Portfolio Composition using Fundamental Analysis and Rule Optimization

Renato Afonso Henriques
renatohenriques@tecnico.ulisboa.pt
Instituto Superior Técnico

***Abstract*—Artificial intelligence and the rapid growth of computational power has revolutionized the way financial trading is done. Computers are now able to make faster and more accurate decisions thanks to their capacity to process larger quantities of data in a shorter amount of time. In this work, we propose a trading system that selects companies based on their financial performance and historical stock data to actively manage a portfolio. The proposed system starts by filtering companies with a steady financial growth in order to restrict possible investments to stocks that are less vulnerable to unexpected price drops. A set of trading rules is used to evaluate how far a stock is from its desirable conditions, buying and selling it accordingly. The importance weights given to each rule are optimized in order to adapt this evaluation to the underlying market conditions, with the goal of maximizing the portfolio's profitability. We explore the use of Particle Swarm Optimization for solving this problem, testing adaptive variants and different topologies. The proposed strategy was simulated using daily data from S&P500 index constituents during the period between January 2015 and January 2018, which allowed to test the approach under several market conditions. The system achieved an annual average return of 11.23% and outperformed the benchmark during periods of high volatility. The financial filtering method itself showed very promising results during uptrending markets.**

***Index Terms*—Trading System, Portfolio Composition, Fundamental Analysis, Technical Analysis, Particle Swarm Optimization, Evolutionary Computing**

I. Introduction

The stock exchange market is known for moving a vast amount of capital. The fortunes of the top 1% richest people in the world derive mainly from investments held in this market. By trading stocks, investors seek to generate an income from their idle money, hoping to increase their wealth in the future. However, such reward is not easily obtained. Stock investing, as the majority of investment types, carries the potential risk of losing part or all of the money invested.

While some ventures are successful and bring fortune to the investor, others, due to bad luck or merely negligence, may remain fruitless, failing to provide any profit or even devalue the invested capital. However, no intelligent investor likes to leave his hard-earned money in the hands of faith. For this reason, over the years, this subject has been thoroughly studied by many experts and financial researchers.

The uprise of a digital world and increase in computational power brought forth intelligent and automatic trading systems, capable of mitigating most of the previous limitations. Furthermore, Artificial Intelligence allowed the creation of trading systems capable of making faster and more accurate investment decisions than what a human is typically capable of, mostly thanks to their ability to examine larger quantities of financial data in a shorter amount of time.

Nevertheless, the process of creating trading rules, or guidelines to decide when to buy or sell a company's shares, remains a big challenge due to the immense amount of factors that move the stock market. To circumvent such difficulties, the selection of optimal combinations of rules and computation of suitable parameters for indicators are performed by optimization techniques, while the selection of the trading rules to use are commonly left for the investor or expert to decide.

Problems with foundations like these are commonly classified as Rule Combination problems [1], and Evolutionary Computing algorithms are a typical choice to approach them [2–8]. This class of algorithms has proved to be capable of efficiently finding nearly optimal solutions for a wide-range of problems. Moreover, their adaptability makes them suitable for optimizing complex functions, which are typically hard to solve. Particle Swarm Optimization (PSO), a metaheuristic from this family of algorithms has been praised by its simplicity and low computational complexity. However, the application of this algorithm in solving Rule Combination problems has hardly been explored.

Previous works that focused on developing trading strategies featuring rule combination showed promising results [3–8], provided providing indications that it is a viable approach to investing on the stock market. However, a strategy capable of performing well during both uptrend and downtrend markets has yet to be presented.

In this work, we develop a system capable of actively managing a stock portfolio in a profitable way. We explore the use of fundamental analysis to restrict investments to companies more likely to have a value increase and less susceptible to market volatility. In addition, a rule optimization approach is used to adapt the effectiveness of each trading rule to the present market conditions. We propose the use of PSO to solve this problem, testing adaptive variants and different topologies to find the most suitable.

This document is organized as follows: Section II presents the background of this work and approaches the Rules Combination problem using Evolutionary Computing techniques. In Section III, we describe the proposed system. In Section IV, we present the experiments performed and evaluate the system performance. Finally, the final conclusions are stated in Section V.

