Universiteit Leiden
Opleiding Informatica

Finding Consensus for Financial Models Time-series Prediction

Name: Gregory Michel Maduro

Date: 27/07/2016

1st supervisor: Dr. Michael Emmerich

2nd supervisor: Dr. André Deutz

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

This thesis researches the error distributions of possible high frequency pricing models in the foreign exchange market. The models are tested on the EUR/USD exchange rate. The main focus of the research is to analyze the models on the Pareto-front using Multi-objective Optimization and Genetic Programming. The results of different non-dominated models are compared against each other and an analysis of the error distributions regarding the models found during six simulations over the high frequency data-set is given. The models found during the simulations of the Evolutionary Algorithm (EA) has shown to converge toward a (possible) global optimum. The EA was able to minimize the error of all three objectives (prediction error, skew and excess kurtosis) of the models. By deriving the models found after the simulations, it was possible to create a theoretical model for different holding periods.

Contents

1 Introduction

1.1 Motivation

In the past years there has been many hedge funds that claim to outperform the forex (foreign exchange) market. These trading firms mostly use proprietary software that handles automated high frequency trading algorithms to buy and sell shares when there exist inefficiencies in the market [10]. It is still unclear how these firms actually create their models. What is known is that they trade on very short time periods (See Figure 1).

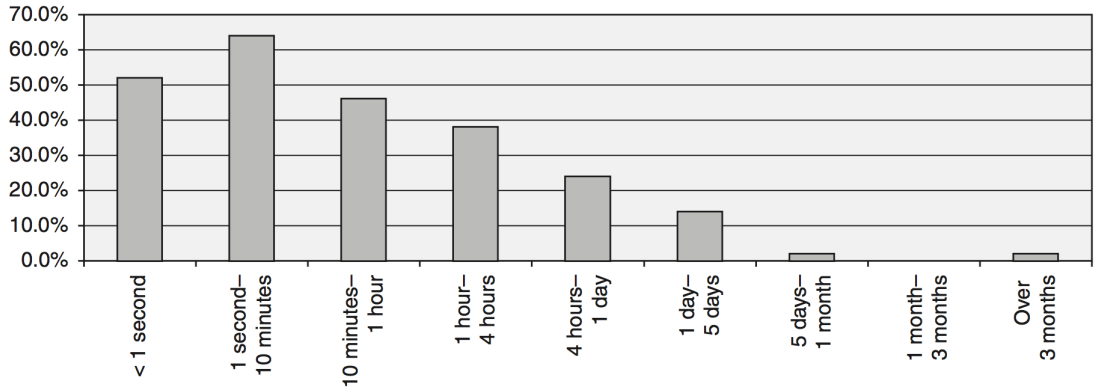

Figure 1: Survey of which holdings positions are classified as HFT, adopted from Aldrige, I. (2010, p.22)

In the past Neural Networks have been widely used for prediction in financial markets, because they are able to minimize the error of the target function [4]. The problem here is that the underlying structure of these networks can be very difficult to understand because they are too much of a black box, which is a disadvantage. In trading, prediction models should be stable and easy to understand because without stability, risk can be underestimated and can lead to more losses than profits. Risk management and prediction models used in trading are of equal importance. But to manage the trading risk, the error distributions of the prediction models should be well understood.

In the past decade trading has become autonomous process for many trading firms. Autonomous means that the models used do not have to be interfered and also possibly left unmonitored. But because black-box models are too much of a risk if left unmonitored due to the lack of theoretical foundation, it is possible that trading firms do not use them that often. Therefore the term Grey-Box models needs to be explained.

Grey-box models lay between White- and Black-box models. In contrary to White-box models that are deterministic because of physical knowledge, grey-

box models are not deterministic but they do make use of prior physical knowledge. As to Black-box models which do not use any prior knowledge in the data for it's output, Grey-box models do use a combination of it's prior physical information in the data with newly gathered information.

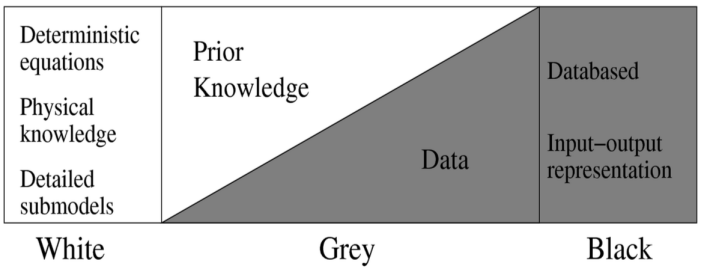

Figure 2: Grey Box, adopted from Bacher, B. et al. (2014, p.5)

Grey-box models are well suited for K-steps forecasts and control of both observed and hidden states. It helps the model with deficiencies when the model is missing proper description and or time-tracking of unexplained variations in e.g. parameters [7]. Therefore this paper will try to make use a Grey-box model in it's experiments. White-box models are not relevant because markets are noisy environments due to all of the factors influencing them, which makes it impossible to have one model that has always the perfect output.

1.2 Hedging Strategies

The amount of financial products available today is enormous. A good understanding of the difference between all of these products is beneficial to someone who is willing to become a trader because each product can be used differently and can influence the trading strategy. Most hedging strategies used today by hedge funds are built using options. Hedge funds are companies trading financial products claiming to use strategies that can generate high returns with low risk. Hedge funds differ mostly in the type of methods (e.g. aggressive or passive) and strategies (e.g. butterfly, straddle, iron condor etc.) they use, but they still use the same kind of financial products (e.g. options or futures contracts) to obtain the results they desire. The main benefit of using options is that they can reduce risk and generate higher returns than when just buying or selling a stock, this is why hedge funds use them.

Options are financial derivatives that represents a contract sold by one party, to another party. There are two different types of an option, the call option and the put option. The call option gives the holder the right to buy the underlying asset on a certain date for a certain price. The put option gives the holder the right to sell the underlying asset on a certain date for a certain price. The price listed in the contract is known as the exercise price or strike price, and the date

as the expiration date or maturity of the contract.

Suppose a trader at a hedge fund wants to buy on the first of February a one month call option Google contract with a strike price of \$520. He checks with his broker and sees that the offer price is \$32.00. This is the price for an option to buy one share. Normally in order to purchase one option contract, the size of the contract to buy or sell must equal 100 shares. Therefore, the trader must arrange for \$3,200 to be sent to the exchange through the broker. The exchange will then arrange for this amount to be passed on to the party on the other side of the transaction. If the price of Google does not rise above \$520 at the end of February, the option is not exercised and the trader loses only \$3,200 even if the price of Google went to \$0. But if Google does well and the option is exercised when the bid price for the stock is \$600, the trader is able to buy 100 shares at \$520 and immediately sell them for \$600 taking a profit of \$4,800 when the initial cost of the option is taken into account [6].

Options make it possible for traders to hedge their risk using different types of strategies that can create a spread that minimizes their risk. One possible strategy among others using options is the butterfly strategy. The butterfly strategy is a non-directional strategy where the trader has to take both sides (selling and buying) of the trade. This involves taking three positions with different strike prices using European options which can be exercised only on the expiration date. The strategy can be created by buying a European call option with a relatively low strike price K_1, buying a European call option with a relatively high strike price K_3, and selling two European call options with a strike price K_2 that is halfway between K_1 and K_3. The payoff pattern constructed by using the butterfly strategy (See Figure 3) will only give a positive payoff if the the price S_T stays inside the range $K_1 < S_T \leq K_3$ when it expires (See Figure 4).

