Reims Management School
Master of Science in Finance and International Banking
Professional Thesis

Vincenzo RUSSO
russovincenzorocco@gmail.com

Supervisor: Sébastien LLEO. PhD.

ALGORITHMIC TRADING

Does using Firefly algorithm to find effective optimized trading rules produce greater results compared to a simple B&H strategy?

ACKNOWLEDGEMENTS

First of all, I would like to thank my family and my friends for the great support they offered me. Without their assistance, I wouldn't be able to write this thesis.

I also want to share my recognition to Professor Sébastien Lleo, which was my thesis supervisor, for his hints and guidance during the whole preparation of the thesis.

Then I want to thank Professor Christophe Bouteiller, Director of the Master of Science in Finance and International Banking program, for allowing me to do this master and where I learned a lot of things.

Finally, my expression of gratitude goes to Reims Management School for offering such a great program and allowed me to be part of it.

OUTLINE

« Does using Firefly algorithm to find effective optimized trading rules produce greater results compared to a simple B&H strategy? »

ABSTRACT

The objective of this document is to find out if the Firefly Algorithm (FA), a nature-inspired metaheuristic optimization algorithm, can be used to find effective optimized trading rules and compare the results to a simple Buy and Hold strategy. FA is one of the most recent evolutionary computing models and was studied by several researchers who concluded that it is an optimal and powerful algorithm to resolve complex problems. Despite the fact that some algorithms like Genetic Algorithm or Particle Swarm Optimization proved in the last decades to be optimal metaheuristics to find the fittest parameter for a trading rule, it was found out with limits and gaps such as the frequent transaction costs. This paper is to show if FA can be used to find optimal technical trading rules and see if it can resolve the limits of the other algorithms. We use the FA to find out effective optimized trading rules and use them firstly for the testing phase with daily prices of the S&P500 index during the last recession period and compare it with an upward period for our training phase to be able to prove the effectiveness of the algorithm. After that, we compare it with a benchmark used in most litteratures, the Buy and Hold (B&H) strategy. After transaction costs, FA produced higher results compared to the B&H strategy during the training period. However, during the out-of-sample tests, it showed contrary results. On one side, FA gained a simulated out-of-sample profit of over 4% for the chosen periods when daily transaction costs were taken into account. On the other side, FA could not outperform the benchmark when monthly transactions were taken into account. This is probably due to the selected and predefined trading system. However, globally FA demonstrated to be a more efficient metaheuristic algorithm than the PSO.

Keywords: Firefly algorithm, trading rules, Genetic algorithm, metaheuristic optimization, nature inspired algorithms, Particle swarm optimization, algorithmic trading

INTRODUCTION

Algorithmic trading has never been so prominent since the introduction of electronic computers and its evolution allowed better execution, lower transaction costs and a better decision making process. It has been in the last decade of great importance in the finance industry as it was demonstrated in some researches to produce abnormal returns compared to a human trader for example. The first automated trading systems embedded strategies like Trend following, Mean reversion or even Pairs trading. These are models preset by humans and have to be updated constantly. Later, artificial intelligence approaches made their appearances such as neural networks, Reinforcement Learning (RL) or even the Genetic Algorithm (GA), which is part of the most effective and self-learning algorithm and part of the evolutionary computation field. Because these are independent strategies they exclude psychological characteristics that can result in poor trading decisions, such as fear, greed or just imprudence [20].

The aim to create artificial intelligence algorithms goes back to the very initations of the computer age. Early scientists like Norbet Wiener or John von Neumann had visions like creating computer agents that reproduce a life-like system, that can learn and control its environments. To achieve this, they inspired themselves from the nature and used biology and psychology to meet their standards. Since this, models such as neural networks, machine learning approaches and the latest called evolutionary computation (EC), of which GA, Particle Swarm Optimization (PSO) and Firefly Algorithm (FA) are part of it, have surged and were used to predict for example the financial market.

Evolutionary algorithm was thought to be used early in the 1950s and 1960s as an optimization tool for engineering problems. It is a subset of EC and a generic-based metaheuristic optimization algorithm. Using operators inspired by biological evolution, such as mutuation, recombination, reproduction and selection, the idea was to develop a population of feasible solutions to a given problem [9]. After that, a cost function is used to determine the quality of these solutions. Then the population starts to evoluate by the constant application of the operators mentioned before.

Since this period, several new methods were created like the GA invented by John Holland , his colleagues and students at the University of Michigan in the 1970s. He didn't want to develop an algorithm to explain specific problems. He wanted to see and learn the adaptation effect as it happens in nature. Then, he tried to transfer the mechanisms of natural adaptation into computer systems. Several studies [5,9,10,12] showed the effectiveness of the GA into different fields. Franklin Allen and Risto Karjalainen (1993) for example demonstrated that GA could be used successfully to find technical trading rules [9]. Altough it was proven that the algorithm is a powerful method for optimization problems, other studies suggested that GA produced lower results compared to other algorithms due to the amount of computation time. Or even produced same results compared to a simple buy-and-hold strategy after transaction costs were taken into account.

Recently, the FA made its appearance, another and recent nature-inspired metaheuristic algorithm, and whose mechanics are inspired by the swarming or collaborative behavior of biological populations. It was invented by Xin-She Yang at the Cambridge University in 2007 and he inspired himself by studying the behaviour and motion of fireflies [7]. FA and GA are identical in the sense that these two evolutionary algorithms are population-based search methods. However, FA is part of the swarm intelligence technique. These kind of algorithms are based on the collective behavior of decentralized, self-organized systems in the search process. FA shows growing interest in its use in financial economics but so far there has been little formal analysis.

The purpose of this paper is to demonstrate how FA can be used to find effective optimized trading rules. Overall, my objective is to show that swarm algorithms are more robust than simple evolutionary algorithms like the GA to solve optimization problems. I chosed this topic because FA is a very promising metaheuristic algorithm and that very few studies exist. Preliminary studies [1,6,8] indicate that FA is superior over the GA and the Particle Swarm Optimization, another swarm algorithm. However, my motivation in this paper is to go further and illustrate that FA is very simple to implement, that it requires less functions and that it can find the best trading rules and produce greater results compared to the benchmark model, namely the Buy and Hold strategy.

Regarding technical trading rules, a lot of researches has been made to prove the effectiveness of these rules [21]. However, most of them found out that predefined rules do not produce any results. A great example would be with Alexander (1961) who tested

several trading rules and succeeded to overpass the buy-and-hold strategy. He advised a trader to buy if the price rises a fixed percentage (4% eg.) and sell if the price declines by the same percentage. However, after including transaction costs, he found out that his rules weren't profitable at all. Other researches came out with same results like Fama and Blume (1966) who find no evidence of profitable trading rules for Dow Jones stocks. Overall, studies during the 1960s provide no results by using preconfigured rules. In 1970, Fama concluded to reject technical analysis as a beneficial method and support the efficient market hypothesis.

The problem was that previous studies used predefined specifications of trading rules and lead to biased results because of possible data snooping. Recently, Allen and Karjalainen (1999) as mentioned earlier, succeed to find profitable trading rules using genetic algorithm for the Standar and Poor 500 Index. Nevertheless, their updated work concluded that by using in the out-of-sample test periods, the rules found with the GA weren't better than a simple buy-and-hold strategy. Therefore, in this paper, my motivation will be to show how FA can be used to find effective optimized trading rules and compare the results to a simple B&H strategy. In that way, by using an optimization algorithm, ad hoc specifications will be avoided.

This paper is organized as the following. Section 1 describes what algorithmic trading and metaheuristic algorithm mean and will introduce the Firefly algorithm. Section 2 will discuss about the chosen objective function, followed by the data preparation, the backtesting and to end with the application of FA. In Section 3, results are presented and discussed. Finally, this paper ends with the conclusion and possible future works.

1 LITERATURE REVIEW

1.1 Algorithmic trading

1.1.1 Definition and characteristics of AT

There are various definitions for Algorithmic trading (AT) that were used in various academic and general literatures. The most general one is from Prix et al 2007*"Computerized trading controlled by algorithms"* [22]. Another more detailed one from Chabaud et al. 2009 [23]:

> *"In algorithmic trading (AT), computers directly interface with trading platforms, placing orders without immediate human intervention. The computers observemarket data and possibly other information at very high frequency, and, based on abuilt-in algorithm, send back trading instructions, often within milliseconds. Avariety of algorithms are used: for example, some look for arbitrage opportunities,including small discrepancies in the exchange rates between three currencies; someseek optimal execution of large orders at the minimum cost; and some seek toimplement longer-term trading strategies in search of profits."*

To be clearer, what algorithmic trading does is that it takes transaction decisions in the financial markets by using predefined rules and guidelines. These pre-specified filters are called trading strategies. For instance, buy if the price rises a fixed pourcentage (4% eg.) and sell if the price decreases by the same pourcentage [25]. Other strategies are based by determining the size, the timing and the price of the orders. Many advantages of AT is that they are efficient, anonymous, lowers commissions, reduces market impact and timing risk.

Here below on table 1 some main characteristics of AT that we can find in most of todays academic literature:

Specific characteristics of AT
1 Pre-configured trading decisions
2 Monitoring market data in real-time
3 Automated order submission & management
4 No human intervention during the process
5 Main objective is to achieve a specific benchmark

Table 1: Characteristics of Algorithmic trading

1.1.2 Trading strategies: VWAP, Mean Reversion, Buy-and-Hold strategy

Many algorithms are very easy to use and to implement. Some basic algorithms focus solely on benchmarks based only on market generated data. One of them is the Volume Weighted Average price algorithm which main purpose is to match or beat the volume weighted averag price (their benchmark) over a specific period of time. Here a snapshot of the VWAP formula taken from Wikipedia:

$$P_{VWAP} = \frac{\sum_j P_j \cdot Q_j}{\sum_j Q_j}$$

where:

P_{VWAP} is Volume Weighted Average Price;

P_j is price of trade j;

Q_j is quantity of trade j;

j is each individual trade that takes place over the defined period of time, excluding cross trades and basket cross trades.

And its definition:

> *"In finance, volume-weighted average price (VWAP) is the ratio of the value traded to total volume traded over a particular time horizon (usually one day). It is a measure of the average price a stock traded at over the trading horizon."*

The purpose of using VWAP between professional traders is to use it as a passive benchmark for their daily executions. For this purpose, it is frequently used by pension

funds and some mutual funds. In addition, VWAP usage is to be sure that the trader who executes the order does in-line with the volume on the market.

Another simple trading strategy is the so-called Mean reversion. The main idea behind this strategy is that the high price and the low price of the stock are temporary and that the stock's price will move back towards the mean or average. In other words, the price will always return to the moving mean. The average can be the historical average of the price or return such as the average return of an industry or the growth in the economy. Here a simple explanation how the process works: When the present market price is higher than the average price then it is recommended to go short. On the other part, when it is below the mean, the stock is considered attractive to buy. Here below on Figure 1[24] a simple graph that shows how this concept works:

Figure 1: Example of a Mean Reversion strategy

Then, there is the Buy and Hold strategy, which is usually used for long-term investments. This strategy is frequently used as benchmark in studies for comparison purposes. It is the benchmark we are going to use to answer our research question. The main idea behind this strategy is that investors should buy a position and hold it the longest possible to be sure to make returns and that despite periods of volatility or declines. The reason why this strategy is usually used as a benchmark is because it's the only one of many strategies that has the lowest transaction costs. The costs incurred on all transactions are costs from brokerage

firms or from the bid/offer spread. Buy-and-hold strategy involves the fewest transactions for a given amount invested in the market. We have also seen that some investors suggesting to never sell the security unless you need the money. However, today we have in the academic world contractory studies about whether a buy-and-hold strategy is actually superior to an active investing strategy or not.