II. Backgroud and Related Work

A. Stock Trading

A stock is a type of security that indicates ownership of a corporation and represents a claim on part of its earnings. Stocks can be profitable to its owner in either two ways: by trading stocks, i.e, playing with price variations to attain profit, or by receiving dividends. Investors typically divide their capital throughout several companies in order to form a stock portfolio, reducing the inherent risk of putting money into a single company. A portfolio can be managed passively, which means that an investor tries to mirror the performance of a market index, thus its composition is not frequently changed. In opposition, an active management of a portfolio implies the application of trading strategies and investment decisions with the aim of outperforming the market. There are two main methods to evaluate investments and find trading opportunities: Fundamental analysis and Technical analysis.

1) Fundamental Analysis: This type of analysis considers the financial performance of a company the main influencer of the evolution of its stock price. This field provides means of evaluating an investment based on an in-depth analysis of related financial, economic, qualitative and quantitative factors. It is also used to estimate a quantitative value for the expected price of a stock and compare it with its current real price, estimating if the stock price is undervalued or overvalued. Following the premise that the market is fair and prices tend to adjust to reality over time, this information can be used to predict how a stock price will behave in the future.

2) Technical Analysis: Technical analysis focus on identifying trading opportunities by examining trends in historical stock data, like price movement or volume. This discipline considers that fundamental properties of a company are already represented on its stock price, and therefore, it is the only relevant information that an investor needs to analyze. Technical analysis also assumes that price movements are not random, moving instead in identifiable patterns and trends that repeat over time.

B. Particle Swarm Optimization

Particle Swarm Optimization is a population-based stochastic optimization technique inspired by social behaviors in flocks of birds and schools of fish. It solves a problem by using a population (swarm) of candidate solutions (particles) that move around in the search space in order to find positions that correspond to optimal solutions. Every iteration k, each particle i moves by updating its position $\overline{x}$ using

$$\overline{x}_i^{k+1} = \overline{x}_i^k + \overline{v}_i^{k+1}, \tag{1}$$

where $\overline{v}_i$ corresponds to its velocity. This vector controls the direction and distance of the movement and is given by

$$\overline{v}_i^{k+1} = \omega \overline{v}_i^k + c_1 r_1 (\overline{p}_i^k - \overline{x}_i^k) + c_2 r_2 (\overline{g}^k - \overline{x}_i^k). \tag{2}$$

The velocity is a sum between three terms: the previous velocity of the particle, the local best found position found by the particle ($\overline{p}_i$), and the best position found by any particle ($\overline{g}$). The influence of each term is controlled by the inertia weight ω and the acceleration weights c_1 and c_2. By balancing each term, the swarm is expected to move towards the best solution to the problem. The scalars r_1 and r_2 are random values generated in each iteration and their purpose is to induce randomness into the search operation. The search ends when all the particles present the same solution, a previously defined fitness level is reached, or when a maximum number of iterations is performed.

C. Approaches on Rule Combination using Evolutionary Computing algorithms

Rule combination is a type of optimization problem that involves combining logical statements or weights with a set of trading rules in order to achieve a certain goal, such as profitability or risk reduction. Gorgulho et al. [4] proposed an approach based on this concept to estimate the importance that a system should give to a set of technical indicators in order to maximize a portfolio's profit. The authors introduced a classification model to calculate the score ψ of a company's stock, given by

$$\psi = \sum_{i=0}^{N} W_i * Score(X, i), \tag{3}$$

where variables W_i are unknown weights that reflect the importance assigned to the technical rule i, N is the number of technical indicators used and $Score(X, i)$ is the classification that rule i gave to stock X in a given day.

They used single-objective genetic algorithm (GA) to find the weights W_i and the limits of ψ associated with each trading action (buying and selling the stock) that maximize the profit of the strategy during an time period. This strategy provided satisfactory results, especially during the financial crisis of 2008-2009, in comparison to the S&P500 index.

Wang et al. [6] uses a model similar to (3), however, in this case, the weights are not to evaluate which technical indicator is more effective, but rather to judge which settings are more appropriate to use for a single indicator. Since these settings have a larger number of possibilities, the optimization object ends up being very large (145 dimensions), which may difficult the task of finding quality solutions. This work also proposes a system of reward and penalty to emphasize the performance of each rule while trading. The general idea is to decrease or increase the weight of the rule according to its performance (measured in profitability) during the trading simulation. The degree of penalty or reward is controlled by two parameters that are also part of the optimization object. A Time Variant Particle Swarm Optimization is proposed to solve this problem. The results showed that using an optimal combination of rules outperformed the use of the rules individually.