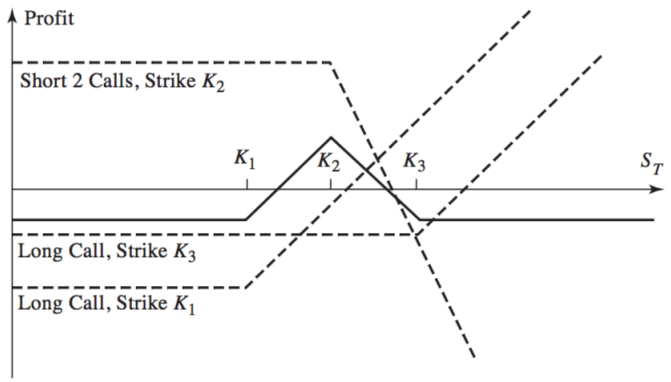

Figure 3: Butterfly Spread, adopted from Hull, J. (2012, p.242)

The payoff of the butterfly strategy can be maximized if the prediction of

the model is equal to K_2 each time at expiration, only then we can say that the prediction model has priced the asset correctly. It is extremely difficult to price an asset with high confidence, therefore if the distribution of the prediction errors is known beforehand the construction of the position can be adjusted in order to obtain a higher probability of expiring at or near the highest payoff K_2, so that a profit can be realized.

Normally in financial models, there are different issues when dealing with risk. Most of the time attention is only paid to the prediction error, but the distribution of prediction errors is also of importance for measuring the correct risk of the model. First, the distribution of the prediction errors can be skewed left or right, which means that the distribution is not symmetrical. Secondly, if there is high excess kurtosis the error distribution can have very long tails which can lead to big unexpected losses. Therefore all three types of errors should be minimized in order to estimate the risk associated with the prediction model better. A prediction model that has normally distributed errors should be the one wanted from a risk-management perspective, because unexpected losses could be better estimated than when the model found has a leptokurtic or platykurtic distribution of errors. When having a leptokurtic distribution of errors there is a higher expectancy of extreme values (errors), this could mean that the model found could be overfitting the training set and perform weakly on the test set. When having a platykurtic distribution the model could be doing the opposite of a platykurtic distribution of errors, this model can be generalizing to much on the training set which is also not what we want from a risk management perspective. The best distribution of error is therefore one in between the leptokurtic and the platykurtic distribution of errors like the normal distribution of errors. Therefore in this paper attention is mostly paid to the distribution of the prediction errors, where we want to optimize to a prediction error distribution close to that of the normal distribution. The butterfly strategy is only used to show the relevance of why the prediction error distribution must be optimized when searching for the correct pricing model. Without optimization of the prediction error distribution, it is more likely that the models will be exposed to generalization or specialization risk.

Stock price range	*Payoff from first long call*	*Payoff from second long call*	*Payoff from short calls*	*Total payoff**
$S_T \leqslant K_1$	0	0	0	0
$K_1 < S_T \leqslant K_2$	$S_T - K_1$	0	0	$S_T - K_1$
$K_2 < S_T < K_3$	$S_T - K_1$	0	$-2(S_T - K_2)$	$K_3 - S_T$
$S_T \geqslant K_3$	$S_T - K_1$	$S_T - K_3$	$-2(S_T - K_2)$	0

* These payoffs are calculated using the relationship $K_2 = 0.5(K_1 + K_3)$.

Figure 4: Butterfly Spread Payoff, adopted from Hull, J. (2012, p.243)

2 Background

This thesis is focused on finding a theoretical model for prediction in high frequency trading scenarios using an evolutionary method. To build the grey-box models proposed earlier, the characteristics of high frequency data in general must be known beforehand. This will help guide us in finding the superior model.

2.1 Value of High Frequency Data

The high frequency data studied in this thesis concerns the EUR/USD exchange rate. In the forex market currencies are traded in different markets, almost everyday during the year, except weekends. It is considered to be the financial product traded in a market that is close to perfect competition. One of the conditions for perfect competition is perfect information. This means in a market all consumers and producers are assumed to have perfect knowledge of the price and the effects that have influence on it. This is an interesting property because games of perfect information can be solved. If the process of the EUR/USD exchange rate can be seen as a game played with perfect information like Chess or Go. This could mean that there could also be dominant moves/models that can lead to winning strategies for traders. Therefore the more information we have about the process, the closer we are to solve this problem.

Using high frequency data has several advantages and disadvantages. Some advantages among others are:

1. The market is always and ultimately right.
Collecting as much information about market prices as sensible, is of extreme importance. For high frequency traders collecting this is necessary, because they are then capable of reacting faster above the competition [2].

2. Less biased data
Many lower frequency traders will use data that is biased by outliers of higher frequency intervals. A possible scenario in the forex market is that it was possible that during 23 hours of trading a very low volatility was seen compared to the last hour of the day where it was very high. For a day-trader using day intervals for their volatility models the observed volatility (uncorrected) would be a lot higher than the real volatility. This example is far from complete but it is critical to see that without all the information it would be easier to make incorrect trading decisions [13].

3. First on price formation.
When there is any new information about the price, whether it is good or bad

this will directly influence the price (price formation). Price formation can be seen as the acceleration of the current price. Lets assume there is good information only a few people know, than these people if rational will take long positions driving the price high, giving it a higher acceleration on the short term. Lower frequency traders would have not noticed this if they did not make use of high frequency data.

The disadvantages of high frequency data are less of influence on the price itself but on drawbacks to acquire and process the data. Processing-time can be drawback because the amount of data is enormous leading to lots of computation time, and in many cases it is also noisy which makes it difficult for a human eye to analyze easily. It may also be difficult to acquire data because the data is expensive compared to lower frequency data which is normally free. Another difficulty would be that they may have slips due to system errors when acquiring the data from the market.

2.2 Genetic Programming

Genetic Programming (GP) [9] is an artificial intelligence technique used to evolve computer programs (functions) to solve or converge to a specific goal. These computer programs are represented as trees or linear structures which can be easily modified using evolutionary operations. The new programs that are generated are hopefully better than the programs from previous generations. The evolution of the population is an iterative process with the population in each iteration called the generation. The evolution of processes in nature are not bounded by time, but for the tests in this paper it has to be stopped, otherwise we will never be able to know when to evaluate the models evolved. This is normally done using a stopping criterion like maxGeneration. When the amount of generations equals maxGeneration this

The general algorithm is easy to implement. It randomly creates an initial population of programs from the available primitives and calculates the fitness for each of the programs in the population. Afterwards the program will enter a while loop where first selection and secondly genetic operations will take place until a stopping criterion is met. At each generation the fitness of the new individuals will be again evaluated. When the stopping criterion is met, the program will stop and return the best so far individual/model.

As referenced earlier one of the disadvantages of high frequency data is that it is noisy. In the past GP was capable of dealing gracefully with certain amounts of noise in the data especially if steps are taken to reduce over-fitting, but cleaner data will mostly make the learning process easier for any system, GP included. One of the other particular values of GP is in its power of ex-

ploring poorly understood domains. A Lot of research has been done on lower frequencies but still a lot needs to be done on higher frequencies [9]. Therefore GP is a good method for research on higher frequencies.

