.002891	1.0132523	1.013149	NA	1.014523	1.009597	NA	NA	1.016698	1.025689	1.024021	1.001024	NA
13	NA	NA	1.008563	0.9866085	1.008768	1.0092918	1.009607	1.028266	1.028335	1.022721	1.031858	1.003085	1.006918	1.025189	1.0022300
14	NA	NA	NA	NA	1.000882	0.9970033	1.018100	1.025221	1.040406	NA	1.028656	0.988060	NA	1.074830	0.9945004

Figure 13: *Feature map of the 15x15 SOM found in iteration seven in Figure 12. The feature to be displayed is the average return for the neuron. NA indicates that the neuron was not used.*

4.2 Performance

For statistical reasons it could be better to look on the performance of all used patterns. The reason for this is simply that the more trades that are analyzed, the greater is the chance of finding results of significance instead of statistical abnormalities. Studying the probability density functions from trading with the patterns compared to random trading gives deeper understanding of how the patterns might distort their density function in a desirable way. This can be seen in Figure 14. The difference does not seem to be dramatic. However, comparing the different daily means and calculating the expected yearly returns reveals another story. Average daily return using patterns is $r_{d,pattern} = 1.00381$ compared to random daily return of $r_{d,random} = 1.00157$. This would yield a yearly return of $r_{y,pattern} = r_{d,pattern}^{250} = 2.59$ for the pattern compared to $r_{y,random} = r_{d,random}^{250} = 1.48$ for random trading, that is an increase in price with 159% compared to 48%. 48% is still a very good return and can be explained by the fact that the test period happened to be en the last third of the period January 2009 to January 2011, which experienced a very positive trend. The performance of the patterns is under the assumption that a pattern could be found every day and that there are 250 trading days

during a year. A Welch's t-test explores whether the difference in mean

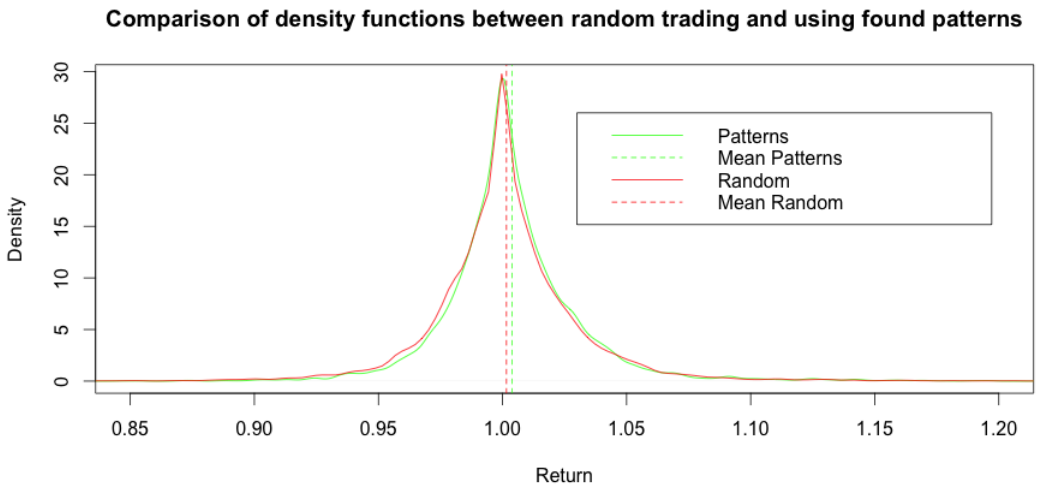

Figure 14: *Probability density of trades using patterns compared to random trading.*

is real or a statistical abnormality. Doing a Welch's t-test comparing the mean of returns generated from trading with the used neurons in the SOM and the mean from random trading in the same period yielded a p-value of $4.383 * 10^{-8}$. The probability that these mean would be the same is therefore almost negligible. However, the Welch's t-test assumes that the statistics are normally distributed, but since the number of trades are relatively high, > 5000, the mean of the returns should be very close to normal according to the central limit theorem. Conducting a Mann-Whithney test, which doesn't assume normality, gave results very close to the Welch's t-test. This confirms that the mean of the returns are in fact close to normal. In the remainder of this report the Welch's t-test will be used for simplicity.

4.2.1 Examples of Patterns

The simulations generated hundreds of different patterns. This section is not intended to analyze them all. Four of the patterns can be seen in Figures 15, 16, 17 and 18. Typically what the patterns do is that they decrease the number and size of negative trades. This can be seen in the Q-Q plots as a upwards curve in the left bottom away from the straight line. In Figures 15, 17 and 18 the best trades were actually done by random trading. This assures that the patterns' increased means are not due to a small number of outliers.

The pattern in Figure 16 used f_{frac} as objective function. The pattern increased fraction positive trades, ρ, from 66.2% to 84% compared to random trading with the same l_{window}. This can be visually confirmed in the Q-Q plot as a shift upwards of the whole data set.

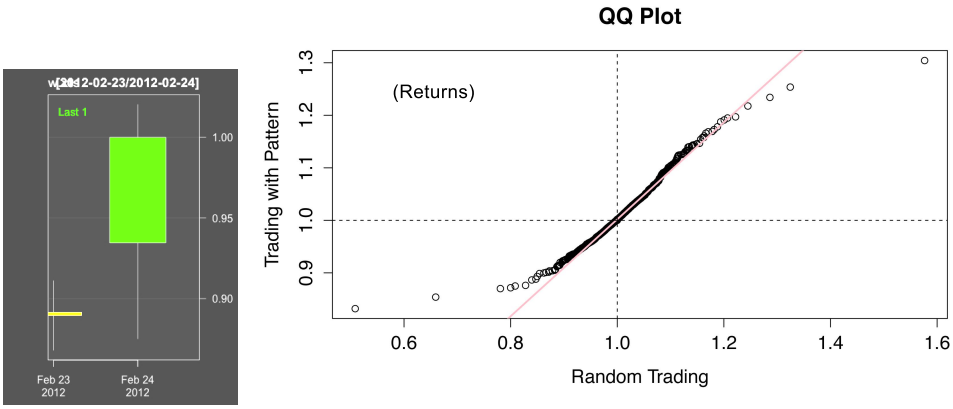

Figure 15: *Left: An example of a pattern with* $l_{window} = 2$, $l_{target} = 1$, $d = 0.192$, $N_{pattern} = 5760$, $\mu_{pattern} = 1.00375$, $\rho_{pattern} = 57.9\%$, $N_{random} = 20000$, $\mu_{random} = 1.00157$, $\rho_{random} = 54.9\%$ *and* $p = 2.7 * 10^{-7}$. *The pattern was in average applicable on* 15% *of the days on every of the 250 stocks. The QQ plot to the right plots the distribution of returns from random trading versus the returns of trading with the pattern.*