Most works found in the literature focus on rules based on technical indicators [4–6, 8]. Silva et al. [3], on the other hand, developed an approach involving both disciplines of

stock analysis. They started by selecting several fundamental indicators to evaluate how the company is performing financially and, using model (3), calculate the global value of a company. This value is used to find if the company is a desirable investment. Then, using a simple moving average the system finds if its stock is under an uptrend or downtrend, buying it or selling it, accordingly. Both this work and [9] include the parameters of several risk management techniques on the optimization object to find the ones that most suit the market conditions.

III. Proposed System

The goal of this system is to simulate a trading strategy capable of managing a stock portfolio profitably regardless of the trending state of the stock market. We define a set of financial conditions that a company must fulfill to be considered a possible investment. We use these filters to find companies that present a steady financial growth, since this financial stability means that their stock has a higher potential to increase and is less susceptible to unexpected price movements.

After obtaining a smaller set of companies, to select which company's shares to buy and sell, the system uses a set of trading rules to rank each stock according to its price trend and volatility. Thanks to the dynamism of the stock market, the effectiveness of each trading rule tends to vary with time. This means that for a trading strategy to remain successful over time it needs to adapt itself to new market conditions. Therefore, we do not provide the same weight to each trading rules during this evaluation. Instead, we develop a system that optimizes the importance of each trading rule by finding the combination that leads to the highest profits in past market conditions. Both the weights and the parameters of the ranking system are optimized using Particle Swarm Optimization. This process is repeated for each trading day during a simulation period to actively manage a portfolio.

A. System Architecture

The system is composed by four main components: *User Input*, *Data Processing*, *Rules Optimization*, and *Trading Simulator*. Different components interact with each other with the end-goal of determining the optimal weights and parameters to apply to the strategy in a future, untested time period.

As the name specifies, *User Input* is where the investor specifies the settings to run the system, either investment parameters or configurations of the optimization methods. In the *Data Processing* component, financial data, such as historical stock prices and financial statements, are loaded and processed. The *Rules Optimization* component includes the optimization methods used, namely Particle Swarm Optimization and its variants, which provide solutions for the defined problem. These solutions are evaluated using the *Trading Simulator*, which simulates an investment environment with similar conditions to those of a real scenario, where the trading strategy is used to manage a portfolio. In the end, the solution with the best performance during this time period is returned and used to configure the trading strategy to be applied in a later period of time. The system procedure and the interactions between components are represented in Figure 1.

In a summarized and concise manner, the system procedure is described by the following steps:

1) The user defines the investment and optimization parameters;
2) The required datasets are loaded and technical and fundamental indicators are calculated;
3) PSO generates several solutions to the problem, which traduce in combinations of weights associated with each trading rule;
4) Each solution given by the PSO is evaluated by the *Trading Simulator* in terms of the *Return on Investment*. Here, the portfolio is actively managed, during a certain time period, and stocks are bought and sold according to the trading signals generator whose parameters are derived from the solution;
5) The PSO continues to navigate the search space to find better solutions during a defined number of iterations;
6) After the optimization process is completed, the best solution found is validated using the Trading Simulator, during a different period of time, to verify its performance.

B. Data processing

The *Data Processing* component is used to load datasets from external files when the system starts, and process them to be used by the trading strategy. The system requires two types of financial data from a company to work: its daily historical stock prices, and its financial statements.

The historical stock prices are used to calculate the technical indicators required to define the trading rules. In addition, the *Trading Simulator* also requires this data to check the stocks prices during the simulation. The financial statements are a cluster of information regarding the financial results of a company. This strategy requires the financial statements that are released quarterly and annually. This information is used to perform the financial evaluation using the financial filters. To reduce the system's overhead, both the trading rules and the results to the financial filters are pre-calculated for each company and for each trading day.

1) Financial filters: The Financial Filters are a set of financial conditions that a company must fulfill in order to be considered as a possible investment. One of the main objectives of this thesis is to explore the use of fundamental analysis to select companies that manifest consistent financial growth and avoid those that presented negative results in a recent past. This way the system ends up with a higher chance of finding companies whose stock price will keep growing, while reducing the overall risk.

One of the stages of the system development is to select financial information that is able to share insight about a company's performance when compared with any other company. That is, indicators that are independent of companies' characteristics, such as their sector, size, or product/service.