Over the years, people tried various technical trading rules for beating the market but without great success. Most of them did not produce any results. For example, Alexander [25] as mentioned before was one of the first to test different filter rules. He succeeded to create greater results compared to a simple buy-and-hold strategy. However after applying the transaction results, most of his filters weren't profitable anymore. The results of Fama and Blume [26] intensify the conclusions from Alexander. Both also found out that no profitable rules can be created. Since then until the early 1990s, the research academia concluded that technical analysis were worthless. The main problem of most of these studies was that the filters used ad hoc specifications of trading rules and this lead to biased results.

However more recently, new algorithms appeared to show some interests in the world of researchers and investors. These are the so called machine learning algorithms, like Reinforcement learning or Q-Learning which are advanced algorithms and prevent to be predefined specifications of trading rules and that could lead to biased results. Most of them are learning algorithms. Other are methods called metaheuristics optimization methods which we are going to discuss soon. In this study, we are going to use a recent metaheuristic algorithm to discover optimized trading rules.

1.1.3 Technical Indicators: MACD, TRB, MA

In this subsection we will introduce three simple and popular technical indicators used in most academic literatures and in our research. But before introducing them, we will explain what they mean and how they are used.

Technical indicators are useful tools for trading analysis and allows us to obtain trading rules. These trading rules study the past stock price (eg.close price) or volume over defined time periods to be able to forecast the trend of the future stock price. In fact, these rules generates buy/sell signals according to the prediction and later the investor creates profit by

going long or short by following an increasing or decreasing stock price trend. A trading rule can also develop null signals. In that situation, no positions are taken.

a) Moving Average Convergence Divergence (MACD)

Below a definition taken from Wikipedia :

> *« ...The MACD "oscillator" or "indicator" is a collection of three signals (or computed data-series), calculated from historical price data, most often the closing price. These three signal lines are: the MACD line, the signal line (or average line), and the difference (or divergence). ... The first line, called the "MACD line", equals the difference between a "fast" (short period) exponential moving average (EMA), and a "slow" (longer period) EMA. The MACD line is charted over time, along with an EMA of the MACD line, termed the "signal line" or "average line". The difference (or divergence) between the MACD line and the signal line is shown as a bar graph called the "histogram" time series… »*

This technical indicator is mostly used for trends purposes. The MACD indicator can be used in different ways. For instance, in up trends, the prefered choice is to put buy signals as the price is near the up trend line. When the MACD lines go downward, cross and turn to up ward, you will have a buy signal. Histogram can be used with the MACD lines. They are useful because it allows to forecast more precisely the trend. For instance, when the MACD lines are going down and the lines seem like they try to attempt a cross, having a look at the histogram bars allows to depict the real signal. In other words, when we have a case where the lines are willing to turn upward and the histogram bars are getting less negative and growing shorter, then you can put a buy signal. This is the so called crossing point.

On the other part, when the lines are willing to go down, having a look on the histograms allows you to grab the real sell signal.

Mathematically the 3 signals are defined as :

1. MACD = [stockPrices,12]EMA - [stockPrices,26]EMA
2. signal = [MACD,9]EMA

3. histogram = MACD – signal

where EMA stands for exponential moving average
And below an example of a MACD indicator :

$$shortema = 0.15 \times price + 0.85 * shortema_{[-1]}$$
$$longema = 0.075 \times price + 0.925 * longema_{[-1]}$$
$$MACD = shortema - longema$$

b) Trading Range Breakout (TRB)

The trading range breakout technical indicator can also be named Channel Breakout. The main idea behind this tool is that it calculates the highest and lowest close price of past n days. Below the formulas :

$$H_{t,n} = \max(p_{t-1}, p_{t-2}, ..., p_{t-n})$$
$$L_{t,n} = \min(p_{t-1}, p_{t-2}, ..., p_{t-n})$$

Where p is the close stock price on trading day t, and n the channel length.

Now suppose that the close price of trading day t is p_t. When $p_t > H_{t,n}$, the close price breakouts the channel and a buy signal is created, when $p_t < L_{t,n}$, a sell signal is created and when the price is inside both threeshold a null signal is created.

TRB can have in total 20 values as parameter setting [3]. Below the parameter value set for n (channel length) : 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 70, 75, 80, 85, 100, 125, 150, 175, 200, 250 (20 values);

c) Moving Average (MA)

The Moving Average indicator is one of the most used and famous tool when creating trading strategies. To get the right signal from this indicator, MA is splitted into 2 parts. The first is the long-period moving average of stock prices and the second is the short-period one. Over 2 moving windows of nl days for the long period and ns days for the short period, we have the following formulas:

$$Avg_{t,nl} = \frac{1}{nl} \sum_{i=t-nl+1}^{t} p_i$$

$$Avg_{t,ns} = \frac{1}{ns} \sum_{i=t-ns+1}^{t} p_i$$

where t is the current trading day and p_i is the close stock price on day i. In additon nl > ns and both averages $Avg_{t,nl}$ and $Avg_{t,ns}$ are recomputed and updated on each trading day.

The singals generated by the MA indicator is modest. For example, if the short-period moving average $Avg_{t,ns}$ is greater than the the long-period moving average $Avg_{t,nl}$, than MA created a buy signal. On the other hand if $Avg_{t,ns} < Avg_{t,nl}$, a buy signal is produced.

Concerning the parameters, for the nl (long-period moving average length) we can have in total 13 values [3]: 15, 20, 25, 30, 40, 50, 75, 100, 125, 150, 175, 200, 250; for the ns (short-period moving average length) a total of 16 values are counted : 1, 2, 5, 10, 15, 20, 25, 30, 40, 50, 75, 100, 125, 150, 175, 200.

Finally, as mentioned before, because ns is less than nl, the total number of rules generated by MA is 130.

1.2 Metaheuristic optimization algorithms

So what are metaheuristics algorithms?

Before explaining the meaning of this word, let's explain what heuristic means. Heuristic comes from the Greek "find" or "discover". In this context it means to find or discover by trial and error. The solutions can be found in a reasonable amount of time but there is no guarantee that optimal solutions are reached. Here is what Wikipedia defines heuristic:

> *"Heuristic refers to experience-based techniques for problem solving, learning, and discovery that gives a solution which is not guaranteed to be optimal. Where the exhaustive search is impractical, heuristic methods are used to speed up the process of*

finding a satisfactory solution via mental shortcuts to ease the cognitive load of making a decision."

Meta- means "beyond" or "higher level". Therefore, metaheuristic generally performs better than simple heuristics.

Metaheuristic algorithms are one of the advanced optimization methods out there to solve complex problems. The principal source of these algorithms is nature. Metaheuristic techniques try to reproduce elements from the nature and social behaviour. We mostly find biologically-inspired algorithms. These algorithms' main function is to select the fittest from the bio-systems which have evolved by natural selection over thousands of years.

Intensification and diversification are two important characteristics of the metaheuristic optimization methods. Intensification allows to search around the current best solutions and to choose the best candidates. On the other part, diversification, often done by randomization, enables the algorithm to explore more profitably the search space. Some other characteristics of these algorithms is the use of randomization and local search. Randomization provides a way to move away from local search.

We can find today various bio-inspired optimization algorithms like the most popular one, the Genetic Algorithm which we will present in the next subchapter. Then we have others like the particle swarm optimization and recently the Firefly Algorithm.

1.2.1 What are optimization algorithms ?

In the context of financial marlkets, optimization algorithms are tools to tune or train automated trading systems. They are usually applied during the development of a trading system, usually in the end, and can also be used during a real time session.

The main idea behind the optimization process it to find the the fittest, also called optimal or effective values for selected parameters (for example averaging period for Moving Average indicator) that gives the highest profit. When developing automated trading systems, we will have several parameters. If we have a goal in our mind, for instance, producing the maximum profit out of the trading system, then we want to find out which parameters ables

us to achieve this goal. This is where the optimization tools come in. We gave them the parameters to be optimized and the cost function, also called objective function which is in simple english our goal, that is to say, producing the maximum profit. Then you decide what the minimum and maximum is allowed for the parameters and in what increments the values should be updated. Then, the optimization tool performs multiple back tests using all possible combinations of the parameters values. Later, when this process is finished, we will get our best possible values for the parameters that will give the best results for our cost function.

Here below on figure 2 [19] we can see a representation of the creation process of a trading system with the use of optimisation tools in the middle of the development.

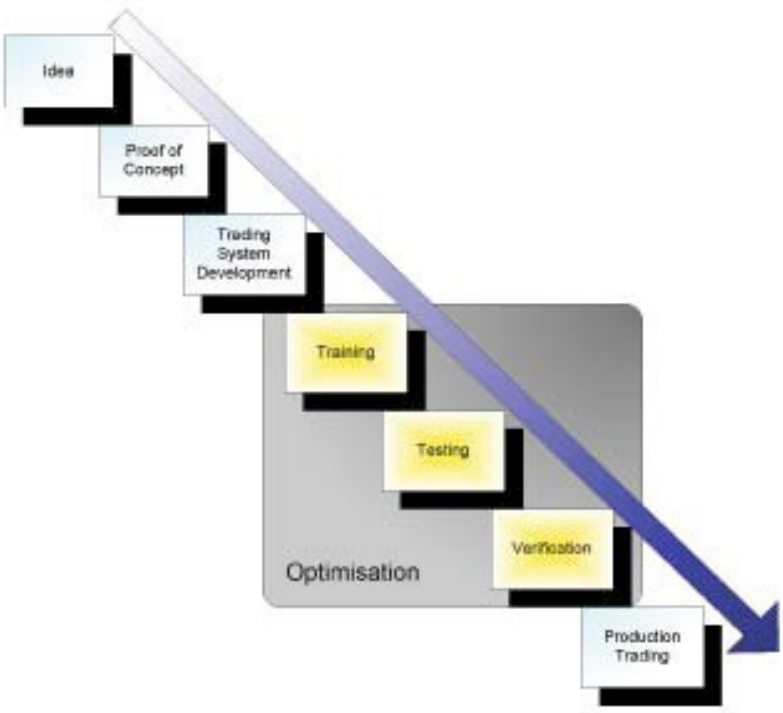

Figure 2: Use of optimisation tools during a trading system creation

Between the time the system is developed and the time it goes in production, the optimisation process aims to find out what produces the best results for the system. This is how we are going to process in this paper.