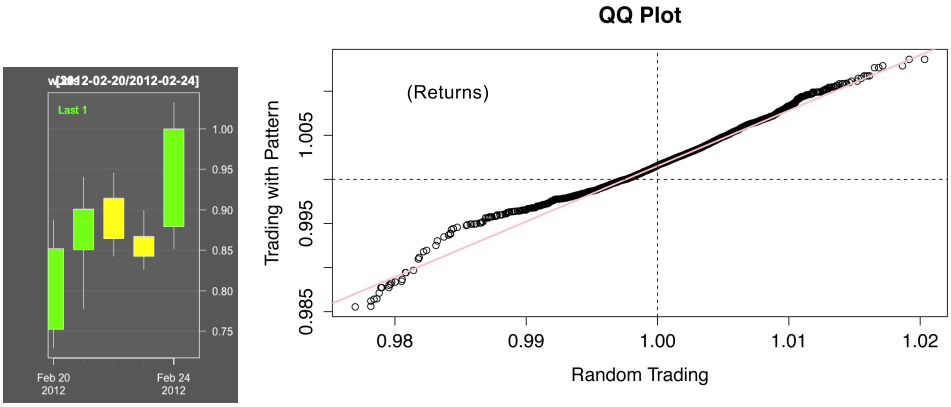

Figure 16: *Left: A slightly longer pattern with* $l_{window} = 5$, $l_{target} = 44$, $d = 0.409$, $N_{pattern} = 5703$, $\mu_{pattern} = 1.00239$, $\rho_{pattern} = 84\%$, $N_{random} = 20000$,$\mu_{random} = 1.00119$, $\rho_{pattern} = 66.2\%$ *and* $p = 2.2 * 10^{-16}$. *The pattern was in average applicable on* 44% *of the days on every of the 250 stocks. The QQ plot to the right plots the distribution of returns from random trading versus the returns of trading with the pattern.*

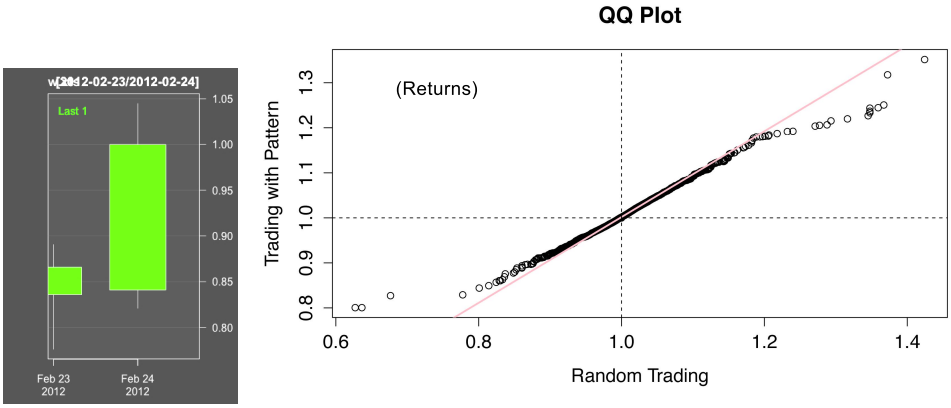

Figure 17: *Left: Another example of a pattern with* $l_{window} = 2$, $l_{target} = 1$, $d = 0.3499$, $N_{pattern} = 11350$, $\mu_{pattern} = 1.00345$, $\rho_{pattern} = 59.6\%$, $N_{random} = 20000$, $\mu_{random} = 1.001612$, $\rho_{random} = 55.9\%$ *and* $p = 4.1 * 10^{-8}$. *The pattern was in average applicable on* 31% *of the days on every of the 250 stocks. The QQ plot to the right plots the distribution of returns from random trading versus the returns of trading with the pattern.*

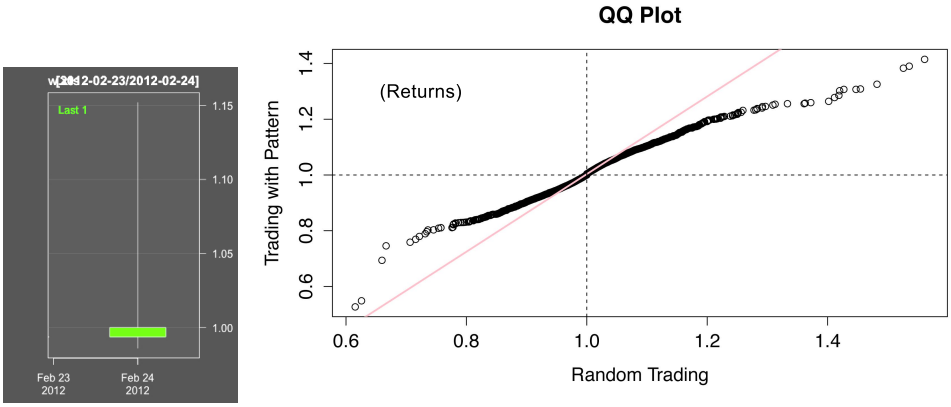

Figure 18: *Even one day pattern was found to be performant. Here with* $l_{window} = 1$, $l_{target} = 1$, $d = 0.1197$, $N_{pattern} = 38424$, $\mu_{pattern} = 1.00256$, $\rho_{pattern} = 53.3\%$, $N_{random} = 20000$, $\mu_{random} = 1.000125$, $\rho_{random} = 52.3\%$ *and* $p = 2.2 * 10^{-16}$. *The pattern was in average applicable on* 1.8% *of the days on every of the 2000 stocks. The QQ plot to the right plots the distribution of returns from random trading versus the returns of trading with the pattern.*

4.3 Different Objective Functions

To optimize with PSO you need to mathematically quantify which results are better than others. Three different such objective functions were tried, see section 3.4.3. These objective functions were used in four simulations each during the same period. The performance on the test set can be seen in Figure 19 and 20.