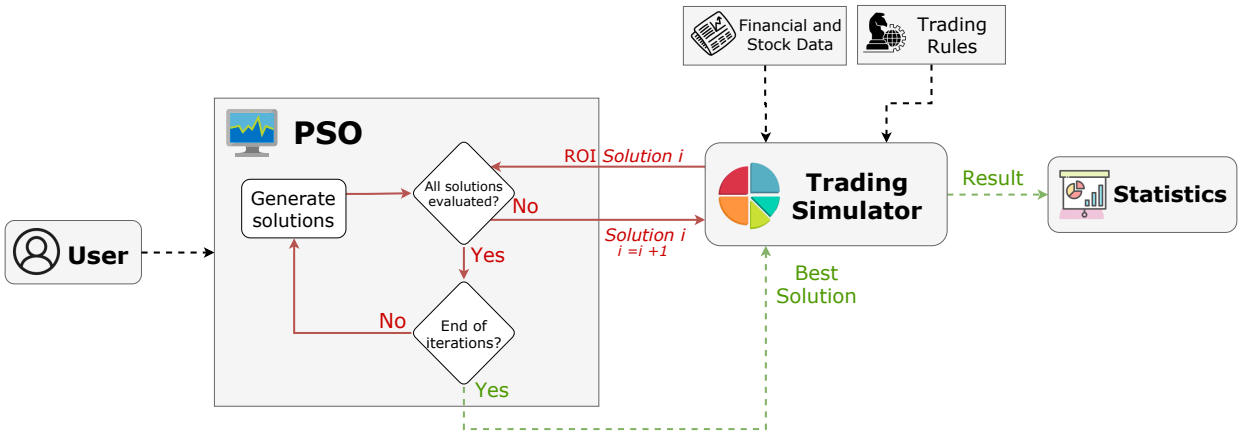

Fig. 1. Flowchart with the complete system procedure.

From the available possibilities, we consider that the two that comply the most with this requirement are the Revenue and the Net Income. Revenue is the income that a company receives by selling its goods or services to customers. The net income is the organization's income minus the cost of goods sold, expenses, depreciation, interest and taxes. In simplistic terms, it is the cash left over after paying all the expenses of an endeavor. Since these two figures are very representative of the performance of a business, we expect them to have a high impact on the company's stock price. We perform an in-depth study to analyze how they influence the growth of the stock price of a company. This analysis is presented in Section IV-A, where, based on the obtained results, we define each Financial Filter.

2) Trading rules: The trading rules are a set of rules that verify if a stock gathers the desired conditions, at a given time, to be bought or sold. These conditions reflect if the stock price is likely to increase or decrease, and, as a response, trigger an action such as buying or selling shares.

These rules are defined by establishing trading conditions over the state of a given technical indicator. To connect each indicator to a trading operation, we develop a score system that is divided in three levels: *Buy signal* with a score equal to 1, that indicates that the moment is appropriate to buy the stock, *Sell signal* with a score equal to -1, that indicates that the trade should be closed, and, finally, the *None signal*, which means that the trading rule is not providing meaningful information at an instant, so no action should be taken. Table I provides a summary of the score system used.

TABLE I
SUMMARY OF THE SCORE SYSTEM USED BY EVERY TRADING RULE.

Signal	Score	Description
Buy	1	Indicates that the stock should be bought
None	0	No action should be taken
Sell	-1	Indicates that the stock should be sold

From the vast range of technical indicators available, it is important to find those that meet with the goals of the strategy. Since the system also has a strong basis on fundamental analysis, we look for trading rules that can express if a company's shares are not too volatile, i.e., they do not show frequent price swings, or if they are likely to increase in a close future, either by evaluating if it is under an uptrend or assessing if it is oversold. Additionally, we want indicators that react mainly to big price movements, ignoring small dips that may generate false signals that the market's trend is reversing.

The trading rules employed are based on three technical indicators: the Double Crossover of Exponential Moving Averages, Bollinger Bands and the Exponential Moving Average. Note, however, that the system can easily be extended to accept a larger set of technical indicators.

a) Exponential Moving Average: The Exponential Moving Average (EMA) is a type of moving average that places a greater weight to the most recent data points. EMAs are used to confirm market trends or evaluate their strength. In this strategy, the EMA is used as a support sell signal indicator to confirm strong downtrends before the other indicators do. A 50 days Exponential Moving Average is employed with this purpose. The score system and the summary of the rules are presented in Table II.