One important thing is that we have to express our goal to the optimiser. Below are steps that represents a general procedure of an optimizer [19]:

a) First thing to do is to generate a random set of possible solutions (called a population)
b) Secondly, the optimizer evaluates each candidates by trading it over some time/data window (named an environment)
c) After that, the optimizer will assign fitness values to each individuals
d) The optimizer then combines the most 'fit' solutions to make a new and 'better' solution(called offspring)
e) When the optimizer does not find any best candidates any more, it will stop and take the current best one as the optimal solution. When it hits the end, we say it has *converged*.

When we use powerful search techniques to find the fittest parameters on some training data called ‘curve fitting’, we need to proof the effectiveness of these optimized parameters. What we are looking for is not just to curve fit the parameters to a particular window of data, but find out parameters that are better on many other windows of data. The goal by doing this is to find effective and strong parameters. The convention most people follow and should follow is "Train, Test, and Verify".

The idea behind the "Train, Test, and Verify" [19] process is that after training our population to be effective in an environment we gave them – the window of data – we need to test our population to different windows. Therefore, at the end of the process, we will have a robust and effective trading system that has been trained and tested on different datas. Finally, the verification process is simply the step where the system is used with real-time data but by doing fake trading.

Metaheuristic algorithms can be divided into two categories which we are going to present below. The first one is Evolutionary algorithm and the other one Swarm intelligence. Both are subsets of metaheuristics.

1.2.2 Evolutionary computation: Genetic Algorithm

Evolutionary computation which is part of the advances in modern machine learning led us to analyze data more efficiently. Moreover, it gives us the ability to understand possible underlying patterns available in the financial market. Before explaining how it works, here below on figure 3 [13] a graph which represents the process of such algorithms:

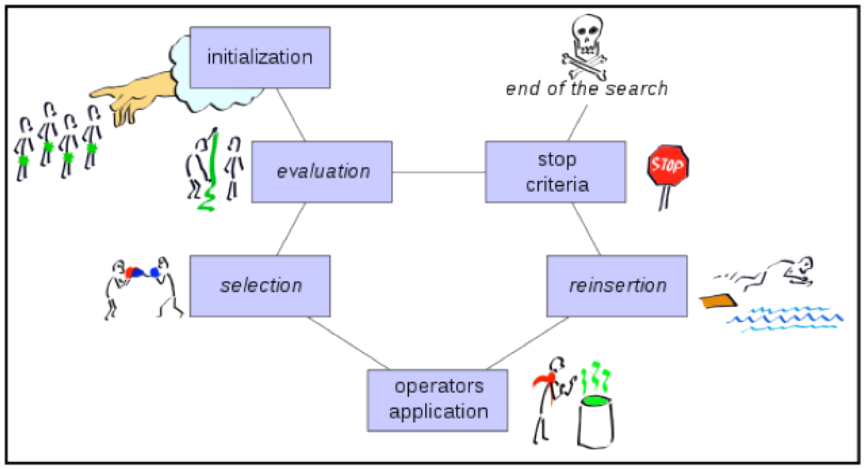

Figure 3 : General schema of an Evolutionary Algorithm (EA)

As showed on the schema, the basic idea of an evolutionary algorithm is to maintain a group of solution candidates at the first stage. Later, this population is evaluated to obtain the quality of each solution candidate regarding to a problem-specific fitness function. In fact, fitness function defines the environment for the evolution of the candidates. In phase 3, we get thanks to this fitness function fit members. After this, these new solution candidates are recombined through various operators such as mutuation or duplication. Finally, candidates where operators were used, are reinserted to be able to get the best solution(candidate) possible.

Evolutionary algorithm can be often applied when the size of the search space causes difficulties comparing to traditional optimization methods. But they do have some limits. For instance, keeping a population of genetic structures increases the execution time. This is mainly because of the number of times the objective function is used and evaluated. In addition, they are unlikely to produce good results and are less efficient than algorithm that are purposely designed for specific domains! This is because evolutionary algorithms have limited problem knowledge. Anyway, this compelling exploitation of new computation

methods will help financial organizations to take effective decisions, which will further improve their competitive edge. Approaches like Neural Networks (NN) or Genetic Algorithms (GA) have been widely used to forecast the financial market [27].

Let's take the example of the Genetic Algorithm. John Holland and his colleagues and students at the University of Michigan introduced the Genetic Algorithm in the mid 1970s [9,10,12]. Principles of genetics and evolution were used as inspiration. GA follows the model of reproduction behaviour observed in biological societies. In addition, it follows the principles of "survival of the fittest" in the process of searching to select and create individuals (design solutions) that are adapted to their environment (design objectives/constraints). Over a number and repeated generations, optimal candidates with desirable characteristics will evolve and remain in the genome composition of the population compared to weaker ones which have unwanted qualities.

Here below on figure 4 [28] a diagram wich shows the operation of a simple genetic Algorithm:

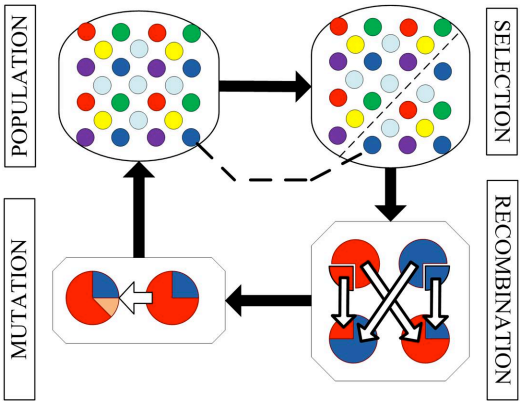

Figure 4: Operation of GA

We can see in this diagram that in each iteration (or 'generation'), a group of possible solutions is evaluated and the best ranking solutions are selected as 'parents' of the next generation. After that, characteristics from their parents are taken for the design of the next generation solutions, including casual random variations or 'mutations'. The previous operation is repeated to be able to increase the chances to find the 'fitness' of successive

populations. Hence, we will be sure to increase the chances of finding the optimal solution from the chosen population.

To be more explicit in the process of the algorithm: We initially start with a population where randomly generated candidates can be found. The following generation of optimal candidates is by taking characteristics from promising candidates. The previous step which involves the recombination mechanism is by taking two fit candidates randomly from the population and set them as parents. After that, the parents are recombined through a "Crossover" operator. The operator splits the genetic characteristics of the parents apart at randomly chosen locations and joins a slice from each parent to create a new generation. Then, the GA evaluates the fitness of the new generation and replaces members of the population that are unfit. This is continuously done until a best criterion is found. Finally, we are left with a population which provides solution candidates that can be used for the original problem.

GA is frequently used to solve difficult optimization problems because it can handle both continuous and discrete variables, and nonlinear constrain and objective functions without requiring gradient information. The effectiveness of GA was proved by several academic papers for different kind of fields [5,9,12]. Franklin Allen and Risto Karjalainen (1993) for example [9] demonstrated that GA could be used successfully to find technical trading rules. Altough it was proven that the algorithm is a powerful method for optimization problems, other studies suggested that GA produced lower results compared to other algorithms due to the amount of computation time. Or even produced same results compared to a simple buy-and-hold strategy after transaction costs were taken into account [9].

1.2.3 Swarm intelligence: Particle Swarm Optimization

Metaheuristic's other subset is the so-called Swarm intelligence (SI) algorithms. The speciality about these algorithms is that they mimick biological agents such as fish, birds, humans and others. Particle swarm optimization for instance was developed by inspiring from the swarming behavior of fish and birds.

PSO was developed by Kennedy and Eberhat in the 1990 by trying to reproduce the motion of swarms of birds for a sociocognitive study which the main subject was to study and investigate the ‘’collective intelligence ‘’ in biological groups [14]. PSO is quite similar to GA in the sense that their search method are population-based. GA and its many tuned versions have been successful in the last decades due to the easy implementation, the ability to solve nonlinear optimization problems and its intuitiveness. However, the drawback of GA is that it uses heavy computer ressources that causes costs. The story is different for PSO which uses less ressources.

PSO is a population search method and works by mimicking the swarming and collaborative behavior of biological populations. Similar to GA, PSO move from a initial population to a new combined one trough a single generation pass (iteration). The recombined group is done by probabilistic and deterministic rules. Here below on Figure 5 [29] an explicit presentation of the process :

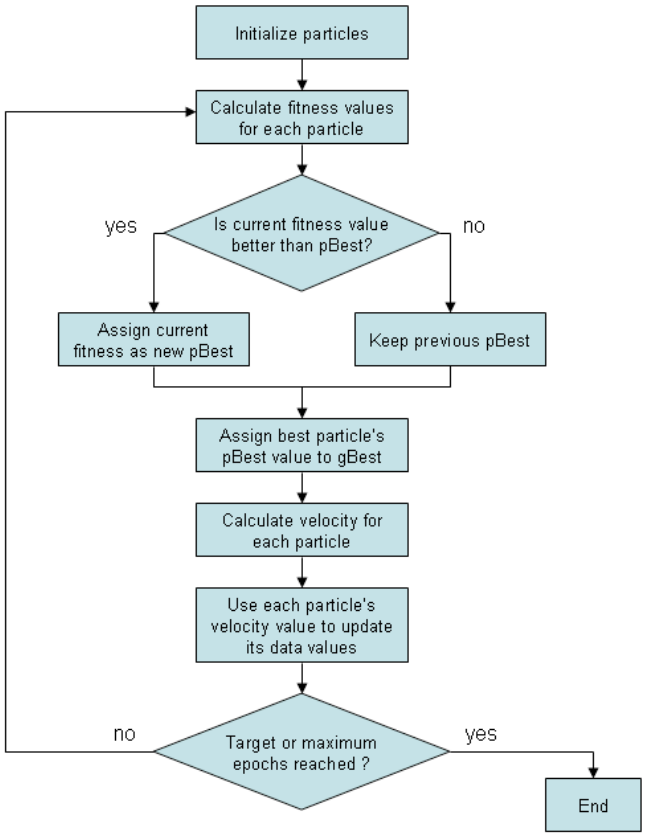

Figure 5: Workflow example of the Particle Swarm optimization.

As seen on the figure, we initially start with a random generated solutions/particles (our initial population). Each particle represents a candidate solution. Every particle knows its own position, its own direciton and velocity, the position of its own best solution, the position of the best currently known solution(pBest) of the whole swarm. They are then evaluated against the pBest particle. If the current evaluated particle is not better thant pBest, pBest doesn't change. However if it is better than pBest, we will get a new pBest. Then trough repeated iterations (generations) with included informations about the best optimal solution(pBest), the particles travels inside this space and share the space's informations(eg.presence of a predator) together until to finally arrive to the optimal solution if criterions are met [3].

In the last 20 years we have seen new algorithms appearing in the market such as particle swarm optimization, differential evolution, bat algorithm and firefly algorithm. All of them have shown great potential in solving complex engineering problems for example [34]. One of them particularly has shown to be effective in dealing with global optimization problems [30, 31, 32, 33]. It is the so called Firefly Algorithm.

In the next subchapter, we will show the essentials of FA, review its latest developments, the concept behind the algorithm, explain the algorithm and finally highlight the reasons why FA is a strong optimization method.