4.3.1 Effect on Average Return

The three different objective functions effect on the average return can be seen in Figure 19. As is evident from the figure, using the mean as an objective function gives better average returns. Second best is f_{comb} and worst is f_{frac}.

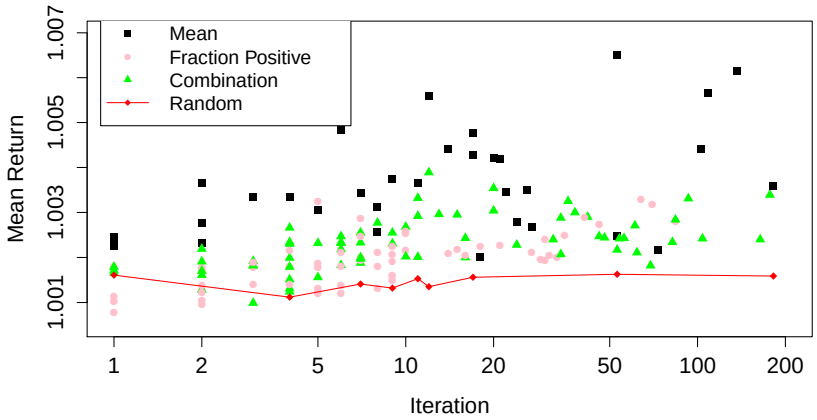

Figure 19: *Average return on the test set of the used patterns in the SOM as the algorithm progresses. Three different objective functions for the PSO algorithm were evaluated.*

4.3.2 Effect on Fraction Positive Trades

The objective function makes a great impact on the metric of fraction positive trades. Using the fraction of positive trades and a combination of this with mean profit yields a high predictability. Whilst using the mean is not performing better than random trading in this metric.

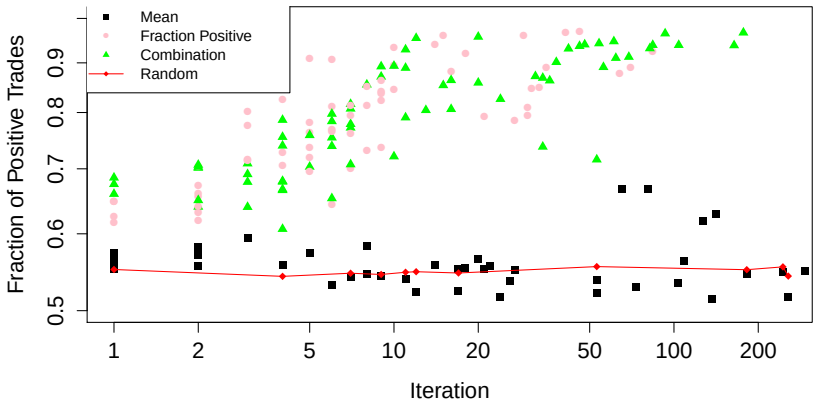

Figure 20: *Fraction of positive trades on the test set made by the used patterns in the SOM as the algorithm progresses. Three different objective functions for the PSO algorithm has been evaluated.*

4.4 Optimized Variables

The same twelve, $4 \cdot 3$, datasets using the three different objective functions four times each in the same time period, as in previous section, were used also for this section. By studying how the optimized variables evolve as the simulation progresses gives insights into where the most predictive information resides. The parameters of interest are window length, target length, return threshold, similarity threshold, grid size and fraction of neurons used.

4.4.1 Window length

Using fraction of positive trades as objective function tends to use a larger window length, see Figure 21. That is using more historical data in general than the others. With this as an exception there seems to be no other apparent patterns. This means that predictive information has been found in all window lengths examined.

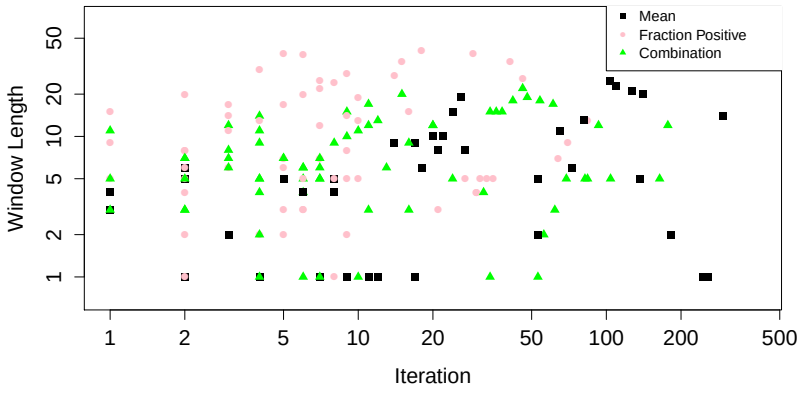

Figure 21: *How the window length changes as the the algorithm finds new optimums for the three different objective functions.*

4.4.2 Target Length

When it comes to target length the data is split into two clusters; one that uses mean as objective function and the other fraction positive trades and a combination of both. Using the mean favors a short time horizon, whilst the other cluster favors a much longer time horizon. This can be seen in Figure 22.