2.3 Multi-objective Optimization

In many real world problems it is necessary to optimize different objectives. A simple example can be that of when someone is buying a car. If this person has only one criterion speed. This person would buy the fastest car in the world. But this is not always the case, the optimization problem becomes more difficult when there are multiple criterion. In the case where there is only one criterion like maximize speed, the car with the maximum speed, would be called the non-dominated car, because no other car can dominate this car in speed. When there are multiple criterion it is possible to have different non-dominated cars only if the Pareto-set is a mutually comparable set. In this paper the research is focused on maximizing the safety in a risk-management perspective. This is done using three objectives. If the model found satisfies the following criteria, then it will be non-dominated. Let J^1, J^2, J^3 be three objective (criterion) vectors. Here J^x represents the vector of non-dominated values for each

$$J^x = \begin{bmatrix} J_1^x \\ J_2^x \\ J_3^x \end{bmatrix}$$

criteria J_1^x,J_2^x and J_3^x for a model x in the population, e.g. J_1^1 is the average absolute error for model one. Then J^1 is non-dominated if $J_i^1 \geq J_i^2 \wedge J_i^1 \geq J_i^3$ $\forall\ i$ is true and $J_i^1 > J_i^2 \wedge J_i^1 > J_i^3$ for atleast one i is true [5]. If this criteria is satisfied the model will be added to the Pareto front used in this paper.

Multi-objective optimization forms an important part of this thesis because downside risk is a lot of the times underestimated in trading of financial products. The downside risk is the financial risk associated with losses. That is, it is the risk of the actual return being below the expected return, or the uncertainty about the magnitude of that difference.

It is possible that some researchers are only optimizing one objective instead of multiple objectives during the search process [12]. The problem is that optimizing one objective (e.g. average absolute error) does not say everything about the distribution of the errors. This can result in a model that has a very low error but disregards the tails of the error distribution which can be very long and that when applied in practice it can take only one of those very long tails to have an enormous downside risk. For this reason, we use multiple objectives to estimate the prediction error distribution better. This way we

can find a model with limited downside risk, by not specializing and also not generalizing too much as mentioned earlier (See Hedging Strategies, Section 1.2). Therefore the three objectives J_1^x,J_2^x and J_3^x to be minimized will be used to help find a model with a close to normal error distribution. To compute the non-dominated value of each objective we use the following three fitness measures; average absolute prediction error, skew (of the prediction errors) and the excess kurtosis (of the prediction errors) to determine the prediction error distribution of the model. The lower the value of the fitness function, the higher the non-dominated value will be for the objective that is being minimized. The three objectives can be computed using the statistical formulas seen in (1), more information on the exact computation is given in section 3.2.2.

$$
\begin{aligned}
Fitness(J_1^x) &= \frac{1}{N}\sum_{i=1}^{N}|\hat{p}_i - p_i| = \frac{1}{N}\sum_{i=1}^{N}|\ \epsilon_i| = \hat{u}_\epsilon \\
\hat{\sigma}_\epsilon^2 &= \frac{1}{N-1}\sum_{i=1}^{N}(\epsilon_i - \hat{u}_\epsilon)^2 \\
Fitness(J_2^x) &= \hat{S}_\epsilon = \frac{1}{(N-1)\hat{\sigma}_\epsilon^3}\sum_{i=1}^{N}\big(\epsilon_i - \hat{u}_\epsilon\big)^3 \\
\hat{K}_\epsilon &= \frac{1}{(N-1)\hat{\sigma}_\epsilon^4}\sum_{i=1}^{N}\big(\epsilon_i - \hat{u}_\epsilon\big)^4 \\
Fitness(J_3^x) &= |\hat{K}_\epsilon - 3|
\end{aligned}
\tag{1}
$$

When hedging using the butterfly strategy, knowing an estimate of all three optimization objectives is of great importance for risk management. Having a distribution that has a low error, a symmetrical form and has zero excess kurtosis would be ideal from a risk management perspective, because this has the characteristics of a normal distribution. When we look at the dark line for the payoff of the option construction in Figure 3, it has the form of a triangular distribution with the highest payoff at the top and a lower or negative payoff when S_t (Stock Price) deviates from K_2 (Strike Price). We need to realize that the butterfly construction can be many times not in the money when the model found contains excess kurtosis. Therefore finding models that minimizes the risk criteria will contribute to the payoff of the spread construction. The butterfly strategy is not the only strategy that exist, but it is a well understood method used to control the risk when trading by using a range instead of a direction. There are also other methods to manage risk [1] in trading but those methods are not relevant for this paper.

3 Methodology

3.1 Data

The total data used for searching and testing of the model contains intervals of 1 minute with a total of 99998 minutes in the range of (16:30) 12/1/2016 to (8:59) 15/4/2016, and it represents 24 hours of trading between Monday and Friday. The data used for searching (Training Data) consist of first 2/3 of the total minutes (See Figure 5) and the remaining 1/3 is used for the test-set. The data was obtained using the Bloomberg terminal and was not filtered.

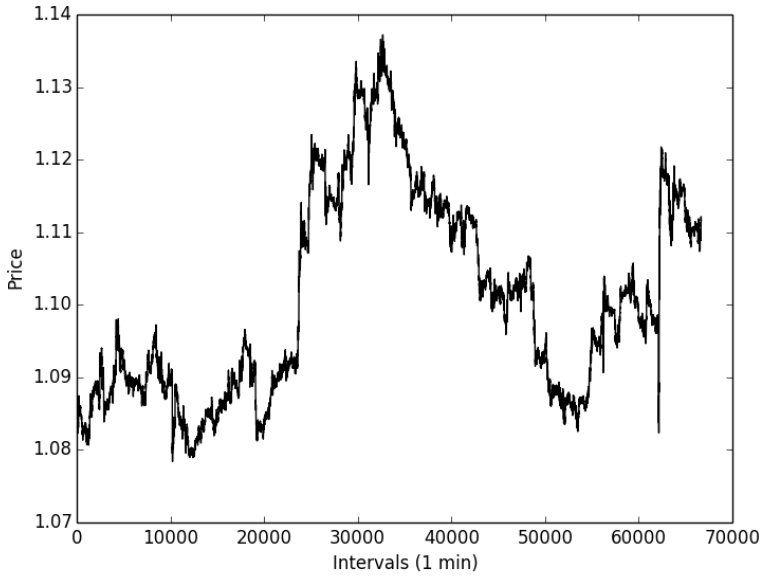

Figure 5: Training Data for the EUR/USD exchange rate

3.2 Experimental Setup

This thesis compares the performance of the "most interesting" model found with a random model from the population. To find the most interesting model a heuristic is used to determine a point on the Pareto front when the search process of the evolutionary algorithm is terminated. The main goal is to minimize the total risk of the model, this will be done by comparing the prediction error distribution between the two models. To minimize the total risk of the model, the three fitness measures seen in (1) need to be minimized. The best model will have for all three factors values close to zero. To find this model a new proposed heuristic $h(J^x)$ is used, see (2). Here SNDV and CPD are used to compute $h(J^x)$. SNDV is the sum of the non-dominated value for each criterion in J^x. CPD is the centroid (of maximum non-dominates values) penalty distance. By subtracting CPD from SNDV we get $h(J^x)$. The centroid $\vec{O}$ necessary for computing a part of CPD is computed using $max(J_i^x)$, this stands for the

$$\vec{O} = \begin{bmatrix} \frac{max(J_1^x)}{3} \\ \frac{max(J_2^x)}{3} \\ \frac{max(J_3^x)}{3} \end{bmatrix} = \begin{bmatrix} o_1 \\ o_2 \\ o_3 \end{bmatrix}$$

most non-dominated value for objective i in the population of models . If there is no other model in the population (of size n) that dominates $max(J_i^x)$, than $max(J_i^x)$ will have a non-dominated value of $n-1$. After dividing $max(J_i^x)$ by three $\forall\ i$, we find the three coordinates o_1,o_2 and o_3) which gives us the location of $\vec{O}$ in three-dimensional space.