1.3 Firefly algorithm: a recent swarm intelligence algorithm

1.3.1 History, definition and concept of FA

Firefly algorithm is defined as a metaheuristic algorithm inspired by the nature. It was created and studied by Xin-She Yang in 2007-2008 at Cambridge University. The algorithm follows the flashing patterns and behavior of fireflies. Here below a photo of a firefly [15]:

Image 1 : Firefly(source : Google Image)

The particularity about these insects is their ability to produce light which we will discuss later. Similar to the PSO, fireflies have a collaborative approach which enables them to share the environment's information to other fireflies of the group [1,6,7,8]. This intelligence puts them in the category of swarm algorithms. The way they comunicate, search for pray and find mates is by using bioluminescence with different flashing frequences. Patterns were observed trough the different rhythms of the flashes, the rate of flashing and the amount of time for which the flashes are observed.

For our study, the FA will follow these three simple rules [7]:

1. All the fireflies are unisex. That means that one firefly is attracted to other fireflies regardless of their sex.

2. Attractiveness is proportional to the brightness of the firefly. If we have two flashing fireflies, the one with the least brigthness will move towards the brighter one. Both attractiveness and brightness decrease when the distance between the fireflies increases. If there is no one firefly brighter than another one, the individual will move randomly in the space. This is the so called *random walk*[1].

3. The brightness can be defined in a similar way as the fitness function in genetic algorithms. For an optimal decision problem, brightness can simply be proportional to the value of the objective function.

[1] A random walk is the process by which randomly-moving objects wander away from where they started. (source : http://www.mit.edu/~kardar/teaching/projects/chemotaxis(AndreaSchmidt)/random.htm)

1.3.2 Light Intensity, Attractiveness, Distance and Movement

Let's explain now the two main points/functions of the FA. The first one is the variation of light intensity and the second one the formulation of the attractiveness. In the next paragraphs and lines, we will assume that the attractiveness of a firefly is determined by its brightness which in turn is associated with the encoded objective function.

First of all, we know that according to the inverse square law [18], the intensity of light I decreases as the distance r will increase. In other words, light intensity as well as attractivennes decreases with the distance from its source. This assumption will give us the following definition :

$$I(r) = \frac{I_s}{r^2} \tag{1}$$

Where I_s is the intensity at the source and r the distance.

We also know that according to the Bouger-Lambert-beer Absorption Law [17], the intensity I of light decreases exponentially with the distance d when the light enters in an absorbing medium. In this study, air is the medium which absorbs light. This gives us the following definition :

$$I = Ioe^{-\gamma r} \tag{2}$$

Where γ is both an absorption coefficient and a constant, and Io the initial light intensity.

To avoid the singularity at r = 0 in the expression (1), we combine these two effects, the inverse square law and the absorption law, which results in an approximate Gaussian form formula :

$$I = Ioe^{-\gamma r^2} \tag{3}$$

We said before in the 3 rules that attractiveness is proportional to the light intensity. This implies that we can define the attractiveness β of a firefly in the (3) Gaussian formula :

$$\beta = \beta o e^{-\gamma r^2} \tag{4}$$

Where βo is the attractiveness at distance r = 0

Exponential function can use a lot of computer ressources. Therefore, if necessary, we can replace it by $1/(1 + r^2)$ which gives us an approximation as :

$$\beta = \frac{\beta o}{(1+\gamma r^2)} \tag{5}$$

Equation (4) ables us to define the characteristic distance $\Gamma = 1/\gamma$ over which the attractiveness changes significantly from β_O to $\beta_O e^{-1}$. This results to the possibility to determine γ.

The area between two fireflies i and j at x_i and x_j defines the Cartesian distance.

$$r_{ij} = \left\| x_i - x_j \right\| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{6}$$

After having all the needed parameters, we can now define the movement function. We know that the movement of a firefly i is attracted to another more attractive (brighter) firefly j. The movement itself consists of two elements in the searching space: Exploitation and exploration. Exploitation occurs when a firefly approaches the better local solutions and exploration occurs during random steps. This will give us the following movement equation that is used in the FA and that represents a Firefly x_i attracted to a firefly x_j:

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2} \left(x_j^t - x_i^t \right) + \alpha_t \varepsilon_i^t \tag{7}$$

The second term is the attraction. The third term is randomization with α being the randomization paramter and ε_i is a vector of random numbers being drawn from a Gaussian distribution or uniform distribution. For instance, ε_i can be changed by (rand − 1⁄2) where rand is a number generated randomly and uniformly distributed in [0, 1].

We can have two special cases from equation 7. When γ is zero, the attractiveness and brightness are constant. This means that a firefly can be seen by all other fireflies. In

addtion, this means there is no air and therefore no absorption effect. This reduces to a variant of particle swarm optimisation. On the other hand, when γ tends to the infinity, attractiveness and so brightness decreases enourmously. Fireflies will not be able to see eachother. It is like having a foggy air. This implies that all fireflies will move randomly and the function becomes a simple random walk (random search technique).

1.3.3 Pseudo Code

Here below the pseudo code [1] which summarizes the FA with the three rules written above.

Begin :

- Initialize algorithm parameters:
- MaxGen: the maximal number of generations
- γ: the light absorption coefficient
- r: the particular distance from the light sourced: the domain space
- Define the objective function of f(x), where x=(x1,.....,xd)T
- Generate the initial population of fireflies or xi (i=1, 2 ,..., n)
- Determine the light intensity of Ii at xi via f(xi)

Function :

While (t<MaxGen)

For i = 1 to n (all n fireflies);

For j=1 to n (n fireflies)

If (Ij> Ii),

move firefly i towards j by using equation (7);

end if

Attractiveness varies with distance r via Exp [$-\gamma r^2$];

Evaluate new solutions and update light intensity;

End for j;

End for i;

Rank the fireflies and find the current best;

End while;

Post process results and visualization;

End procedure

1.3.4 Main steps

Now lets summarize the steps/process [35,36] on how the algorithm is used in optimization tools:

Step 1 (Initialization):

The very first thing the FA does is to initialize the location of S fireflies in T dimensional search space restricted by the search boundary. It is formulated as :

$$x_{st}(0) = rand_{st}(0,1)(x_{st}^{U} - x_{st}^{L}) + x_{st}^{L}$$
$$s = 1,2,3,...S; \qquad t = 1,2,3,...T$$

where x_{st}^{U} and x_{st}^{L} are the upper and lower limits of the *t*th variable in the population. On the other part, $rand_{st}(0,1)$ is a uniformly distributed random value inside the range [0, 1].

Step 2 (Compute the brightness of firefly):

The second step is to determine the light intensity (or brightness) of each firefly at the current generation by the objective function at their present place. We know that the light intensity is directly proportional to the objective function of an individual firefly for a maximization problem situation. For a minimization problem, light intensity is inversely proportional to the cost function.

$$I(x) \propto f(x)$$

$$I(x) \propto 1/f(x)$$

Step 3 (obtain the current global best and rank the fireflies):

In step 3 fireflies are ranked according to their brightness in the current generation. The one with the best brightness becomes the global best (gBEST) and its position will be shared to the other fireflies.

Step 4 (Update the fireflies's location trough the movement function) :

In this step, each fireflies's position moves to the firefly with a greater light intensity. Except for the global best gBEST which position doest not change for the current generation. We know that that the attraction of a firefly is evaluated by the light intensity of this one.

Atraction between fireflies p and k for instance in an D dimensional search space is given by the already previous formulated formula(equation 7) :

$$x_p = x_p + \beta_0 e^{-\gamma r^2} pk(x_p - x_k) + \alpha\varepsilon_p$$

where ε_p is a parameter that produces random numbers obtained from Gaussian distribution. Then we have α which is a randomization parameter, γ is the light absorption coefficient for a given atmosphere. Finally, the attraction between the fireflies is seen by the product of β_0 and $e^{-\gamma r^2}\ pk$.

Step 5 (Repeat until it has converged) :

This is the final step. The algorithm will repeat the steps from 2 to 4 until all generations are completed. At the end, we will have the best firefly gBEST for the global solution and with the selected firefly it will be able to give the most effective fitness value of the cost function.

Below on figure 6, a simple flowchart that summarizes the algorithm process:

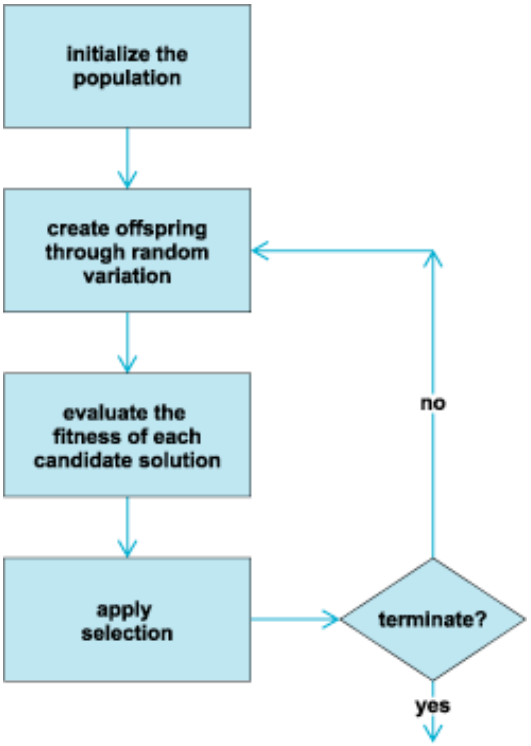

Figure 6: Workflow of the Firefly Algorithm (source: Google Image)

1.3.5 Why is FA so efficient ?

To answer this question, let us analyze the algorithm itself. FA is based on the swarm-intelligence. That points to the conclusion that it has similar advantages than other swarmintelligence based algorithms. Actually, when the absorption coefficient is equal to zero $\gamma = 0$, we obtain a PSO variant called Accelerated PSO [7].

However, FA has one main advantage compared to the other algorithms. It is its ability to subdivise automatically and to deal with multimodality. First, we know that FA's attractiveness decreases with the increasing of distance. This will subdivide the whole population automatically into subgroups. Each group can swarm around each mode or local optimum. The best global solution can be found among all these modes. Secondly, if the population size is enough higher than the number of modes, the subdivision will give the fireflies the ability to find all optima simultaneously. We know that mathematically the average distance of a group of fireflies that can be seen by adjacent groups is controlled by $1/\gamma$. As a result, an entire population can split up into subgroups with a given, average

distance. This ability to automatically split up makes it adequate for extremely nonlinear, multimodal optimisation problems.

We than have other advantages that were discovered from preliminary studies like Horng et al or Banati and Bajaj [7], where FA produced a better performance in terms of time and execution compared to other algorithms. Another advantage discovered is that FA outperforms algorithms like the artificial bee colony (Basu and Mahanti) or PSO(Zaman and Matin) when it is the case to find the best global solution for an optimization problem[7]. This implies that FA can efficiently solve highly nonlinear problems. Additionally, FA can also solve scheduling and travelling salesman problem in a promising way [1]. Then we have Senthilnath el al.'s [37] study which compared FA with 11 different algorithms. FA outperformed all other algorithms and the authors of the study concluded that FA is an efficient method for classifications and clustering.