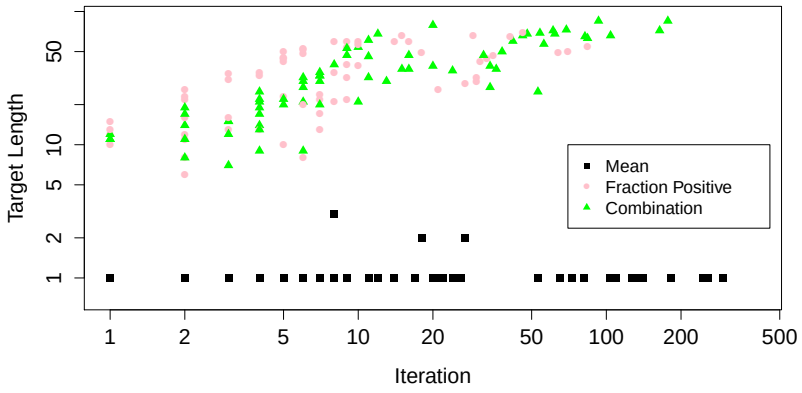

Figure 22: *How the target length changes as the simulation progresses. The different colors represent the three different objective functions used.*

4.4.3 Target Size

By using a filter that removes the points that has a smaller return than the threshold, r_{target}, we get a skewed training set. Using this dataset to train the SOM yields a better result when using the mean as an objective function, which can be seen in Figure 23. However, it is the reverse case for the other two objective functions. Here it seems that it is better to use all data.

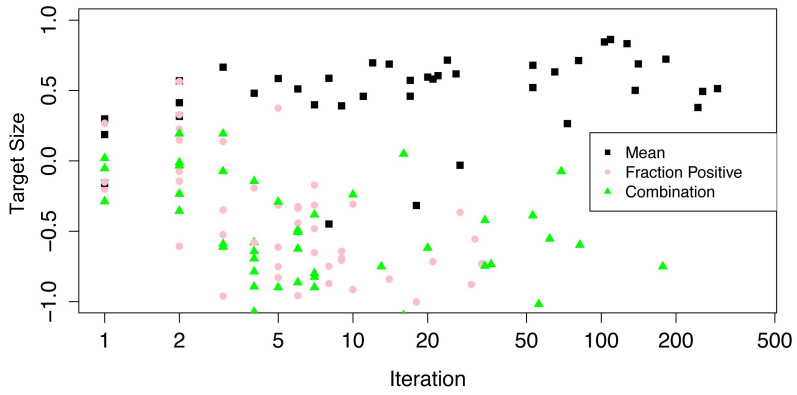

Figure 23: *How the target size changes as the simulation progresses.*

4.4.4 Similarity Threshold

How close a point must be to a used neuron to be classified as a pattern is determined by the similarity threshold. Except for one case of combination as objective function the value of the similarity threshold is quite uniformly distributed between 0.2 and 0.7 after the fifth iteration, as can be seen in Figure 24. However, the euclidean distance is not taking into account that patterns with longer window length has a higher dimension and thus will the distances also be longer. The window length is often longer when using fraction positive as objective function, but the similarity threshold is still comparable with the others. This means that the differences in each dimension on an average most be lower.

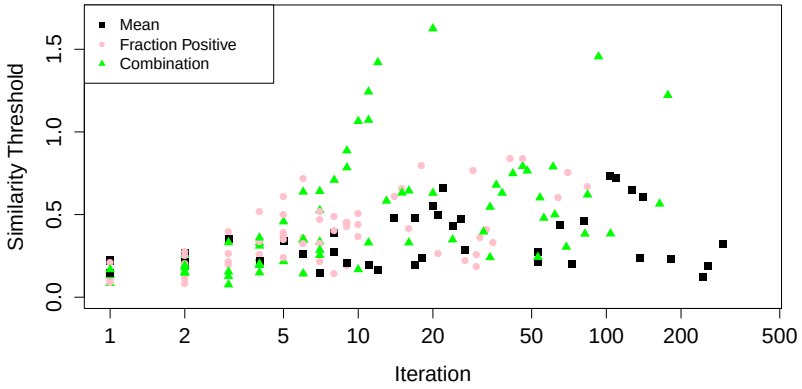

Figure 24: *The similarity threshold, d, of the euclidian distance from a point to that of a matching pattern for the tree different objective functions.*

4.4.5 Grid Size

The grid size is the number of neurons on one side of the square SOM. Thus the total number of neurons is the grid size squared. In Figure 25 it is evident the maximum grid size generated the best patterns. This is clear from the fact that most patterns uses the predefined maximum of $l_{grid} = 15$.

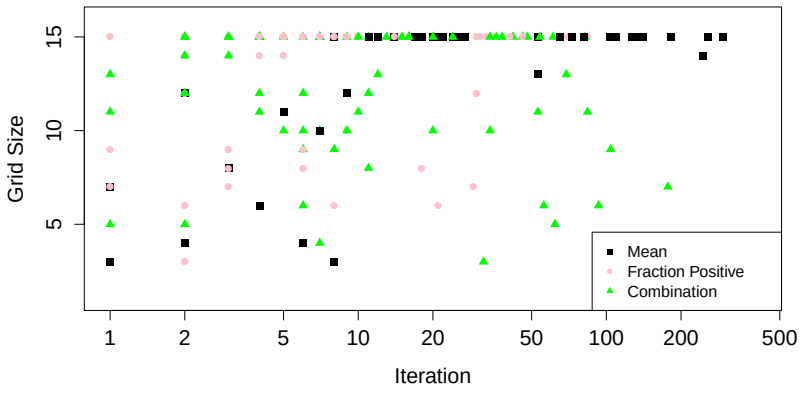

Figure 25: *The length of the side, l_{grid}, of the SOM map measured in number of neurons that was used.*

4.4.6 Fraction of Neurons Used

Not all neurons will be profitable patterns. Therefore only the best should be used in the subset of patterns that are used for trading. This is the reason behind why this variable needs to be optimized. Typical values range between 1% to slightly above 20%, as can be seen in Figure 26.