TABLE II
RULES DEFINED FOR THE EMA INDICATOR. THIS AVERAGE IS CALCULATED USING AN INTERVAL OF 50 DAYS.

Signal	Score	*EMA50*
Buy	1	-
None	0	EMA line has a positive slope
Sell	-1	EMA line has a negative slope

b) Double Crossover of Exponential Moving Averages: This technical indicator is called Double Crossover (DC), and uses the crossing between a shorter and a larger moving average to estimate if a new trend has taken place. An uptrend is confirmed with a bullish crossover, which happens when a short-term moving average crosses above a longer-term moving average. Similarly, a downtrend is confirmed with a bearish crossover, which occurs when a short-term moving average crosses below a longer-term moving average. Table III

presents the definition of the trading rules and their respective score.

TABLE III
RULES DEFINED FOR THE DC INDICATOR. THE EMA(S) CORRESPONDS TO THE SHORTER AVERAGE AND EMA(L) TO THE LONGER AVERAGE.

Signal	Score	*DCS-L = EMA(S) - EMA(L)*
Buy	1	EMA(S) is above the EMA(L) line
None	0	-
Sell	-1	EMA(S) is below the EMA(L) line

The developed system uses two Double Crossovers, one between an EMA of 50 days and an EMA of 200 days (DC50-200), and other between an EMA of 20 days and an EMA of 50 days (DC20-50). The goal of the DC50-200 is to catch the reversal of stocks with long trends, which is what usually the case of companies with reliable financial performances. The DC20-50 is used to catch reversals of stocks whose price significantly increased or decreased more quickly, which is what typically happens when promising companies with undervalued stocks get sudden attention by a large number of investors.

c) Bollinger Band: The Bollinger band (BBANDS) is a technical analysis tool to measure the market volatility. It is also used to evaluate the state of a trend and to check if the market is in a period of consolidation. It is defined by a set of lines that deviate, positively and negatively, from a moving average of the stock's price by a certain standard deviation. These two lines form two bands (upper and lower band) that widen or contract according to how volatile or non-volatile a market is. For this system, the Bollinger bands are calculated using a Simple Moving Average of a period of 50 days and a standard deviation of 2.5$. These rules and respective scores are presented in Table IV.

TABLE IV
RULES DEFINED FOR THE BOLLINGER BAND INDICATOR. THE BOLLINGER BANDS ARE CALCULATED USING A SIMPLE MOVING AVERAGE OF 50 DAYS AND A STANDARD DEVIATION OF 2.5$.

Signal	Score	*BBANDS*
Buy	1	Price line under the Lower Bollinger Band
None	0	Price inside the Bollinger Bands
Sell	-1	Price line above the Upper Bollinger Band

C. Rules Optimization

The dynamics of the stock market are consistently changing due to numerous external factors that are taking place in the world. In addition, the investment world has recently started to heavily adopt the use of algorithmic trading to perform high frequency trades, which is able to quickly react to new information, changing the landscape of the stock market even further. As a consequence of this market dynamism, the effectiveness of each trading rule varies over time. However, by considering that the market conditions of the present are similar to those of a close past, one can try to find which rules were more effective in recent times and apply that knowledge afterwards.

The problem we want to solve consists on finding the real valued vector t, whose elements correspond to parameters of the developed trading system, that maximizes the profits of the managed portfolio. We formulate the problem as a constrained optimization problem (COP), which means that the solution is subjected to constraints that must be fulfilled for it to be accepted.

In total, t is composed by 7 elements. The first four entries correspond to the weights W_i associated with each trading rule i. These weights measure the importance attributed to each rule and are limited between 0 and 1. Additionally, $\sum_{i=1}^{4} W_i \in [0, 1]$. The 5th and 6th elements are the $BuyLimit$ and $SellLimit$, respectively. These two values are used by *Trading Simulator* to select if a stock should be bought, sold, or ignored. The $BuyLimit$ is bounded by 0 and 1, while the $SellLimit$ can be a value between -1 and $BuyLimit$. At last, the 7th element is the $StopLoss$ parameter used to define the maximum percentage which a stock's price is allowed to fall before it is removed from the portfolio. The mapping of the 7-dimensional position vector to the respective parameters is displayed in Figure 2.

t_1	t_2	t_3	t_4	t_5	t_6	t_7
$W_{DC50-200}$	$W_{DC20-50}$	W_{EMA}	W_{BBANDS}	$BuyLimit$	$SellLimit$	$StopLoss$

Fig. 2. Mapping of the 7-dimensional solution vector.