The bigger the distance between the model on the Pareto front and the centroid of the Pareto front the bigger the penalty of CPD will be. This penalty therefore will help with finding a model that is balanced in all three criteria.

$$\begin{aligned} h(J^x) &= SNDV - CPD \\ SNDV &= J_1^x + J_2^x + J_3^x \\ CPD &= \sqrt{(o_1 - J_1^x)^2 + (o_2 - J_2^x)^2 + (o_3 - J_3^x)^2} \end{aligned} \tag{2}$$

It is expected that the highest value of $h(J^x)$ will return the most interesting model found (knee-point model) on the Pareto front. This is because the kee-point model will have a high non-dominated value in each criteria, while still not favouring one criteria before the other. The model with the highest value of $h(J^x)$ is therefore selected and tested on the test set.

3.2.1 Prediction Model

During the initialization of GP there must be a variety of different models initialized to help find the best model during the search process. The population of models consist of semi-parametric models. The lags of the parameters (average price and returns) are chosen randomly within a finite range because otherwise they will not be considered as high frequency lags, but the amount of the return parameters in the models can be infinite because we do not know exactly which lags, how many lags and in which combination of lags combined is able to find the best pricing model. Due the the high dimensionality of the data principal component analysis (PCA) is often used to determine which factors are most of influence in a factor model [14]. The disadvantage of PCA is that the amount of factors of influence is always chosen arbitrarily rather than naturally and therefore it will not be used. Another method will be used instead (See Initialization & Stopping Criteria 3.2.2) for finding the theoretical model of the time-series.

The core component of the model relies on the drift of the EUR/USD exchange

rate process, this can be seen as the average price u_l over some lag l, see (3).

$$u_l = \frac{p_t + p_{t-1} + ... + p_{t-l}}{l} \tag{3}$$

The (multiple) return factors that can be added to the model are noted as r_l over some lag l, see (4). This is the difference between the current price and the lagging price.

$$r_l = p_t - p_{t-l} \tag{4}$$

It may sound strange that a simple model like this would work, in an environment full of other factors that are non-continuous e.g. inflation, interest rates and growth announcements. But the return is always present which is not the same as the other factors that could be of influence during certain periods only. Just like a race car, we know it accelerates fast but this depends on other factors that are unpredictable a few months beforehand like weather and road conditions. But the acceleration of the car around the track year round, would give a better estimation of the future (general) acceleration. A general measure is better than a single measure. Assume the acceleration of the car was measured on a day when there was no wind and a dry track. This will usually lead to an overestimation of the acceleration if the race takes place on a windy day with a wet track. Therefore combining the two components, we get a general form of the theoretical model that will be used in the population to predict the future price p_{t+h} from the current price p_t:

$$p_{t+h} = u_l + w_1 r_l + ... + w_n r_l \tag{5}$$

The total amount of different return factors that have influence on p_{t+h} in the model is noted as n and all must be weighted by w because each will have an independent weight on the future price. The weight w_i associated with the returns are chosen randomly within the range 0.00–1.00. This range prevents the algorithm by adding weights that are too small (e.g. 0.000001), which can create models that have too many factors and are more likely to overfit the training set.

3.2.2 Initialization & Stopping Criteria

The size of the search space in this paper is hard to define, because the implemented algorithm will keep adding (infinitely many) factors to the model during the search process if it is not stopped. If the algorithm will not stop adding factors, the search space for finding the best model is infinite. Because

of this large search space, we implement at the initialization of the population a diverse set of models in order to test different possible models.

The models at initialization would consist of 2 forms, this was chosen because of the problem solving principle of Occam's Razor where "Among competing hypotheses, the one with the fewest assumptions should be selected". Therefore one form would contain only one component (u_l) and the other 2 components (u_l, r_l).

The value of the lagging distance l is selected to be in the range between 50 and 7200 minutes (5 days), with a distance of 50 minutes between each lag. This range is chosen because of Figure 1 which considers holding periods up to 5 days as high frequency holding positions. The distance of 50 minutes between each lag is chosen for the simple fact that if smaller time frames are added, the search space would become much larger and the model found is less likely to be close to the global optimum.

Each model in the population is evaluated for the three criteria using the training set. But because of the large amount of data, it is not efficiënt to compute the values of the objectives on every single data point. Therefore the three objectives of the model are computed using 1000 random points which would make the algorithm more efficiënt and still statistical significant. The target price p_{t+h} that the model should predict was chosen to be 240 minutes. A holding period of 240 minutes is still considered as a high frequency trading holding period. The benefit of longer holding periods is that they are more likely to be in the money than shorter holding periods. When the holding period increases the price volatility also increases which makes it more likely that the trade will cover the transaction costs. But longer holding periods also carry the disadvantages that they are more likely to be declassified as high frequency holding periods and that they are also more difficult to predict because there is more uncertainty during a longer holding period than on a shorter period of time. Therefore combining models of different holding periods would also be interesting for future research, because then there will be more information for risk management between conflicting predictions.

The size of the population is also important w.r.t. the evaluations and the efficiency of the program. If the population is too small it will not be diverse enough, but it cannot be too big either. A population that is too big can take a long time before it can evaluate all the models. This depends on the complexity of the model and how much the model needs to be evaluated during each generation. Therefore a constant population size of 500 is used. To control the size of the population three evolutionary operations (See Evolutionary Operations 3.2.3) will be used. During the search for the prediction model

each criteria is expected to be minimized until a certain point. Each criteria in the optimization problem has an influence on the error distribution of the predictions. One (possible) reliable distribution of the prediction errors can be the one that fits the butterfly spread option seen in Figure 3 as close as possible. With a prediction error distribution that is close to the normal distribution it is possible to expect a positive payoff with higher confidence than when we have a leptokurtic distribution that can contain more extreme (error) values. The problem here is that when one criteria is being minimized it will have an effect on the other criteria therefore multi-objective optimization is necessary to find a model with an error distribution that is close to the normal distribution. The search process is not stopped until the stopping criteria: generation $\leq$ maxGeneration is violated. The value chosen for the max amount of generations was arbitrarily chosen after a few trial and errors. After doing different tests it seemed that 1500 generations was enough for the EA (See Algorithm 1) to converge towards a (possible) global optimum. This was made possible using the recommendation for the mutation rate suggested by Schwefel, and is computed by adding a one in the numerator and dividing it by the squared root of the population size [11]. The Schwefel mutation rate is a robust parameter control scheme that affects the speed of adaptation during mutation. Choosing the mutation rate arbitrarily can lead to different population of models when running the EA. A low mutation rate will give a low adaptation rate, but it will return more precise models. A high mutation rate will give a high adaptation rate, but it will return more imprecise models. Therefore we will use Schwefel mutation rate to control the models in the population more optimally.

Algorithm 1 EA

$populationSize \leftarrow 500$
$maxGeneration \leftarrow 1500$
$maxLag \leftarrow 7200$ /* in minutes */
$k \leftarrow 10$ /* tournament size */
$SchwefelMutationRate \leftarrow \frac{1}{\sqrt{populationSize}}$
$P_0 \leftarrow$ initializePopulation(populationSize)
while $generation \leq maxGeneration$ **do**
  $probability \leftarrow$ Random(0,00:1,00) /* select a random value */
  if $probability \leq SchwefelMutationRate$ **then**
    Mutate(P,$maxLag$,k)
  else
    Reproduce(P,k)
  end
end

3.2.3 Evolutionary Operations

Darwinian evolutionary theory describes the mechanism of natural selection, and uses the fitness of the individual to measure reproductive success. This thesis makes use of three evolutionary operations during each generation of the EA. The evolutionary operations within EA are Remove, Mutate and Reproduce (See Algorithms 2,3 and 4). After performing an operation in the population, the fitness of the model reproduced or mutated is recomputed.