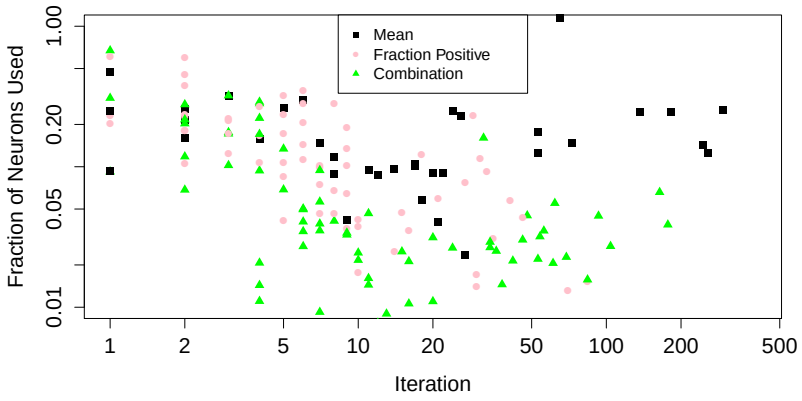

Figure 26: *How big fraction of the neurons in the SOM that was used for trading.*

4.5 Performance in Different Time Periods

For the sake of generality other periods have been tried as well. Three simulations per each two year period using 250 stocks and mean as objective function were carried out. The difference between random trading on the test set and using the patterns found in the training set but traded on the test set for the same period can be seen in Figures 27 and 28. The results indicate that profitable patterns can be found in all of the four periods. The patterns found and used during the two first periods seems to be more stable than the other two periods. The higher volatility during the financial crisis could be the reason for this. A monthly candlestick chart of Nasdaq Composite can be found in appendix A for a reference.

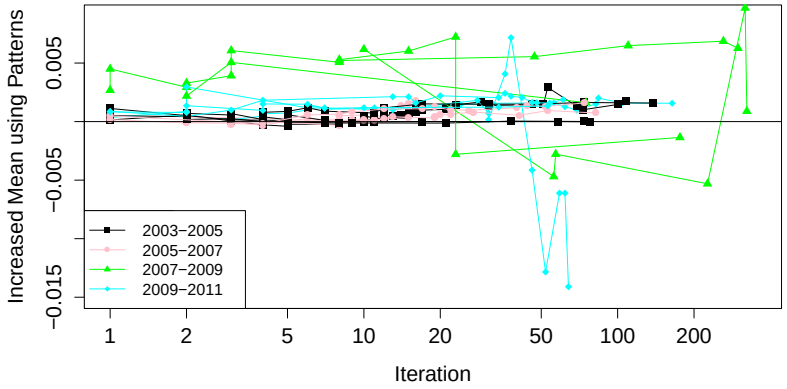

Figure 27: *Difference in performance from trading with patterns and that of random trading on the test set for four different periods. The first two thirds of the data in the period have been used as training set to find patterns. The remaining third was used as test set. 250 stocks were used and the f_{mean} was chosen as objective function.*

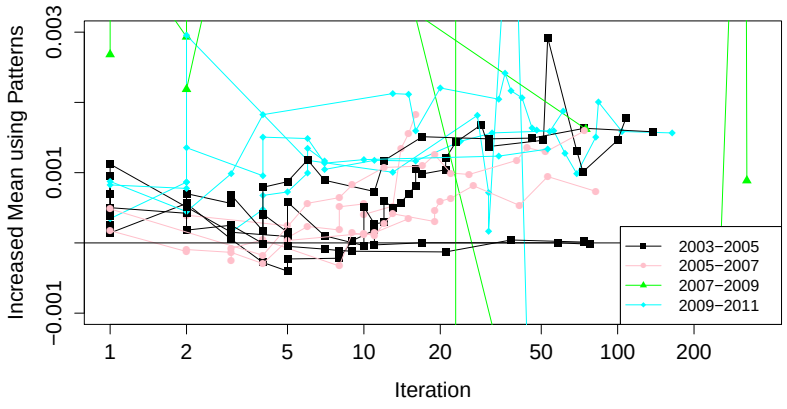

Figure 28: *A zoomed in version of Figure 27.*

4.6 Predictive Power of a Pattern as a Function of Time

To see how one subset of patterns' predictability is affected by time ten SOMs were trained on data from 2003. The performance of the ten SOMs is then tested on different time periods. The performance can be seen in Figure 29. The difference between the patters performance and random trading during the same period can be seen in Figure 30. The performance of the patterns

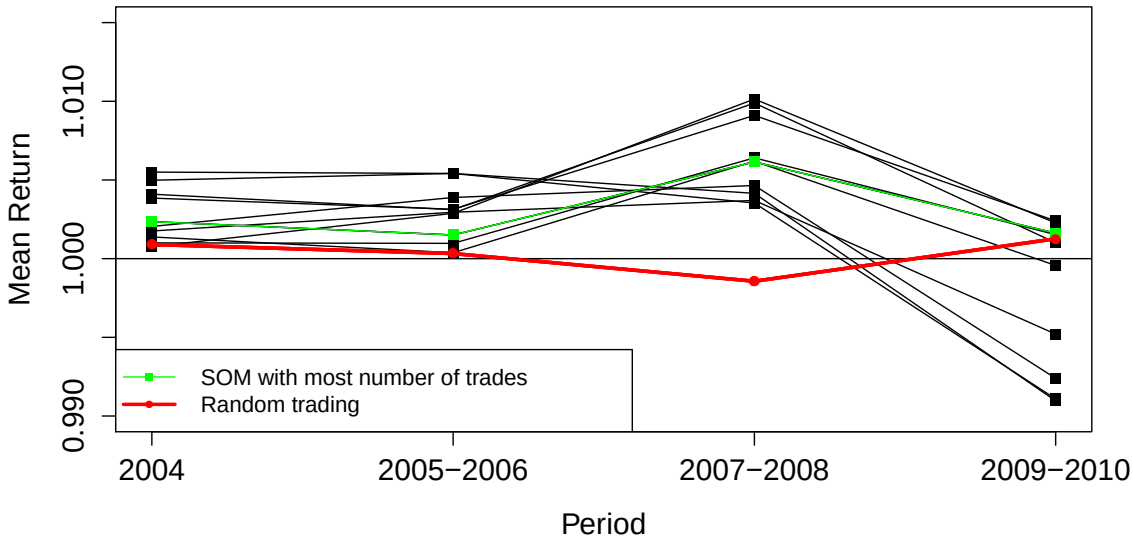

Figure 29: *The performance of ten different SOMs and random trading in four different time periods. The green SOM represents the SOM with significantly more trades in all periods than the other SOMs. The red line is the average of random trading.*