1.4 Thesis motivation

There are several reasons regarding my motivation to study the FA and use it to optimal trading rules. The first one is that the preliminary studies all of them showed great results concerning FA to find optimal solutions for different problems[1,6,8,30,31,32,33,37]. The second reason is that most studied trading strategies were biased. However, recently, Allen and Karjalainen (1999), succeed to find profitable trading rules using genetic algorithm for the Standar and Poor 500 Index [9]. This proves that evolutionary algorithms can be an optimal way to find optimal trading rules. Nevertheless, their updated work concluded that by using in the out-of-sample test periods, the rules found with the GA weren't better than a simple buy-and-hold strategy. Therefore, in this paper, my motivation will be first of all to avoid ad hoc specifications; to find optimal effective trading rules and finally to show that it can produce better results than the B&H strategy. To sum up, my thesis work will be to use the Firefly algorithm to find effective optimized trading rules which will produce excessive returns compared to the benchmark.

2 DISCUSSION OF METHODOLOGY

2.1 The chosen trading system

In this section I will describe which trading system I used in conjunction with the Firefly algorithm.

We can find several studies about the use of specific trading strategies and where technical indicators were optimized. For instance in one of them [4,38], weights were associated with the technical indicators. Then they used the Genetic Algorithm to optimize the weights and find the best combination that maximized the objective function. In their study they selected Moving Average, Price Channel Breakout, Price Trend and Order Book Volume imbalance as technical indicators. From the optimized combined weight of the indicators, a decision was taken. As objective functions they used Sharpe ratio and sortino ratio.

The trading system used in our research will also be based on optimized set of weights with selected indicators. In that way, we will be able to compare the performance between PSO and FA. However, our research is based on a single objective function approach. In our tests, we will only use the Excessive Return as our fitness function. The Excessive return is simply the difference between the summed return from the whole period and the B&H return. Therefore, we can easily compare our returns to the selected benchmark, the B&H strategy.

Concerning the technical indicators we have decided to choose the Moving Average (MAV), the Moving Average Convergence Divergence (MACD), the Trading Range Breakout(TRB) and the Price Trend (PT). These indicators were evaluated over the two training periods and the two testing periods.

We decided to go after stock indices and chose the S&P 500 for our research as they are more stable and are not risky to possible price speculations. We decided to test our trading system on two different periods. The first one during the 2007-2009 crisis and the second one during an upward trend which will be from the beginning of 2012 to September 2013.

The trading system works as follow: Each indicator generates a signal. We will express this

signal as Si. If the indicator produces a Buy or Long position, Si will have the value of 1. On the other hand, if the indicator is in a short position, Si will have the value of -1. After that, a weight wi was attached for each technical indicator. Below on table 2 an example produced on excel which shows the different signals the system can trigger.

TESTING DATA
(monthly)

Period	Date	Open	High	Low	Close	Average	MAV	MAV Buy/Sell Rule	Signal
1	2007-01	1,418.03	1,441.61	1,403.97	1,438.24	1,425.46			
2	2007-02	1,437.90	1,461.57	1,389.42	1,406.82	1,423.93			
3	2007-03	1,406.80	1,438.89	1,363.98	1,420.86	1,407.63			
4	2007-04	1,420.83	1,498.02	1,416.37	1,482.37	1,454.40			
5	2007-05	1,482.37	1,535.56	1,476.70	1,530.62	1,506.31			
6	2007-06	1,530.62	1,540.56	1,484.18	1,503.35	1,514.68			
7	2007-07	1,504.66	1,555.90	1,454.25	1,455.27	1,492.52	1,455.40	BUY	1
8	2007-08	1,455.18	1,503.89	1,370.60	1,473.99	1,450.92	1,466.58	BUY	1
9	2007-09	1,473.96	1,538.74	1,439.29	1,526.75	1,494.69	1,471.08	SELL	-1
10	2007-10	1,527.29	1,576.09	1,489.56	1,549.38	1,535.58	1,485.58	BUY	1
11	2007-11	1,545.79	1,545.79	1,406.10	1,481.14	1,494.71	1,499.12	BUY	1
12	2007-12	1,479.63	1,523.57	1,435.65	1,468.36	1,476.80	1,497.18	SELL	-1
13	2008-01	1,467.97	1,471.77	1,270.05	1,378.55	1,397.09	1,490.87	SELL	-1
14	2008-02	1,378.60	1,396.02	1,316.75	1,330.63	1,355.50	1,474.96	SELL	-1

Table 2: Example of the use of the MAV technical indicator and the resulting Signal

For instance for the MAV 6 months technical indicator, we got a BUY rule if the Average Price t-1 is greater than the Moving Average 6 months price. On table 2, we can see for period 7, that the Average price of period 6 is greater than MAV price of period 7. As a result, this indicates us a trend to move in the market and will have the BUY rule and Si to 1.

Previously was just an example of one single technical indicator. If we aggregate each signals of each technical indicators with the corresponding weights, we will have our final trading decision. For each periods of the selected window(eg: TESTING DATA) for example, the final trading decision of our trading system depends on the value of $\sum S_i w_i$. A trade is created if this value is bigger than 0.5. On the other hand, if the value is less than 0.5, the trade is closed. Below on table 3 a random example of a trading decision for a specific period:

w	S	w*S
0.2	1	0.2
0.1	-1	-0.1
0.2	1	0.2
0.4	1	0.4
0.1	-1	-0.1
1	***Total***	**0.6**

Table 3: Example of a final trading decision

In this example, the decision is to execute the trade because $\sum S_i w_i = 0.6$ and it is greater than 0.5.

2.2 The objective function

Let's define now our fitness function [4]:

Given

- [MAV, MACD, TRB, PT] = [1,2,3,4]
- Si signal vector associated with the Indicator i
- Where Si*j* = 1 if position = Buy at *jth* trading day

  Si*j* = -1 if position = Sell at *jth* trading day
- w = (w1 , w2 . . . w4) , w i $\in$ R

  weights associated with the 4 indicators
- WDTR(w) the weighted decision trading rule defined by:

  Buy Dj > $0.5 \sum w_i$

  No action Dj = $0.5 \sum w_i$

  Sell Dj<$0.5 \sum w_i$

  where

  Dj= $\sum S_i w_i$

  Maximize y = f(w) = ExcessReturn(WDTR($\vec{w}$))

Our objective function will be the excess return's formula: $\mathbf{E = r - r_{bh}}$ which was used in the study from Dome Lopetch & David Corne [5] where their research was related by finding technical trading rules with Genetic Programming and comparing it against a Buy

and Hold strategy. The first term r is simply the return of an investment of $1,000. The second term r_{bh} is the return achieved with a buy and hold strategy.

The first part of our fitness function is calculated following the formula:

$$r_{bh} = \sum r_t + \ln(\frac{1-c}{1+c})$$

The first term r_t of r calculates the return on an investment when the trader is in the market. The term is constructed as $r_t = \log P_t - \log P_{t-1}$ where P_t is simply the price at time t. Then we have $I_B(t)$ which can have 1 or 0 as value. 1 suggests a Buy signal at time t and 0 if it is a Sell signal. The second term is simply the risk-free return. In the study [5], the component r_f is taken from the published US Treasury bill (data available from http://research.stlouisfed.org/fred/data/irates/tb3ms). Then comes $I_S(t)$ which is quite similar to $I_B(t)$ but produces the opposite effect.

To finalize, the third term of the return formula is for the transaction costs. In our paper we will set 0.25% as cost for each transactions made. For example for a transaction of share of $1,000, it will cost me $2.50. The parameter n is for the number of transactions made during the whole perdiod.

The second part of the objective function, r_{BH}, is obtained by:

$$r_{bh} = \sum r_t + \ln(\frac{1-c}{1+c})$$

where r_t is simply the return described above, in other words the return on an investment when the investor is in the market.

When we summarize our objective function and implement it in a programming language like VBA, which is the language used in excel and the one we are going to use for our research, then we obtain:

```
Function fitness(Arr() As Double) As Double
    Dim excess_return, component1, component2, retTB, costs, return_inv As Double
    Dim signal As Double
    Dim k, periodindex, r, decision As Integer

    retTB = 1.73

    Sheets("Calculation").Activate

    component1 = 0
    component2 = 0
    costs = 0
    return_inv = 0
    excess_return = 0

    'First term
    j = 0
    For i = 1 To 36
        periodindex = i + 5
        j = 0
        signal = 0
        While j < D
            Cells(i + 45, j + 9).Value = Arr(j)
            signal = signal + Cells(periodindex, j + 9) * Arr(j)
            j = j + 1
        Wend

        Cells(i + 45, 14).Value = signal

        If signal > 0.5 Then
            decision = 1
        Else
            decision = 0
        End If

        Cells(i + 45, 15).Value = decision

        'r = (1 - 0) * Rnd 'EKAF
        'component1 = component1 + r * Cells(periodindex, 21) 'r by signal
        component1 = component1 + decision * Cells(periodindex, 21)

        Cells(i + 45, 16).Value = decision * Cells(periodindex, 21)

        If decision = 0 Then
            component2 = component2 + retTB
            Cells(i + 45, 17).Value = retTB
        End If

    Next
```

2.3 Type of Research and population choice

For our research, I decided to use two different periods. As shown on table 4, the training period will focus during a turmoil market, namely the financial crisis 2007-2009. One of the reasons to chose this period is to see if the trading strategy can perform well with optimized parameters. Secondly, this period is typically an out-of-market period. As a result, less Buy transactions and therefore smaller cost transactions. This will probably affect the results of our research when comparing to the B&H strategy.

Table 4: Training and testing periods

Frequency	Training Range	Testing Range
Daily	03/01/2007 to 31/12/2009	03/01/2012 to 06/09/2013
Monthly	03/01/2007 to 31/12/2009	03/01/2012 to 06/09/2013

We will firstly investigate on daily returns. After this, we will use monthly returns to see if changing frequency and therefore transactions frequency could trigger a better trading, a better return and a higher return compared to a B&H strategy.

Secondly, after training the trading system on the turmoil market, the second phase will be to test it on a different time period. This step is to show if the trading system is robust. In addition, as mentioned earlier in this paper, it is part of the optimization process. Training, Testing and Validation are the 3 main steps before putting a trading strategy in the market.

For the population we decided to go after the Standard and Poors 500 (S & P 500) stock index as it is the most used one and used as a reference in several academic studies [5,9]. In addition, as mentioned earlier in my paper, stock indices are more stable and less risky to potential price speculations.

We can see the S&P 500 as a global portfolio that contains 500 stocks. When we aggregate all of them, it will give us for instance the daily, the opening or the closing price of S&P500. In other words, the sum of all the stocks of S&P500 opening prices is the value of that particular day.

2.4 Data gathering and preparation

2.4.1 Data Processing

First of all, our main goal is to backtest our trading strategy. Back-testing is simply the process of optimizing a trading strategy using historical data. After that, the process is to see if the trading system has the ability to predict the market.

To be able to test our trading system, we need historical price datas. As said before, we will use the S&P 500 stock index and mainly focus on two different periods.