The problem solver navigates the collection of all possible solutions to the optimization problem (search space), in order to find the solution that maximizes the objective (or fitness) function. This objective function is the *Trading Simulator* described in Section III-D. For each solution vector provided to the *Trading Simulator*, it will return a score or fitness, that is measure of its quality. This measure corresponds to the *Return on Investment* (ROI), in percentage, of the portfolio during the period of simulation.

1) Particle Swarm Optimization: We solve this problem using Particle Swarm optimization. This metaheuristic is suitable to search very large spaces of candidate solutions and deal with complex objective functions, which is the case of this problem. We employ, in specific, a PSO variant called Adaptive Particle Swarm Optimization (APSO). This method, proposed by [10], uses a technique called Evolutionary State Estimation (ESE) to identify the evolutionary state which the optimization process is going through, and automatically control the inertia weight and acceleration coefficients, accordingly. This technique is expected to improve significantly the convergence speed of the algorithm, while increasing the possibility of the swarm getting trapped in local optima, as a consequence. As a countermeasure, APSO also uses an Elitist learning strategy (ELS) to help the globally best particle jump out of these regions.

Since the canonical PSO is not predefined to solve this COPs, we apply the method suggested in [11] to permit the APSO to solve this type of problems. In addition, as suggested

in [12], to prevent the swarm from exploding, the velocity vector is bounded by the difference between the lower bound and the upper bound of the position vector values.

A detailed analysis of the performance of PSO and APSO, as well as a comparison between swarm topologies, are provided in the extended version of this document [13].

D. Trading Simulator

The trading simulator, as the name indicates, runs investment simulations to test a strategy and evaluate its performance. The simulation starts with an empty portfolio with a limit size, predefined by the user. The procedure of the trading simulator can be observed, in summary, in Figure 3.

The first step is to find, from a set of companies, the ones that fulfill the Financial Filters discussed in Section III-B1 and defined in Section IV-A. Only the companies that satisfy these conditions are considered during the remaining process.

For each day of the trading simulation, a company's stock price is evaluated using the set of trading rules, described in Section III-B2. Since the model of expression (3) showed satisfying results in related works [4, 6], we use it to combine the scores of the trading rules with their respective weights. This model returns a value ψ that corresponds to the rank of the stock. This ranking reflects how suitable are its present conditions to be bought. Higher the rank, better the opportunity. We use this information to select the set of companies that should be bought or sold. Figure 4 presents an example on how the ranking system selects which stocks are bought or sold on a certain trading day. In this simplified case, only three companies are considered.

To select which stocks should be bought, we start by sorting them by their rank. The companies whose stock's ranking is higher than the $BuyLimit$ threshold are included in the portfolio. In the example of Figure 4, the rank of company X's stock is higher than the given $BuyLimit$, so a buy signal is sent.

Since the portfolio has a limited size, it is not possible to buy all the desired companies. Therefore, the number of different stocks chosen to buy will depend on the number of available slots in the portfolio. For example, if the portfolio only has two slots available, then the two stocks with higher rank will be the ones that the system will acquired. The system invests equally on each slot of the portfolio.

The stocks that compose the portfolio can be sold when either one of the following two conditions arise:

- If the rank of the company's stock falls below the $SellLimit$;
- If the price of a stock is inferior to the value of the stop-loss.

If the ranking of stock already present in the portfolio is inferior to the $SellLimit$ then the stock is removed from the portfolio. However, note that this only happens if the stock has already been profitable. We decided to implement this exception so that the stocks are not immediately sold if they have a bad start, providing them with an initial slack. In the example of Figure 4, company Y's stock ranks below the $SellLimit$. If this company is already part of the portfolio then its stock is sold. Otherwise, the system does not execute any trade. A company whose stock ranks between $BuyLimit$ and $SellLimit$ is ignored, regardless if it is part of the portfolio or not. This is the case of company Z.