outperformed the random trading for all ten SOMs. However, during 2009 and 2010 random trading performed better than six of the SOMs. Moreover, five of the SOMs had a return below 1. This could be considered strange since Nasdaq Composite was in a very positive trend during this period, as can be seen in figure 31 in appendix A. This can be understood by looking on the previous period from 2007 to 2008. Nasdaq Composite was performing very good during 2007, but then the financial crisis struck and 2008 subsequently showed a strong negative trend. It is tempting to make the conclusion that the SOMs traded heavily during 2007 compared to 2008 and that this could

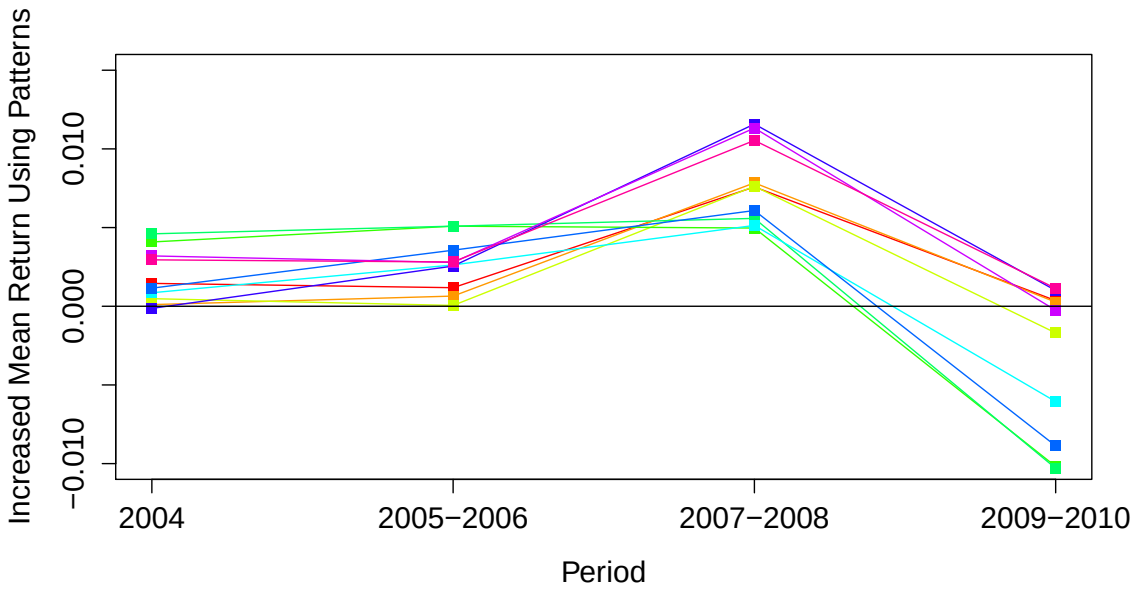

Figure 30: *The performance of ten different SOMs traded in four different time periods compared to random trading.*

explain the excellent performance during the period. However, the opposite is true. More than 80% of the trades were done during 2008 in the strong negative trend. A possible explanation to this could be that many of the patterns in the SOMs is looking for reversal signals in short time trends, that is a change in the trend from one direction to the other.

The green line in Figure 29 is a SOM which did significantly more trades compared to the other SOMs and could therefore be seen as a more representative example from a statistical standpoint. The number of trades for the four periods were 19539, 26782, 8465 and 32777, whilst for the other SOMs the number of trades varied in the range of 500 to 3000. The four worst SOMs in the last period all had a number of trades lower than 1500. The number of trades for the random trading was set to 20000.

5 Discussion

This section covers several discussion on the results in the previous chapter. Strengths, weaknesses and future work are also discussed.

5.1 Successful Methodology or Not?

Many patterns were found that statistically significantly outperformed random trading on the test set. This implies that the methodology suggested in this thesis does work. However, the practical use for a chartist using the patterns is arguable. Since the similarity threshold, d, often was relatively high a satisfactory input for a pattern definition could vary quite much. Thus it could potentially be difficult for a chartist by pure visual inspection to tell if the input is close enough to the pattern. The relative high d might come from the fact that a SOM needs to make at least 2000 trades to be considered. A higher d would make more trades, since the input does not to be as close to the pattern, and therefore increase its chance to reach 2000 trades and hence be evaluated. To mitigate from this, more stocks could be used or a longer time period for the second training set and test set could have been used. Alternatively an optimization around the parameters of the SOM found could be used to make it better. However, the methodology works unaltered for automated trading.

5.2 What Objective Function to Use?

What objective function to optimize is a subjective question. The simple answer would be the one with the highest average return on the test set. In that case one should use the mean return as the objective function to maximize. However, using high leverage, that is trading partly with borrowed money, one takes a significant risk, e.g. if only one third of the trades would be profitable you will often lose much before your earn much. This can result in serious liquidity issues. Then it might be a good idea to incorporate how big fraction of trades that are positive. Using the fraction of positive trades as objective function did show some problems in this thesis. Since the window length was not in practice limited by the PSO and the training data showed a positive trend, patterns with long window length was found. Also the test set showed a positive trend, so by using a long window length almost every trade was successful also on the test set. This pattern is obviously not a good one if the test set would be in a downtrend. Even though this is simple to understand in this case the same issue might arise elsewhere and where it is not easily spotted.

5.3 Generality of Patterns

The dynamics of the stock market is not the same at all times. For example it might show a positive trend for many consecutive years and then abruptly start decreasing. The dynamics can never be known in advance. Hence it is important that a pattern either works for all kind of dynamics or the present dynamics, in which the patterns are successful, is assumed to continue.