The data for our research was prepared as followed:

The historical data was taken from Yahoo finance. Below on table 5 a snapshot of the file we recover when exporting the data from yahoo and importing it to excel:

	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Volume	Adj Close
2	31/12/2009	1126.6	1127.64	1114.81	1115.1	2076990000	1115.1
3	30/12/2009	1125.53	1126.42	1121.94	1126.42	2277300000	1126.42
4	29/12/2009	1128.55	1130.38	1126.08	1126.2	2491020000	1126.2
5	28/12/2009	1127.53	1130.38	1123.51	1127.78	2716400000	1127.78
6	24/12/2009	1121.08	1126.48	1121.08	1126.48	1267710000	1126.48
7	23/12/2009	1118.84	1121.58	1116	1120.59	3166870000	1120.59
8	22/12/2009	1114.51	1120.27	1114.51	1118.02	3641130000	1118.02

Table 5: Historical data from Yahoo finance

On the first row we have the description of the columns. Then on each rows we have the date, the open, the high, the low, the close and the adjusted close price. The difference between adjusted close and close price is that adjusted close covers the dividends and splits.

We have in total 756 observation points for the period 2007-2009 and 442 for the second period.

As we want to test our trading system with different frequencies, we transformed the data into monthly ones. As a result, we will have 36 monthly datas for the test period and 21 for the training period. On each row, the opening price will be the first day of the month and the

closing price will be the last day of the month. In addition, the lowest and highest price are the prices reached during the month.

We also used data before the selected period to be able to calculate daily, monthly returns and the 4 chosen technical indicators for the first days or months.

2.4.2 Analysis & statistics

In this subsection we will talk about quantitative methods we used to analyse the data and make statistics out of it.

What we did in the first time is to take the historical datas, transformed it into monthly datas and made a discrete statistic analysis. Discrete statistics is simply a way to analyze on a limited size of data.

Below on Figure 7 a snapshot of the quantitative methods used for the discrete statistic analysis for the recession period.

	A	B	C	D	E	F
1	**TRAINING DATA (monthly)**					
2	**Date**	**Index price**	**Monthly Return**		**Statistics**	
3	31/01/07	1438,24			Number of months	36
4	28/02/07	1406,82	-2,18%		Largest monthly return	9,39%
5	30/03/07	1420,86	1,00%		Smallest monthly return	-16,94%
6	30/04/07	1482,37	4,33%		Average monthly return	-0,70%
7	31/05/07	1530,62	3,25%		Average by using continuous compounded returns	-0,55%
8	29/06/07	1503,35	-1,78%		Month of maximum return	30/04/09
9	31/07/07	1455,27	-3,20%		Month of minimum return	31/10/08
10	31/08/07	1473,99	1,29%		Variance of monthly returns	0,0033
11	28/09/07	1526,75	3,58%		Standard deviation of monthly returns	5,74%

Figure 7: Statistics on monthly training data

As shown on Figure 7, the number of months for the training data, namely the recession period, after transforming the data into monthly values, the number of observations is 36. The largest monthly return occurred during April 2009 and S&P 500 produced 9,39% of monthly return. This shows us that during the 2 years of recession, the Stock index only recovers at the end of the selected period. On the other part, when we look which month produced the worst return, our statistics told us that October 2008 was the worst one. This month created a negative return of -16,94%. Overall, during the recession period, the S&P 500 produced a negative average monthly return of nearly -1%. This suggests us that most of the time, returns were small and often negative.

When taking now the testing period, namely the supposedly upward trend period, we have different results. Below on Figure 8 a snapshot of the testing data:

	A	B	C	D	E	F
9						
10	TESTING DATA (monthly)					
11	Date	Index price	Monthly Return		Statistics	
12	31/01/12	1312,41			Number of months	21
13	29/02/12	1365,68	4,06%		Largest monthly return	5,04%
14	30/03/12	1408,47	3,13%		Smallest monthly return	-6,27%
15	30/04/12	1397,91	-0,75%		Average monthly return	1,11%
16	31/05/12	1310,33	-6,27%		Average by using continuous compounded returns	1,21%
17	29/06/12	1362,16	3,96%		Month of maximum return	31/01/13
18	31/07/12	1379,32	1,26%		Month of minimum return	31/05/12
19	31/08/12	1406,58	1,98%		Variance of monthly returns	0,0008
20	28/09/12	1440,67	2,42%		Standard deviation of monthly returns	2,78%
21	31/10/12	1412,16	-1,98%			

Figure 8: Statistics on monthly testing data

On Figure 8 we can see that we have in total 21 monthly observations. The month that produced the largest result is January 2013 with a return of approximately 5%. The smallest return can be seen on May 2012 with a value of -6,27%. It is more than half of the smallest monthly return of the training data. The average monthly return for the testing period is more than 1%. This proves that this period was more confident than the recession period. In addition, we can see that most of the time, S&P 500 index' return was often positive.

In a second time, we took our imported historical and daily datas and made a continuous statistic analysis. Continuous statistics is simply a way to analyze on continuous data on a continuously way.

Below on Figure 9 a snapshot of the quantitative methods used for our continuous statistics analysis on the recession datas:

	A	B	C	D	E	F
1	TRAINING DATA (daily)					
2	Date	Index price	Daily Return		Statistics	
3	03/01/07	1416,6			Number of days	756
4	04/01/07	1418,34	0,12%		Largest daily return	11,58%
5	05/01/07	1409,71	-0,61%		Smallest daily return	-9,03%
6	08/01/07	1412,84	0,22%		Average daily return	0,03%
7	09/01/07	1412,11	-0,05%		Average by using continuous compounded returns	-0,01%
8	10/01/07	1414,85	0,19%		Day of maximum return	13/10/08
9	11/01/07	1423,82	0,63%		Day of minimum return	15/10/08
10	12/01/07	1430,73	0,49%		Variance of daily returns	0,0001
11	16/01/07	1431,9	0,08%		Standard deviation of daily returns	1,16%
12	17/01/07	1430,62	-0,09%		Average annual return	-3,51%
13	18/01/07	1426,37	-0,30%		Variance of annual returns	0,0021
14	19/01/07	1430,5	0,29%		Standard deviation of annual returns	4,62%
15	22/01/07	1422,95	-0,53%			

Figure 9: Continuous statistics on the training data

We have in total 756 observations. The largest daily return happened the 13th October 2008 with a return of nearly 12%. When we compare this statistic to the discrete one, we have interesting results. We metioned earlier that the month with the largest return on the discrete statistics anlysis happened in April 2009. This clearly shows us that October 2008 probably was a special month. In fact, when we want to figure out which day had the smallest return, October 15th of 2008 comes out. In addition, we also know after doing our analysis on monthly datas that the month we the smallest return was October 2008. We can conclude, that this month was the most volatile one and the one that impacted the most on the S&P 500.

After this obvious conclusion, we went further and calculated the average daily return in two different ways. The first one is by taking the very first daily return divided by the last data return. Then we took the number of observations as the exponent. Below the formula used on excel for the calculation of the average daily return:

$$=(B3/B758)^{(1/F3)}-1$$

Where B3 is the first daily price, B758 the last one and F3 the number of observations.

The second way is by using compounded daily returns. By calculating in these two different ways we got distinct results. However, they are not far away eachother as both gives us a daily average return of nearly zero. This cleary demonstrate that this period wasn't profitable at all. This can also be showed by computing the average annual return which gives us a result of nearly -4%.

When analysing the testing period, we can sum up different results. Below on Figure 10 a snapshot of the testing data:

H	I	J	K	L	M
TESTING DATA (daily)					
Date	Index price	Daily Return		Statistics	
03/01/12	1277,06			Number of days	422
04/01/12	1277,3	0,02%		Largest daily return	2,54%
05/01/12	1281,06	0,29%		Smallest daily return	-2,50%
06/01/12	1277,81	-0,25%		Average daily return	-0,06%
09/01/12	1280,7	0,23%		Average by using continuous compounded returns	0,06%
10/01/12	1292,08	0,89%		Day of maximum return	02/01/13
11/01/12	1292,48	0,03%		Day of minimum return	20/06/13
12/01/12	1295,5	0,23%		Variance of daily returns	0,0001
13/01/12	1289,09	-0,49%		Standard deviation of daily returns	0,77%
17/01/12	1293,67	0,36%		Average annual return	16,27%
18/01/12	1308,04	1,11%		Variance of annual returns	0,0009
19/01/12	1314,5	0,49%		Standard deviation of annual returns	3,06%
20/01/12	1315,38	0,07%			

Figure 10: Continuous statistics on the testing data

In total we have 422 daily returns. First of all, the largest daily return occurred in January 2nd 2013 with a result of 2,54%. In the discrete analysis, the largest monthly return was January 2013. This shows us that the year 2013 began positively for the S&P 500 index. Concerning the day with the smallest return, the data points to the 20th of June 2013 with a value of -2.50%. May 2013 was the month with the smallest return as demonstrated previously on the discrete anaysis. This shows that May and June of year 2013 was an unproductive period. When looking at the average continuous compounded daily returns, the testing data produced better results compared to the training period. Despite the small downward trend on May and June 2013, the average annual return is largely above the one from the recession period. With a value of more than 16%, the selected period clearly is an upward trend.

2.4.3 Backtesting with the Buy and Hold strategy

After having used quantitative methods to produce statistics out of the historical datas, we will apply the Buy and Hold strategy on both periods. Therefore, we can use these results later in the third Chapter of this paper.

The Buy and Hold strategy is expressed in the following way:

$$\text{Return/Cost*100}$$

Where Return is simply the sale price minus the cost price.

When we apply the formula during the training period, we receive the value of -23% which is a huge loss for the concerned investor. This shows us that this was a turnoil period and that the investor should have kept his position longer.

When the Buy-and-Hold strategy is used in the testing period, we get the result of approximately 26%. This is quite a big gain compared to the training datas.

With these two opposite results we can compare in the third chapter with the results of the optimized trading strategy.

2.5 Application of the FA to find optimized parameters

In this subchapter we will show how the Firefly Algorithm was used to find the most effective optimzed parameters for trading system.

First of all, we converted the FA Matlab code of Xin-She Yang (appendix A) to a VBA code to be able to use it with Excel. The complete code can be found on the appendices.

Before running the code, we personalized it by including the input parameters and by writing the fitness function, namely the Excess return. Concerning the input parameters, the dimension of the search space will be set to 4 as we have in total 4 parameters to optimze, namely the weights. The rest of the FA configuration can be seen on table 6 :

Parameter	Value 1	min	max	recommended
Population Size/Solution Number/N°fireflies	15	15	100	25 to 40
Iteration Number/Generations/N° of pseudo time steps	50			
Objective function evaluations	750			
Initial value for randomization parameter α	0,2	0	1	
Attractiveness at r = 0 βo	1	0	1	
Light Absorption coefficient γ gamma	1	0,01	10	
Dimension	4			

Table 6: FA input parameters

The value we are going to use in our paper is taken from column “Value 1”. We select these values as in most academic papers studying the FA, the authors claimed to get positive results [1,6]. In additon, Xin-She Yang in his main research paper [7] advised to select values between the minimums and maximums range for each input parameters.

For the population size we have 15 fireflies. Each firefly will have different weights values. The population changes its position during 50 generations. This will make us in total 750 objective function evaluations. Concerning the attractiveness, it will be set to the maximum 1 and the light absorption to the very low level, namely 1. To finish, the randomization parameter is set to 0,2. These values were used for the training and testing periods.