Algorithm 2 Remove(P,k)

model $\leftarrow$ TournamentSelection(P,Low,k) /* select low fitness model */
$P \leftarrow$ Pop(*model*) /* remove low fitness model from population */

Algorithm 3 Mutate(P,$maxLag$,k)

Remove(P,k)
model $\leftarrow$ TournamentSelection(P,$High$,k) /* select high fitness model */
range $\leftarrow$ Random(1:$maxLag$) /* select a random lag in range */
operator $\leftarrow$ Random(+,-) /* choose randomly between + and - */
model $\leftarrow$ *model* (*operator*) r_{range} /* add random factor to model */
$P \leftarrow$ Push(*model*) /* add mutated model to population */

Algorithm 4 Reproduce(P,k)

Remove(P,k)
model $\leftarrow$ TournamentSelection(P,$High$,k) /* select high fitness model */
$P \leftarrow$ Push(*model*) /* add high fitness model to population */

Notice that in Algorithm 1, it is possible only once during each generation to Mutate or Reproduce a model in the population. Both of these evolutionary operations include a Remove operation on the population before a new model is chosen for mutation or reproduction. The Remove operation removes a low fitness model from the population using the *Pop()* function to make sure that only high fitness models are later evolved in the search process. After Removal, the model used for mutation or reproduction is added to the population using the *Push()* function, this keeps the size of the population constant and helps the algorithm stay efficiënt.

For all the evolutionary operations, a selection procedure takes place. This selection procedure happens by k-tournament selection, with k = 10. The fitness measure used during tournament selection is calculated using almost the same heuristic as for finding the knee-point model shown earlier. The only difference in this case is that we do not look at the Pareto front, but just at the random

models selected for tournament selection. When reproducing *Reproduce()* we are selecting the model with the highest fitness and when removing *Remove()* we are selecting the model with the lowest fitness from the tournament. The same function *TournamentSelection()* is used each time for k-tournament selection in these functions, but there is only a sub procedure that determines the type of fitness (*Low* or *High*) of the model that has to be returned.

The *Mutate()* function is also dependent of k-tournament selection, however the mutation itself is random. The mutation in the models happens by adding a random return factor to the existing model which makes the size of the model grow. The random factor added during mutation can have either the addition operator or the subtraction operator. When using the two operators for mutation, the EA is able to self-correct itself randomly during mutations when searching for the best model.

A random model before mutation can look like.

$$p_{t+h} = u_l + w_1 r_l \tag{6}$$

And after mutation.

$$p_{t+h} = u_l + w_1 r_l - w_2 r_l \tag{7}$$

The self-correction that is now possible during mutation supports the model during both overestimation and underestimation of p_{t+h}. Assume the model is overestimating p_{t+h}, then if a mutation happens in this model with a subtraction operator, it will decrease the overestimation of p_{t+h} and vice versa. The advantage of this type of mutation is that it is simple, but it can also have bad adaption if the weights or lags are too high. If this happens the weight or the factor added may overpower the other factors in the model dramatically, making it harder for the model to be selected during future generations because of bad adaptation resulting in a lower fitness of the model.

4 Results

For testing the stability and performance of the results obtained by the evolutionary algorithm the algorithm was simulated six times. The simulations show that the algorithm was able to converge towards the goal of the optimization problem. During the search process (See Section 6.1) we see that the algorithm was able to minimize all three criteria during each simulation. We also see that the Pareto fronts (See Section 6.2) at the end of each simulation are similar.

But it may occur sometimes that the points on the Pareto front may be further away from each other than in the other simulations. We can also see that the error distributions of the knee-point models were close to that of the normal distribution. When comparing the knee-point models against the random models we see a big difference in the form of the distribution. Almost none of the random models had a form close to that of the normal distribution. The random models were in not even one criteria better then the knee-point models found (See Section 6.3). The difference in prediction and target between the two models from one of the six simulations can be seen in Figure 6 and 7 . In Figure 6 we can see the predictions of the knee-point model and in Figure 7 the predictions of the random model. The disadvantage of the random model was that the predictions were being to much generalized compared to the knee-point model which had more specialized predictions. Predictions that are too much generalized can give distributions like those seen in Figure 28,29,30 and 31 (See Section 6.3.2) . These distributions are definitely not wanted, because they are far from being normal distributed.

The usefulness of using genetic programming is that we can see directly which factors are more of an influence on the prediction target. By analyzing the underlying structure of the knee-point models found we can possibly find better models in the future, by looking at more peculiar search spaces.

$$\begin{aligned} p_{t+240} &= u_{1050} + 0.33r_{2400} \\ p_{t+240} &= u_{3100} + 0.54r_{2950} \\ p_{t+240} &= u_{1850} + 0.65r_{2100} \\ p_{t+240} &= u_{1050} + 0.41r_{1850} \\ p_{t+240} &= u_{3750} + 0.56r_{2900} \\ p_{t+240} &= u_{2750} + 0.86r_{1450} \end{aligned} \tag{8}$$

When looking closely at the knee-point models found (8) after the six simulations, we find some interesting characteristics. The most interesting one, is that all the models had only one return period as factor even though there were models that had more than one return period as a factor, and also models with only the average price as predictive factor. This satisfies Occam's razor principle that among competing hypotheses, the one with the fewest assumptions should be selected. When looking at models here above we see that the average price lag range is between 1050 and 3750 minutes, and for the range of the lagged returns this is between 1450 and 2950 minutes. Thus the future price, is highly dependent on this range, and there are many different combinations of lags that still leads to a close to normal distribution for the errors of the knee-point models found when using this range. We also see that for short time frame predictions of 4 hours that the lagging indicators do not range in the