Section 4.6 showed that there exists patterns that, no matter what the underlying dynamics is, still perform great. Section 4.5 used the other approach by first training the patterns on data and then used the resulting patterns for the subsequent period. Later the patterns were then trained again, so that the present dynamics would be incorporated and assumed to be the same for the subsequent trading period, and so on. This approach was also shown to be successful. However, the performance of some patterns during the financial crisis varied.

5.4 Does Nervous Market Conditions Effect Predictability?

As discussed in section 5.3 the performance of a pattern can vary with time. The patterns in this section had its best performance during 2008 and its worst the subsequent two years. All of these three years could be considered to be a part of the financial crisis, with many unpredictable events with great impact on price movements. 2008 was a part of the crisis and still the patterns used had an excellent performance it seems that the crisis did not affect the possibility of trading with patterns. However, this particular set of patterns were better suited for a downtrend than in the following heavy uptrend during 2009 and 2010. Several of the patterns, including the one with most carried out trades, did perform well during this period as well.

5.5 Data Normalization

Stock data needs to be normalized to be comparable with other stocks and also with itself at another point in time. The normalization used in this thesis makes the last days of the pattern more forgiving than the first days, since all patterns last day's closing price are moved to exactly one. Thus, the variation in price between patterns will be small for the last days and much larger in the beginning days of the pattern. This might result in shorter window lengths than if this problem could be mitigated.

5.6 Alternative Methods

Instead of PSO a genetic algorithm could have been used instead. One advantage of using the latter would be that a number of chromosomes could be saved and used as starting points of generating new chromosomes. In this way not only the current best will influence the search for optimas. Using the PSO there is a chance that there is too much focus on the swarms best position. An alternative to the SOM would be to randomly generate some pattern based on the position of the particle. However, using the SOM avoids the process of how to generate patterns in a clever way. Moreover the SOM's neurons is placed around historical dense positions which implies that these patterns also will be found in the future. Something that a random generation has no notion of.

5.7 Future Work

There are some things that would be of interest to examine closer. One obvious thing would be to introduce patterns that give selling signals. Perhaps inverting the patterns giving buying signals could yield selling signals. Either way, the same methodology used in this thesis could be used to find selling signals as well.

After introducing a selling signal it would have been interesting to select a portfolio of the best patterns and use some ensemble algorithm to create a trading system. An ensemble algorithm combines weak predictors into one stronger.

The algorithm presented in this thesis is quite sensitive to overfitting. When a new global best pattern is found the PSO influence the direction of the particles in that direction. However, there might be better solutions in the vicinity of the old one that yet have not been explored. An approach to fine tune this patterns would be to use them as starting points and perhaps fixate some parameters. Alternatively a genetic algorithm could be used saving the e.g. seven best chromosomes from which new chromosomes could be generated.

Another thing that would be interesting to investigate would be if different stocks is more suitable than other. It might be so that there are clusters of stocks that have the similar dynamics and hence should be used to find patterns. It would be interesting to then see how these patterns perform on a test set of similar stocks compared to non-similar stocks.

6 Conclusions

This thesis has shown that an algorithm combining SOMs and PSO can be used to find predictive patterns that statistically outperform random trading during the same time period. The increase in performance for a specific subset of patterns showed that the annually return increased from a positive return of 48% to 159%.

Moreover, predictive information was found in all window lengths examined. This indicates that traders potentially could choose their preferred frequency of trading and still be successful. When it comes to target length the most useful information varies with the objective function of choice.

An important conclusion that every trading system developer should be aware of is the patterns might yield good results through many years, but when the market conditions change there is a risk that they will perform worse than random trading. This puts stress on the developer to make clever decisions when it comes to which data to train the algorithm with.

When dividing up the time from 2003 to 2011 into four different time periods, patterns that significantly outperformed random trading were found in all periods. This implies that even in times like the financial crises, with many non-predictable world events, it is still possible to make predictions using the methodology suggested in this thesis. This is an important conclusion that implies that the trader or trading system developer need to develop heuristics to determine what market conditions are present and choose trading tools accordingly.

References

[1] Fama, E. (1991) Efficient capital markets II, *Journal of Finance*, Vol. 46, pp. 1575–617.

[2] Fama, Eugene (1970) Efficient Capital Markets: A Review of Theory and Empirical Work. *Journal of Finance*, Vol. 25, No. 2, pp. 383–417.

[3] Bachelier, L. (1900), Théorie de la spéculation, *Annales Scientifiques de l'École Normale Supérieure*. 3 (17): 21–86

[4] Cootner, Paul H. (1964). *The random character of stock market prices*. MIT Press.

[5] Fama, E. (1965) The behavior of stock market prices, *Journal of Business*, Vol. 38, pp. 34 – 105.

[6] Basu, S. (1977) The investment performance of common stocks in relation to their price to earnings ratio: a test of the efficient markets hypothesis, *Journal of Finance*, Vol. 32, pp. 663–82.

[7] Ball, R. and Brown, P. (1968) An empirical evaluation of accounting income numbers, *Journal of Accounting Research*, Vol. 6, pp. 159–78.

[8] Ritter, J. (1991) The long-run performance of initial public offerings, *Journal of Finance*, Vol. 46, pp. 33–28.

[9] Dimson, E. and Mussavian, M (1998) Professional Forum: A brief history of market efficiency, *Europeen Financial Management*, Vol. 4, No. 1, pp. 91-103.

[10] DeBondt, W. and Thaler, R. (1985) Does the stock market overreact?, *Journal of Finance*, Vol. 40, pp. 793–805.

[11] Fernández, F., González-Martel, C. and Sosvilla-Rivero, S. (2000) On the profitability of technical trading rules based on artificial neural networks: Evidence from the Madrid stock market, *Economics Letters*, Volume 69, Issue 1, Pages 89-94.

[12] Kimoto, T., Asakawa, K., Yoda, M. and Takeoka, M. (1990) Stock market prediction system with modular neural networks, *IJCNN International Joint Conference on Neural Networks*; 17-21 Jun 1990, San Diego, CA. pp. 1 - 6, vol.1.