To be able to run the FA VBA code, we took the trading rules of each technical indicators. The trading rules for the training period can be seen on Figure 11 for example.

TRAINING DATA (monthly)						
			Rules			
Period	Date	en	MAV	MACD	TRB	PT
1	2007-01		1	1	1	1
2	2007-02		1	1	1	1
3	2007-03		-1	1	1	1
4	2007-04		1	1	1	1
5	2007-05		1	1	1	1
6	2007-06		-1	1	-1	1
7	2007-07		-1	1	1	1
8	2007-08		-1	1	-1	1
9	2007-09		-1	1	-1	1
10	2007-10		-1	1	1	1
11	2007-11		-1	1	1	1
12	2007-12		1	1	1	1
13	2008-01		-1	1	1	1
14	2008-02		-1	1	1	1
15	2008-03		-1	1	-1	1

Figure 11: Trading rules for training period(monthly)

Those values were obtained by applying the formulas of each technical indicators. We did the same for the daily frequencies and the testing period.

We finally ran the code. The code took nearly 20 seconds to find the best optimized parameters. It was ran on a MacBook Pro 13' 2.3Ghz Intel Core i5.

Below on figure 12 can be found the output of the code for the training monthly period :

GENERATION	FIREFLY	WEIGHT 1	WEIGHT 2	WEIGHT 3	WEIGHT 4	Objective Function
50	1	0,1627118	0,3398116	0,3535223	0,1439544	-301,7202173
50	2	0,1107222	0,3289067	0,3268458	0,2335253	-301,7202173
50	3	0,0445617	0,2887270	0,3729105	0,2938008	-301,7202173
50	4	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	5	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	6	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	7	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	8	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	9	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	10	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	11	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	12	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	13	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	14	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173
50	15	0,1189853	0,3050275	0,4187927	0,2228009	-301,7202173

Figure 12: Output of the FA VBA code

The snapshot shows the final rows the code generates to find the best optimal solution. We can conclude from this figure that our objective function was found before the end of the generations. The optimization process needed less than 50 generations to get the best combined weights. The above figure is the execution of the code to minimize the objective function. In other words, it's to find the weights that produce the worst return.

3 RESULTS + DATA ANALYSIS AND INTERPRETATION

3.1 Results discussion

Below on table 7 a summary of the experimental results from the training period :

Training (Monthly)	**Best**	**Worst**	**Mean**
w1	0,3491	0,1627	0,2559
w2	0,0223	0,3398	0,1811
w3	0,2039	0,3535	0,2787
w4	0,4247	0,1440	0,2843
Return	342,62	-624,86	-141,12
B&H	-323,14	-323,14	-323,14
Objective Function	665,76	-301,72	182,02

Table 7: Results of FA for the training period (monthly)

Table 7 clearly shows that FA was able to outperform the simple Buy And Hold Strategy. The algorithm found that with the following weights, w1 = 0,3491, w2 = 0,0223, w3 = 0,2039, w4 = 0,4247, it produced a return of 342,62$ which is 665,76$ (106,3%) more than the B&H. Concerning the worst scenario, we have a negative return of -624,86$ which is 301,72 more than the B&H.

Below on table 8 a summary of the training period but for daily frequencies :

Training (Daily)	**Best**	**Worst**	**Mean**
w1	0,3130	0,0297	0,1713
w2	0,3400	0,5012	0,4206
w3	0,1699	0,1921	0,1810
w4	0,1772	0,2770	0,2271
Return	73,54	-892,77	-409,61
B&H	-301,50	-301,50	-301,50
Objective Function	375,04	-591,27	-108,11

Table 8: Results of FA for the training period (daily)

The results above show that by doing daily transactions our costs will be higher and as a result our final return will be low. For the training period, our optimized trading system still outperforms the B&H with 375,04$ (24,39%) regardless the heavy costs. The best weights w1 = 0,3130 , w2 = 0,3400, w3 = 0,1699, w4 = 0,1772 produced a return of 73,54$ which is

smaller than monthly transactions. In fact, the total return contains the costs of 756 transactions which in total is about 550$.

Our optimized trading system clearly outperforms the B&H. To demonstrate the robustness of the system, we use it during an upward trend period. Below we can see the results we got during the testing period for monthly transactions:

Testing (Monthly)	**Best**	**Worst**	**Mean**
w1	0,3129	0,1627	0,2378
w2	0,3400	0,3398	0,3399
w3	0,1698	0,3535	0,2617
w4	0,1771	0,1440	0,1605
Return	217,6291	6,4290	112,0291
B&H	342,7600	342,7600	342,7600
Objective Function	-125,1309	-336,3310	-230,7309

Table 9: Results of FA for the testing period (monthly)

Table 9 shows that with the best weights, namely w1 = 0,3129 , w2 = 0,3400, w3 = 0,1698, w4 = 0,1771, the trading system created a return of 217,62$ during the upward trend. However, the results show that after transaction costs taken into account, the optimized trading strategy could not beat the simple B&H strategy. In fact, the B&H strategy performed 63% better than the trading system. The worst weights still creates a positive return. However, it the return is very small and 336,33$ less than the B&H strategy.

When we take into account daily transactions during the testing period, we got the following results showed on Table 10:

Testing (Daily)	**Best**	**Worst**	**Mean**
w1	0,3277	0,0297	0,1787
w2	0,0059	0,5012	0,2535
w3	0,3217	0,1921	0,2569
w4	0,3445	0,2770	0,3108
Return	393,5745	171,7645	282,6695
B&H	378,1100	378,1100	378,1100
Objective Function	15,4645	-206,3455	-95,4405

Table 10: Results of FA for the testing period (daily)

With the best weights w1 = 0,3277 , w2 = 0,0059, w3 = 0,3217, w4 = 0,3445, the trading system produced a return of 393,57$ and outperformed slightly the B&H strategy, even after taken into account the transactions costs which were of an amount of 237$ for a total of 422

transactions. The worst scenario sill produced a positive return but its performance was poorer than the B&H.

CONCLUSION AND FUTURE WORK

In this paper, I tried to answer the question if the recent nature-inspired metaheuristic optimization algorithm can be used to find effective optimized trading rules and if it can outperform the results of a simple Buy and Hold strategy.

The paper successfully demonstrates the effectiveness of the Firefly algorithm to optimize a trading system. By chosen 15 fireflies, 50 generations, 0.2 as the randomization parameter, 1 for both the attractiveness and the light absorption, the algorithm performed the simple B&H strategy in 3 out of the 4 tests. Our trading strategy outperformed the Buy and Hold strategy during the monthly training period. In addition, it could outperform when daily transaction costs were taken into account. Concerning the testing period, the optimized system failed to beat the benchmark during monthly transactions. However, it surprisingly outperformed when daily transaction costs were taken into account.

As a result, we can conclude that our trading system is effective on both periods. Another statement we can add is that changing the transaction frequency will not necessarily improve the trading system. This was proved by the training period results with monthly transactions. Another advantage of the FA is its speed. It just took few seconds to obtain the cost functions for the diverse periods.

These results clearly show the potential behind the studied optimization tool as utility during the creation of stock trading systems. In addition, the positive results globally performed better than the PSO algorithm used in different studies. In the future, we would like to use several optimization algorithms with the same trading system and compare their performances. I also would like to test different technical indicators and compare the performance with the one from this paper. In addition, I would like to use other objective functions or going for a multiple objective approach to test the full potential of the Firefly Algorithm. Finally, using different input parameters for the FA configuration can be an interesting future work.

BIBLIOGRAPHIC REFERENCES

[1] APHIRAK Khadwilard. « Application of Firefly Algorithm and Its Parameter Setting for Job Shop Scheduling » .The Journal of Industrial Technology, Vol. 8, No. 1, January-April 2012

[2] Dr. BRABAZON Anthony, Dr. Michael O'Neill. « Biologically Inspired Algorithms for Financial Modelling ». Ireland, Publisher : Springer, 2006, 276p.

[3] FEI Wang. « Complex stock trading strategy based on parallel particle swarm optimization ». 73p. Thesis : Philosophy : University of Hong Kong : August 2012

[4] BRIZA Antonio C., NAVAL Prospero C. « Design of Stock Trading System for Historical Market Data Using Multiobjective Particle Swarm Optimization of Technical Indicators ». 8p. Thesis : Computer Science : University of the Philippines : 2008

[5] LOHPETCH Dome, CORNE David. « Discovering Effective Technical Trading Rules with Genetic Programming : Towards Robustly Outperforming Buy-and-Hold » 6p. Heriot-Watt University Edinburgh

[6] KWIECIEN J., FILIPOWICZ B. « Firefly algorithm in optimization of queuing systems ». 6p. Bulleting of the polish academy of sciences technical sciences, Vol. 60, No. 2, 2012

[7] YANG Xin-She, HE Xingshi. « Firefly Algorithm : Recent Advances and Applications ». Int. J. Swarm Intelligence, Vol. 1, No. 1, 36-50p, 2013

[8] BACANIN Nebojsa, PELEVIC Branislav, TUBA Milan. « Portfolio Optimization Problem by the Firefly Algorithm ». 6p Thesis : Computer Science : University Belgrade

[9] ALLEN Franklin, KARJALAINEN Risto. « Using genetic algorithms to find technical trading rules ». Journal of Financial Economics No. 51, 245-271p, 1999

[10] MITCHELL Melanie. « An introduction to Genetic Algorithms ». First MIT Press paperback Edition, 1998

[11] RAGHAVENDRA Srinivas, PARASCHIV Daniel, VASILIU Laurentiu. « A Framework for Testing Algorithmic Trading Strategies ». Working Paper No. 139, December 2008

[12] Genetic Algorithm, http://en.wikipedia.org/wiki/Genetic_algorithm

[13] Evolutionary Algorithm, http://en.wikipedia.org/wiki/Evolutionary_algorithm

[14] Particle Swarm Optimization, http://en.wikipedia.org/wiki/Particle_swarm_optimization

[15] Firefly Algorithm, http://en.wikipedia.org/wiki/Firefly_algorithm

[16] Heuristic, http://en.wikipedia.org/wiki/Heuristic

[17] GERWARD Leif. « The Bouger-Lambert-Beer Absorption Law » : Technical University of Denmark, http://www.canberra.edu.au/irps/archives/vol21no1/blbalaw.html

[18] Gravity and the Inverse Square Law, http://www.physics.uc.edu/~hanson/ASTRO/LECTURENOTES/F03/Gravity/Page1.html

[19] Strategies: Optimisation Algorithms for Automated Trading, http://www.automatedtrader.net/articles/strategies/33/strategies-optimisation-algorithms-for-automated-trading

[20] ELDER Thomas. « Creating Algorithmic Traders with Hierarchical Reinforcement Learning ». University of Edinburgh : 2008

[21] Technical Analysis, http://en.wikipedia.org/wiki/Technical_analysis

[22] PRIX Johannes. « Algorithmic Trading Patterns in Xetra Orders », 2007

[23] Chabaud, Alain, Benjamin, Chiquoine, Hjalmarsson, Erik. « Rise of the Machines : Algorithmic Trading in the Foreign Exchange Market », 2009

[24] ROCHE Collen. « Corporate profits and mean reversion ». http://pragcap.com/corporate-profits-and-mean-reversion, 2012

[25] ALEXANDER, S. « Price Movements in Speculative Markets : Trends or Random Walk », Industrial Management Review, No. 2, 1961, 7-26p

[26] FAMA, E & BLUME, M. (1966) « Filter Rules and Stock Market Trading », Journal of Business, No. 40, 226-241

[27] CHAUHAN Anil, « Automated Stock Trading and Portfolio Optimization Using XCS Trader and Technical Analysis ». 100p. Thesis : Artificial intelligence : University of Edinburgh : 2008

[28] Optimisation of Collector Form and Response, http://www.engineering.lancs.ac.uk/lureg/group_research/wave_energy_research/Collector_Shape_Design.php

[29] Particle Swarm Optimization Workflow, http://mnemstudio.org/particle-swarm-introduction.htm

[30] Saibal K. Pal, C.S Rai, Amrit Pal Singh, « Comparative Study of Firefly Algorithm and Particle Swarm Optimization for Noisy Non-Linear Optimization Problems ». Published Online September 2012 in MECS (http://www.mecs-press.org/)

[31] YANG Xin-She, « Nature-Inspired Metaheuristic Algorithms », Luniver Press, London, 2008.