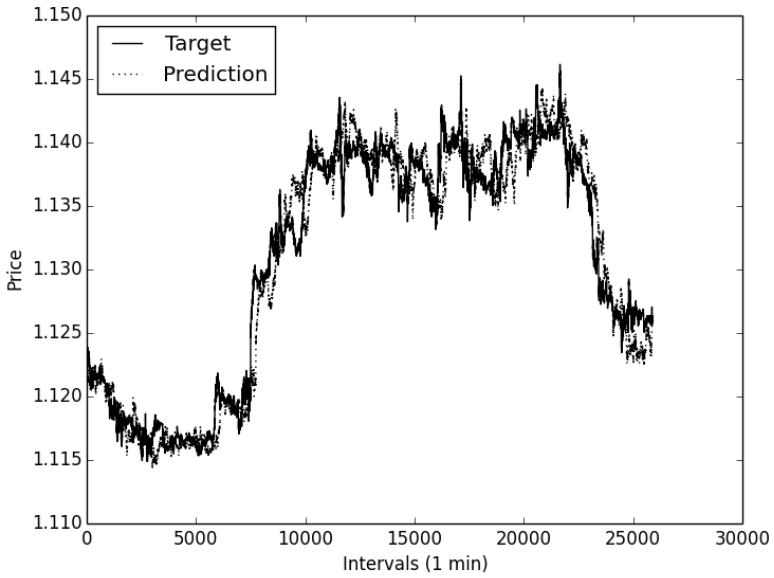

Figure 6: Knee-point Model

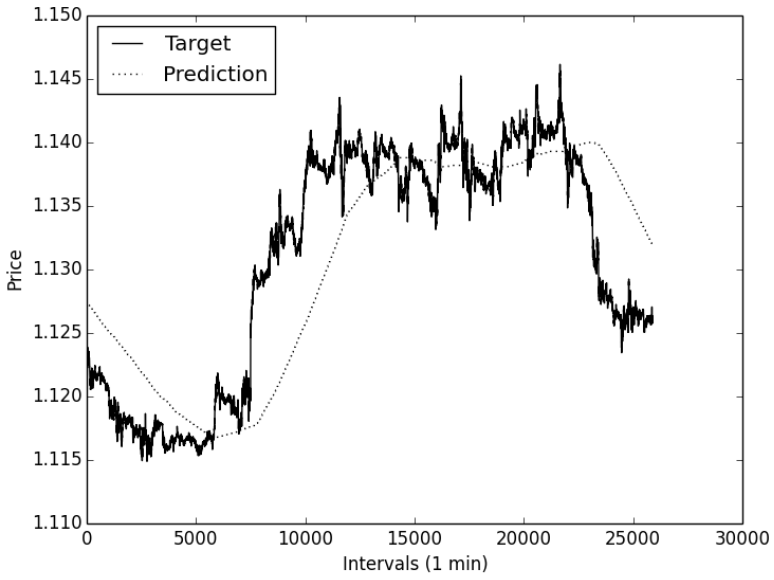

Figure 7: Random Model

period lower than 1050 minutes and neither in the range higher than 3750 minutes. Notice that the initialized population had a minimum lag of 50 minutes and s maximum lag of 5 days. This shows that there is a big space of lagging indicators that had little significance when searching for the knee-point models. There are therefore some lagged distances that should not be included in the model, because it will give the type of error distribution that is not desired from a risk management perspective.

The second most important realization is that of when we take the average of the lagged distance over all the simulations for the average price, return and

the weights, we get:

$$p_{t+240} = u_{2246} + 0.55r_{2275} \tag{9}$$

The interesting part of this, is that we can rewrite this into a more general form like:

$$p_{t+h} = u_{\alpha} + \frac{1}{2}r_{\alpha} \tag{10}$$

The importance of finding a theoretical model like (10) could support Mandelbrots theory of scaling markets [8]. According to rule 5 of his theory he states that Market time is relative. This means that without the identifying legends, one cannot tell if a price chart covers eighteen minutes, eighteen months, or eighteen years. Therefore a formula like this could characterize the behavior of the price evolution over shorter or longer periods just by optimizing α. In this case α is the optimal lagging distance both for the mean and the return. The benefit of having a lagging distance that is the same for both the mean and the return, is the time it takes to optimize the model for a specific holding period. Before we began this research we did not know which type of models or lags were naturally better. Now we know that it is not necessarily needed to search in deep areas of the search space (e.g. models with many variables), which makes it more efficiënt when optimizing the lags of the theoretical model found because there is only two. It is still possible that the lag of the mean and to the lag of the return will not always give a error distribution that is close to the normal distribution, however it is still much more efficiënt to have equal α as a starting point for both the mean and the return in further optimization of the lags. This model also confirms that the EUR/USD exchange rate on a high-frequency level is mean-reverting. Mean reversion refers to the possibility that, while prices fluctuate unpredictably in the short run, they tend to oscillate around long-term trend lines and the further they deviate from the trend lines the more they are pulled toward them [3]. In this case we see that the future short term price is dependent on the long term intraday- mean and return, and if markets do actually scale, equation (10) would be a model that can characterize target prices for different kind of holding periods of the EUR/USD exchange rate. In the case that this theoretical model of scaling markets is rejected, the algorithm should still be able to find models close to the global optimum for other holding periods.

5 Conclusion

The evolutionary algorithm (EA) used in this paper was successfully implemented. After running the EA for six simulations, all the models found per-

formed well in a risk-management perspective on the test set. The EA was able to decrease the error in the three objectives (error, skew and excess kurtosis). The knee-point model found on the Pareto front was similar in all the six simulations, the only thing that varied was the lag of the factors influencing the short term prediction. This was mainly caused by the size of the search space, but the EA was still able to find models in a certain range that were better than randomly selecting a model from the starting population used during the initialization of the EA. During each simulation there was a great variety of different models that were tested, but we still saw that the knee-point models found at the end were very similar to each other. Due to this similarity it was possible to derive a theoretical model for the EUR/USD exchange rate. The beauty of the theoretical model (10) derived is that it agrees with great theories as those from Occam's Razor and with the scaling of markets proposed by Mandlebrot [8]. Therefore the theoretical model derived seems to be an appropriate model, but this still has to be statistically tested for holding periods above and below 240 minutes before it could be accepted.