[13] Sewell, M. (2007) *Behavioral Finance*, http://www.behaviouralfinance.net/behavioural-finance.pdf (2012-02-20)

[14] Pring, MJ (1985), *Technical Analysis Explained: the Successful Investor's Guide to Spotting Investment Trends and Turning Points*, 2nd edition, McGraw Hill, New York.

[15] Leigh, W., Frohlich, C., Hornrik, S., Purvis, RL. and Roberts, TL (2008) Trading With a Stock Chart Heuristic. *IEEE Transactions on systems, man and cybernetics-Part A: Systems and Humans*. Vol. 38, No.1, pp. 93-104.

[16] Leigh, W., Modani, N., Purcis, R. and Roberts, T. (2002) Stock Market Trading Rule Discovery Using Technical Charting Heuristics. *Expert Systems with Applications*. Vol. 23, pp. 155-159.

[17] Wang, JL. and Chan, SH. (2007) Stock market trading rule discovery using pattern recognition and technical analyses. *Expert Systems with Applications*. Vol. 33, pp. 304-315.

[18] Fama, E., Fisher, L., Jensen, M. and Roll, R., (1969) The adjustment of stock prices to new information, *International Economic Review*, Vol. 10, pp. 1–21.

[19] Cowles, A. III (1933) Can stock market forecasters forecast?, *Econometrica*, Vol. 1, pp. 309 – 24.

[20] Jensen, M. (1968) The performance of mutual funds in the period 1945–1964, *Journal of Finance*, Vol. 23, pp. 389–416.

[21] Shiller, R. (1981) Do stock prices move too much to be justified by subsequent changes in dividends?, *American Economic Review*, Vol. 71, pp. 421–36.

[22] Pearson, K. (1905) The problem of the random walk, Nature, Vol. 72, p. 342.

[23] Kendall, M. (1953) The analysis of economic time series, *Journal of the Royal Statistical Society*, Series A, Vol. 96, pp. 11–25.

[24] Benartzi, S. and Thaler, R. (2001) Naive diversification strategies in defined contribution savings plans. *American Economic Review*, pp. 79–98.

[25] Barber, B. and Odean, T. (2001) Boys will be boys: gender, overconfidence, and common stock investment. *Quar- terly Journal of Economics*, Vol. 116, pp. 261–292.

[26] Beechey, M., Gruen, D. and Vickery, J. (2000) The efficient market hypothesis: A survey. *Reserve Bank of Australia*, http://www.rba.gov.au/publications/rdp/2000/pdf/rdp2000-01.pdf (2012-02-10)

[27] Levich, RM and LR Thomas (1993), The Significance of Technical Trading- Rule Profits in the Foreign Exchange Market: A Bootstrap Approach, *Journal of International Money and Finance*, 12(5), pp 451–474.

[28] Osler, CL and PHK Chang (1995), Head and Shoulders: Not Just a Flaky Pattern, Federal Reserve Bank of New York Staff Report No 4.

[29] Neely, C, P Weller and R Dittmar (1997), Is Technical Analysis in the Foreign Exchange Market Profitable? A Genetic Programming Approach, *Journal of Financial and Quantitative Analysis*, 32(4), pp 405–426.

[30] Allen, F and R Karjalainen (1999), Using Genetic Algorithms to Find Technical Trading Rules, *Journal of Financial Economics*, 51(2), pp 245–271.

[31] Montgomery, DC and Runger, GC. (2007) *Applied Statistics and Probability for Engineers*. 4th ed. Hoboken, N.J.: Wiley.

[32] Parzen, E. (1962) On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, Vol. 33, No. 3, pp. 1065-1076.

[33] Rosenblatt, M. (1956) Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, Vol. 27, No. 3, pp. 832-837.

[34] Wilk, MB and Gnanadesikan, R. (1968) Probability Plotting Methods for the Analysis of Data. *Biometrika*, Vol. 55, No. 1, pp. 1-17.

[35] Devore, J. and Farnum, N. (2005) *Applied Statistics for Engineers and Scientists*. 2nd ed. Belmont, CA.: Brooks/Cole.

[36] Welch, BL. (1947) The Generalization of 'Student's' Problem when Several Different Population Variances are Involved. *Biometrika*, Vol. 34, No. 1/2, pp. 28-35.

[37] Mann, HB and Whitney, DR (1947) *The Annals of Mathematical Statistics*, Vol. 18, No. 1, pp. 50-60.

[38] Wahde, Mattias (2008). *Biologically inspired optimization methods: an introduction*. Southampton: WIT Press.

[39] Haykin, S. (2009) *Neural Networks and Learning Machines*. 3rd ed. Upper Saddle River, N.J.: Pearson.

[40] Tan, PN., Steinbach, M. and Kumar, V. (2006) *Introduction to Data Mining*. Boston: Pearson Addison Wesley

[41] Edwards, Robert D., Magee, John and Bassetti, W. H. C. (2007). *Technical analysis of stock trends*. 9th ed. Boca Raton, FL: CRC Press.

[42] *LibCurl*. http://curl.haxx.se/libcurl/ (2012-02-11)

[43] *MongoDB*. http://www.mongodb.org/ (2012-02-11)

[44] *C++* http://www.cplusplus.com/info/description/ (2012-02-11)

[45] *R-Project* http://www.r-project.org/ (2012-02-11)

[46] *Comprehensice R Archive Network* http://cran.at.r-project.org/ (2012-02-11)

[47] Hassan, R., Cohanim, B. and de Weck, O. (2004) A Comparison of Particle Swarm Optimization and the Genetic Algorithm, http://web.mit.edu/deweck/www/PDF_archive/3%20Refereed%20Conference/3_50_AIAA-2005-1897.pdf (2012-02-12)

[48] *Quantmod* http://www.quantmod.com/ (2012-02-12)

Appendix

A Chart of Nasdaq Composite 2002-2012

Figure 31: *Monthly candlestick chart of Nasdaq Composite from 2002 to 2012. The vertical lines represents the four different periods used in the thesis.*