[32] YANG Xin-She, « Firefly algorithms for multimodal optimization ». Stochastic Algorithms : Foundations and Applications, SAGA, Lecture Notes in Computer Sciences 5792, 169-178p, 2009

[33] LUKASIK S., ZAK S. « Firefly algorithm for continuous constrained optimization task ». Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems. LNCS 5796, 97-106, 2009

[34] GANDOMI Amir H., YANG Xin-She, ALAVI Amir H. « Mixed variable structural optimization using Firefly Algorithm », Computers and Structures No. 89, 2325-2336p, 2011

[35] GREWAL Narwant S., RATTAN Munish, PATTERH Manjeet S. « A linear antenna array failure correciton using firefly algorithm », Progress in electromagnetics Research M, Vol. 27

[36] Kamaldeep Kaur, Dr. Vijay Kumar Banga « Optimization of linear antenna array using firefly algorithm », International Journal of Engineering Research & Technology (IJERT), Vol. 2 Issue 6, June 2013

[37] Sentilnath J., S.N. Omkar, V.Mani, « Clustering using firefly algorithm : Performance study » Swarm and Evolutionary Computation No. 1, 2011, 164-171p, Published by Elsevier

[38] H. Subramanian, S. Ramamoorthy, P. Stone, and B.J. Kuipers. « Designing safe, profitable automated stock trading agents using evolutionary algorithms ». In GECCO ’06 : Proceedings of the 8th annual conference on Genetic and evolutionary computation, 1777-1784p, New York, USA, 2006

APPENDICES

A) FireFly VBA Algorithm

```
Public Const MAX_FFA As Integer = 100 'Maximum of fireflies
Public Const MAX_D As Integer = 20 'Maxium of dimensions

'VARIABLE DECLARATIONS
Public NumEval As Integer 'number of evaluations
Public Index(MAX_FFA) As Integer     'sort of fireflies according to fitness values

Public ffa(MAX_FFA, MAX_D) As Double ' firefly agents
Public ffa_tmp(MAX_FFA, MAX_D) As Double 'intermediate population
Public f(MAX_FFA) As Double 'fitness Values
Public Iy(MAX_FFA) As Double 'light intensity
Public nbest(MAX_FFA) As Double 'the best solution found so far
Public lb(MAX_D) As Double 'upper bound
Public ub(MAX_D) As Double 'lower bound

Public n As Integer ' number of fireflies
Public D As Integer 'dimension of the problem
Public MaxGeneration As Integer 'number of iterations
Public alpha As Double 'alpha parameter
Public betamin As Double 'beta parameter
Public gamma As Double 'gamma parameter

Public fbest As Double  'the best objective function


Sub Main()
   'INSERT VALUES INSIDE THE VARIABLES
   n = Sheets("Setup").Range("B2").Value
   D = Sheets("Setup").Range("B8").Value
   MaxGeneration = Sheets("Setup").Range("B3").Value
   alpha = Sheets("Setup").Range("B5").Value
   betamin = Sheets("Setup").Range("B6").Value
   gamma = Sheets("Setup").Range("B7").Value

   Sheets("Output").Range("A:G").ClearContents
   Cells(1, 1).Value = "GENERATION"
   Cells(1, 2).Value = "FIREFLY"
   Cells(1, 3).Value = "WEIGHT 1"
   Cells(1, 4).Value = "WEIGHT 2"
   Cells(1, 5).Value = "WEIGHT 3"
   Cells(1, 6).Value = "WEIGHT 4"
   Cells(1, 7).Value = "Objective Function"

   'OPTIMIZER
```

```
Rnd (1)
Call init_ffa

Dim t, i As Integer 'generation  counter
Dim selFF(3) As Double
t = 1
'i = 0
While t <= MaxGeneration
   'evaluate new solutions
   i = 0
   While i < n
      selFF(0) = ffa(i, 0)
      selFF(1) = ffa(i, 1)
      selFF(2) = ffa(i, 2)
      selFF(3) = ffa(i, 3)
      f(i) = fitness(selFF) 'obtain fitness of solution
      Iy(i) = f(i) 'initialize attractiveness_
      i = i + 1
   Wend

   i = 0
   Sheets("Output").Activate
   While i < n
      Cells(i + 2, 1).Value = t
      Cells(i + 2, 2).Value = i + 1

      Cells(i + 2, 3).Value = ffa(i, 0)
      Cells(i + 2, 4).Value = ffa(i, 1)
      Cells(i + 2, 5).Value = ffa(i, 2)
      Cells(i + 2, 6).Value = ffa(i, 3)

      Cells(i + 2, 7).Value = f(i)
      i = i + 1
   Wend

   'ranking fireflies by their light intensity
   Call sort_ffa

   'replace old population
   Call replace_ffa

    'find the current best
    i = 0
    While i < D
      nbest(i) = ffa(0, i)
      i = i + 1
    Wend
    fbest = Iy(0)
```

```
    'move all fireflies to the better locations
    move_ffa


    t = t + 1
  Wend

  Sheets("Output").Activate
  Cells(2, 8).Value = fbest

End Sub

Sub init_ffa()
  Dim i, j As Integer
  Dim r, tabr(MAX_D), totalr As Double

  'initialize upper and lower bounds
  i = 0
  While i < D
    lb(i) = 0
    ub(i) = 1
    i = i + 1
  Wend

  i = 0

  totalr = 0

  While i < n
    j = 0
    totalr = 0
    While j < D
      r = (1 - 0) * Rnd
      tabr(j) = r
      totalr = totalr + r
      j = j + 1
    Wend

    j = 0
    While j < D
      ffa(i, j) = (tabr(j) / totalr) * (ub(j) - lb(j)) + lb(j)
      j = j + 1
    Wend

    f(i) = 1 'initialize attractiveness
    Iy(i) = f(i)

    i = i + 1
  Wend
```

```
End Sub

Function fitness(Arr() As Double) As Double
  Dim excess_return, component1, component2, retTB, costs, return_inv, signal As
Double
  Dim k, periodindex, r, decision As Integer

  retTB = 1.73

  Sheets("Calculation").Activate

  component1 = 0
  component2 = 0
  costs = 0
  return_inv = 0
  excess_return = 0

  'First term
  j = 0
  For i = 1 To 422
    periodindex = i + 5
    j = 0
    signal = 0
    While j < D
      'Cells(i + 45, j + 9).Value = Arr(j)
      signal = signal + Cells(periodindex, j + 9) * Arr(j)
      j = j + 1
    Wend

    'Cells(i + 45, 14).Value = signal

    If signal > 0.5 Then
      decision = 1
    Else
      decision = 0
    End If

    'Cells(i + 45, 15).Value = decision

    'r = (1 - 0) * Rnd 'EKAF
    'component1 = component1 + r * Cells(periodindex, 21) 'r by signal
    component1 = component1 + decision * Cells(periodindex, 21)

    'Cells(i + 45, 16).Value = decision * Cells(periodindex, 21)

    If decision = 0 Then
      component2 = component2 + retTB
      'Cells(i + 45, 17).Value = retTB
```

```
      End If

   Next

   costs = Range("Z429")
   return_inv = component1 + component2 + costs

   excess_return = return_inv - Range("V46")

   fitness = excess_return
End Function


'implementation of bubble sort
Sub sort_ffa()
   Dim i, j, k As Integer
   Dim z As Double
   i = 0
   'initialization of indexes
   While i < n
      Index(i) = i
      i = i + 1
   Wend

   'Bubble sort
   i = 0
   While i < n - 1
      j = i + 1
      While j < n
         If Iy(i) > Iy(j) Then
            z = Iy(i) 'exchange attractiveness
            Iy(i) = Iy(j)
            Iy(j) = z

            z = f(i)
            f(i) = f(j)
            f(j) = z

            k = Index(i)
            Index(i) = Index(j)
            Index(j) = k

         End If
         j = j + 1
      Wend

      i = i + 1
   Wend
```

```
End Sub

'replace the old population according the new Index values<
Sub replace_ffa()
   Dim i, j As Integer
  'copy original population to temporary area
   i = 0
   While i < n
      j = 0
      While j < D
         ffa_tmp(i, j) = ffa(i, j)
         j = j + 1
      Wend
      i = i + 1
   Wend

   'generational selection in sense of EA
   i = 0
   While i < n
      j = 0
      While j < D
         ffa(i, j) = ffa_tmp(Index(i), j)
         j = j + 1
      Wend
      i = i + 1
   Wend
End Sub

Sub move_ffa()
   Dim i, j, k As Integer
   Dim scaler, r, beta, beta0, tmpf As Double

   i = 0
   While i < n
      scaler = Abs(ub(i) - lb(i))
      j = 0
      While j < n
         r = 0

         k = 0
         While k < D
            r = r + (ffa(i, k) - ffa(j, k)) * (ffa(i, k) - ffa(j, k))
            k = k + 1
         Wend
         r = Sqr(r)

         If (Iy(i) > Iy(j)) Then 'brighter and more attractive_
            beta0 = 1
            beta = (beta0 - betamin) * Exp(-gamma * r ^ 2) + betamin
```

```
            k = 0
            While k < D
               tmpf = alpha * (Rnd - 0.5) * scaler
               ffa(i, k) = ffa(i, k) * (1 - beta) + ffa_tmp(j, k) * beta + tmpf
               k = k + 1
            Wend

         End If

         j = j + 1
      Wend
      Call findlimits(i)
      i = i + 1
   Wend

End Sub

Sub findlimits(k)
   Dim i As Integer

   i = 0

   While i < D
      If (ffa(k, i) < lb(i)) Then
         ffa(k, i) = lb(i)
      End If
      If (ffa(k, i) > ub(i)) Then
         ffa(k, i) = ub(i)
      End If
      i = i + 1
   Wend

End Sub
```