6 Experiments

6.1 Search Process

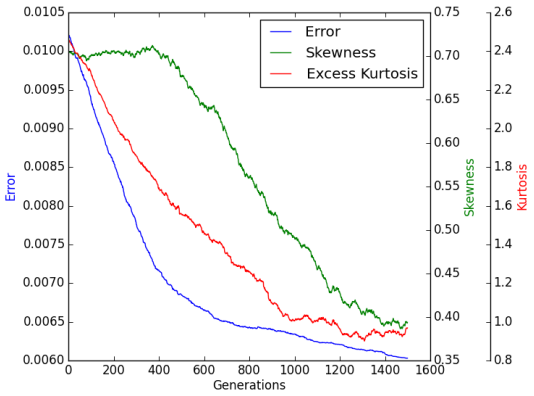

Figure 8: Simulation 1

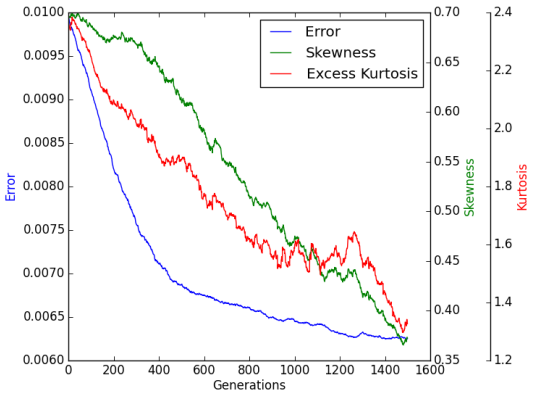

Figure 9: Simulation 2

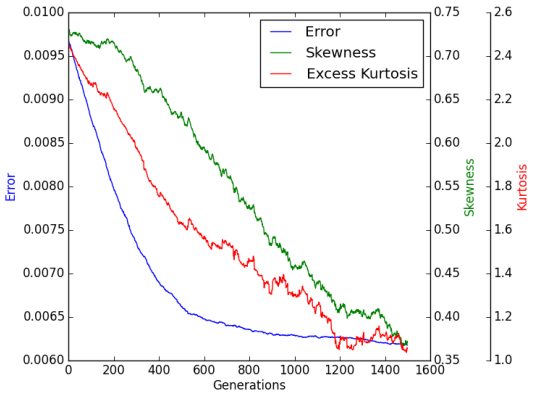

Figure 10: Simulation 3

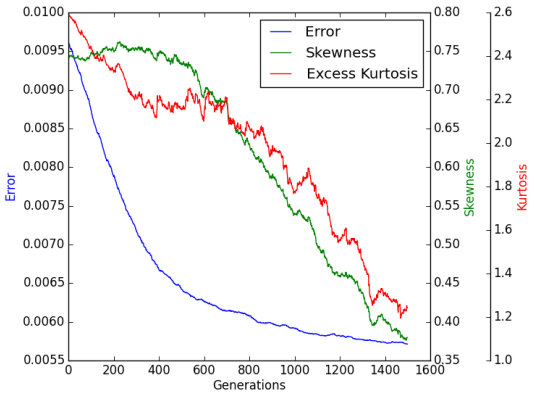

Figure 11: Simulation 4

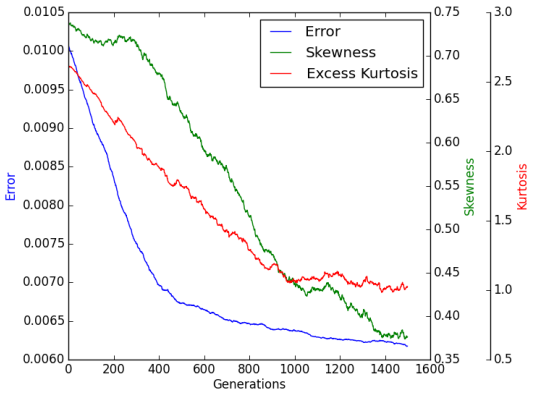

Figure 12: Simulation 5

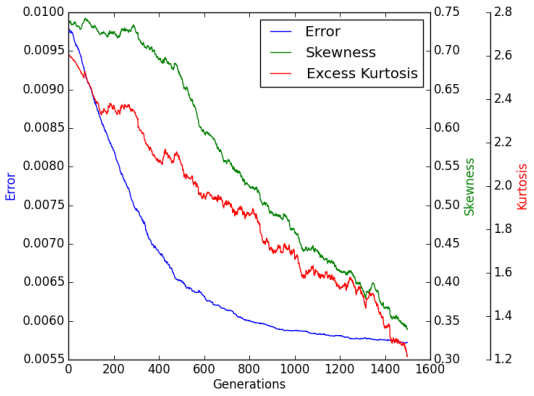

Figure 13: Simulation 6

6.2 Pareto Fronts

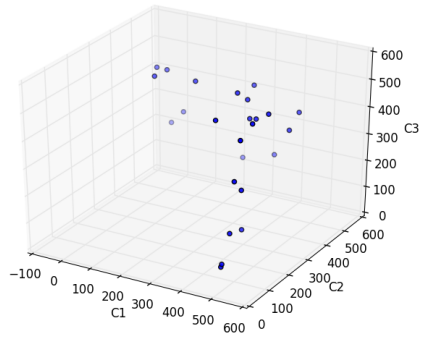

Figure 14: Simulation 1

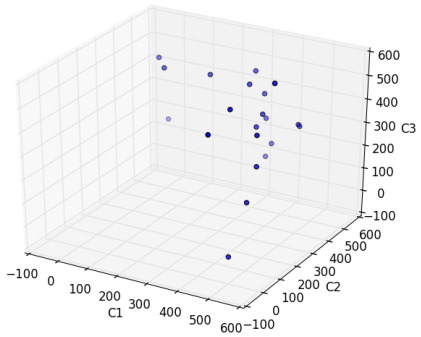

Figure 15: Simulation 2

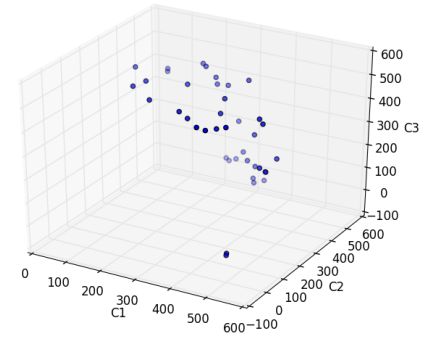

Figure 16: Simulation 3

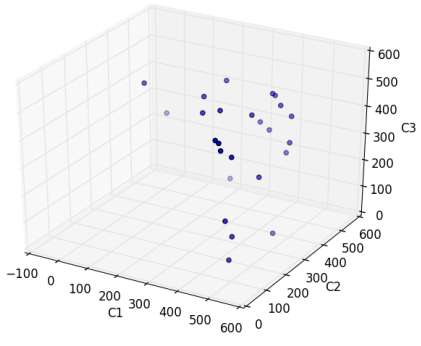

Figure 17: Simulation 4

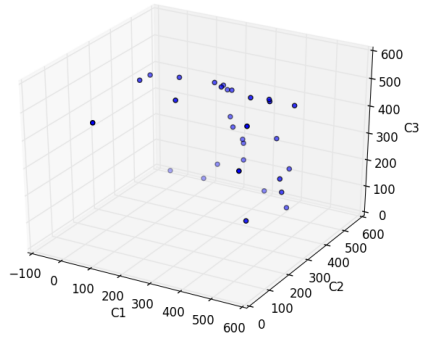

Figure 18: Simulation 5

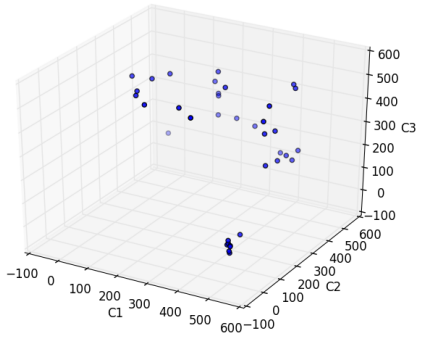

Figure 19: Simulation 6

6.3 Prediction Error Histogram

6.3.1 Knee-point Model

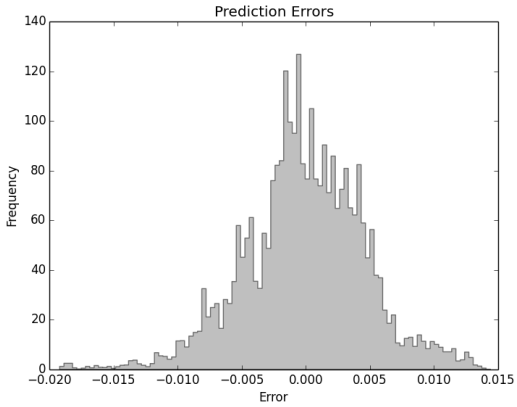

Figure 20: Simulation 1

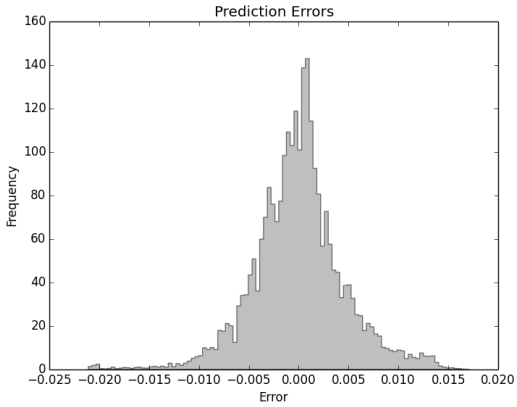

Figure 21: Simulation 2

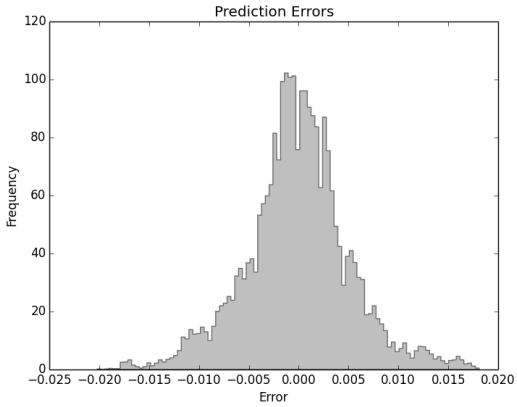

Figure 22: Simulation 3

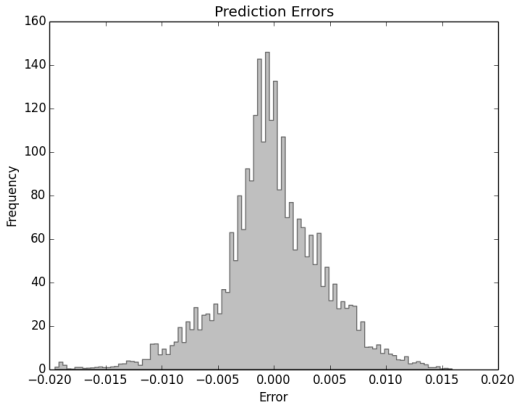

Figure 23: Simulation 4

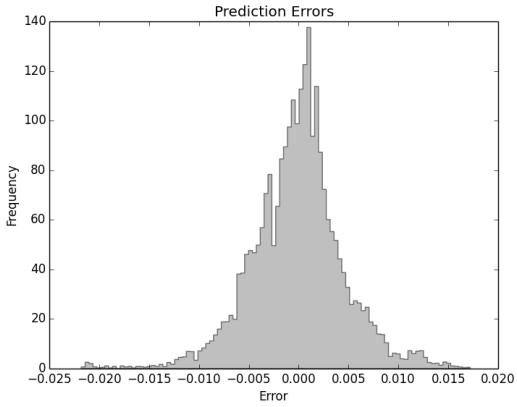

Figure 24: Simulation 5

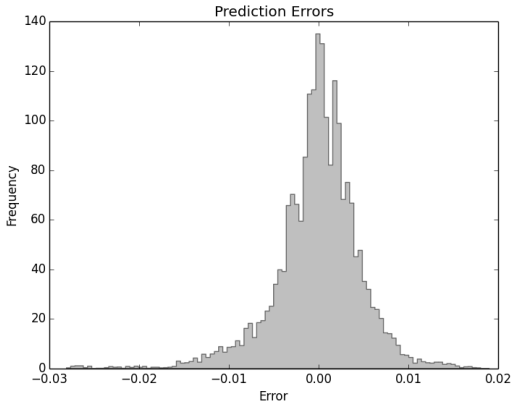

Figure 25: Simulation 6

6.3.2 Random Model

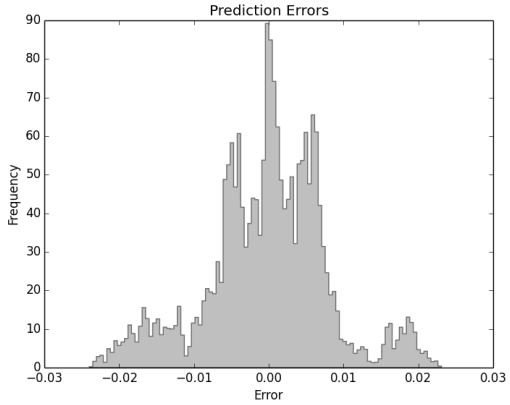

Figure 26: Simulation 1

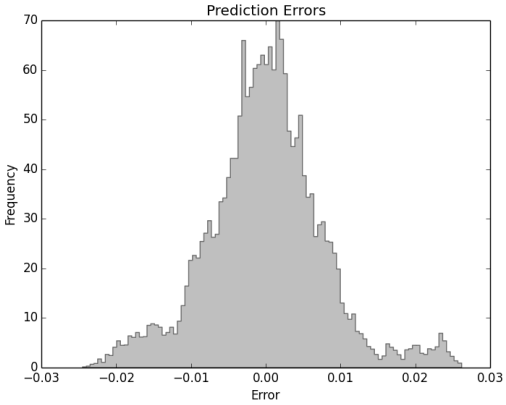

Figure 27: Simulation 2

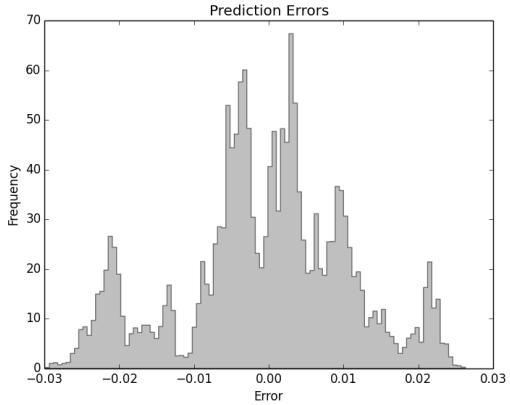

Figure 28: Simulation 3

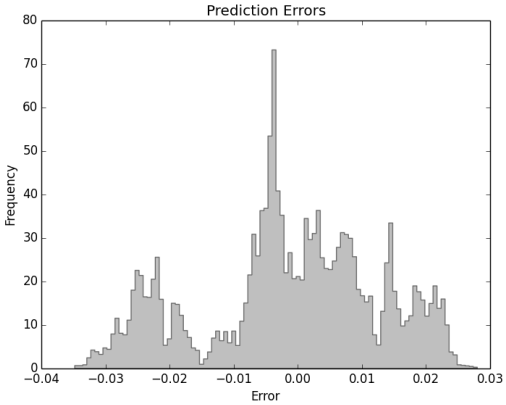

Figure 29: Simulation 4

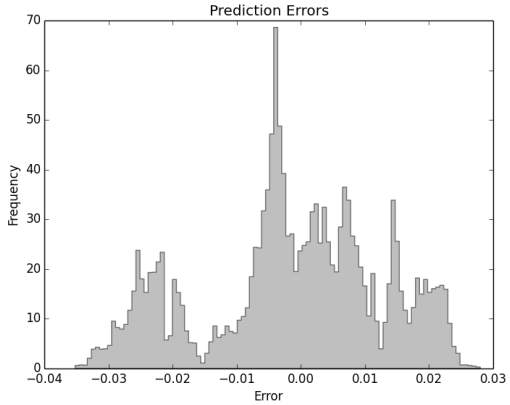

Figure 30: Simulation 5

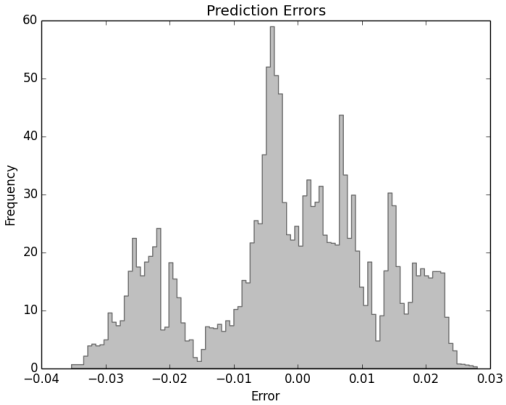

Figure 31: Simulation 6

References

[1] Aldrige, I. *High-Frequency Trading.* Wiley Trading, 2010.

[2] Berk, J., and DeMarzo, P. *Corporate Finance, third edition.* Pearson Prentice Hall, Boston, 2014.

[3] Dubil, R. *An Arbitrage Guide to Financial Markets.* Wiley Finance, 2004.

[4] Dunis, C., and Williams, M. Modelling and trading the eur/usd exchange rate: Do neural network models perform better?, 2002.

[5] Emmerich, M., and Deutz, A. *Multicriteria Optimization and Decision Making: Principles, Algorithms and Case Studies.* LIACS, Leiden University, 2006.

[6] Hull, J. *Options, Futures,and other Derivatives.* Prentice Hall, University of Toronto, 2012.

[7] Madsen, H., Bacher, P., and Juhl, R. Grey-box modeling; an approach to combined physical and statistical model building., 2015.

[8] Mandelbrot, B., and Hudson, R. *The (mis)Behavior of Markets.* Basic Books, 2004.

[9] Poli, R., Langdon, W., and McPhee, N. *A Field Guide to Genetic Programming.* http://lulu.com, http://www.gp-field-guide.org.uk/, 2008.

[10] Rijper, T., Sprenkeler, W., and Kip, S. High frequency trading position paper, 2010.

[11] Schwefel, H. *Evolution and Optimum Seeking.* Wiley NY, 1995.

[12] Sheta, A., Faris, H., and Alkasassbeh, M. A genetic programming model for s&p 500 stock market prediction, 2013.

[13] TransTrend. Trend following: riding the kurtosis, 2013.

[14] TSAY, R. *Analysis of Financial Time Series.* Wiley, University of Chicago, 2005.