Additionally, we use a trailing stop-loss to close positions whose price fall under a certain percentage, given by the parameter $StopLoss$, of the maximum price that the stock achieved while in the portfolio. The technique is employed mainly to sell stocks whose price fell significantly in a short amount of time and the technical indicators were not able to signal fast enough. This way the strategy can more easily manage the risk of each investment, reducing their susceptibility to unexpected market movements. When a position is closed, all the shares of the corresponding company are sold at the same time. This means that the corresponding slot of the portfolio is freed and a new company can take its place.

The process of filtering companies using the Financial Filters, ranking their stock using the Trading Rules and use it to decide which ones to buy or sell is repeated every day over the total duration of the simulation. In the end, the performance of the strategy is measured using the *Return on Investment*, which states how much profit the portfolio made in relation to the initial investment.

IV. Experimental Results

In this section, we start by performing an analysis of the influence of revenue and net income on a company's stock price with the goal of defining the Financial Filters. We then evaluate the developed system and analyze its performance. We use stock data and past financial statements of the constituents of the S&P500 index between 2005 and 2018. The historical stock prices of these companies were obtained from the Yahoo Finance platform [14] and their financial statements were obtained through the *Compustat - Capital IQ* database via the Wharton Research Data Services (WRDS) [15].

A. Financial Filters

To define the financial filters used by the system to find companies with the desirable financial performance, we perform an analysis of the relation between the revenue and the net income of a company with its stock price growth.

1) Influence of revenue and net Income growth on stock price: We start by searching for a relation between growth of a company's stock price and the number of years that its revenue (or net income) has been growing. Since the revenue and the net income are such important measures of a company's performance, a company capable of improving these figures repeatably shows that its business plan is robust and effective, which we expect to attract investors to invest more.

We perform a test that consists of checking the average variation of the stock price of every company whose revenue or net income grew Y years in the last X years. Notice that, for this test, the stock price variation is presented annualized and the last X years refer to the X years prior to 2018-12-28.

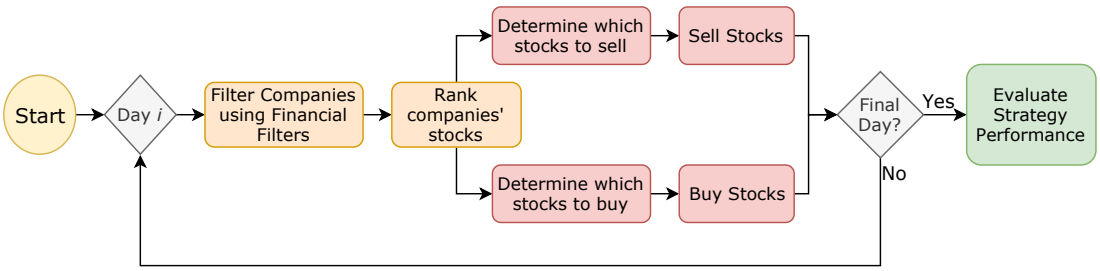

Fig. 3. Flow chart of the procedure of Trading Simulator.

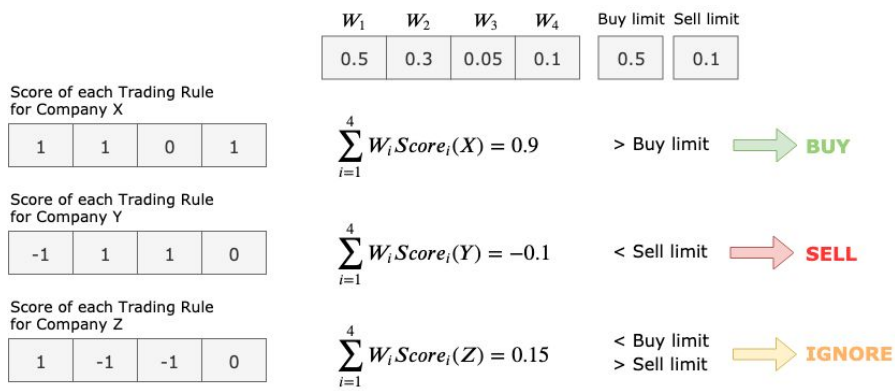

Fig. 4. An example of the use of the ranking system to select each stocks to buy and sell.

Table V presents the average stock price change for different values of X and Y in respect to the revenue. The values of the average stock price change of every company in the previous X years are presented in the last row.

The results show that there is a clear relation between the number of years that a company's revenue increased and the variation of its stock price. In general, regardless of X, the stock price tends to grow with Y. This means that companies whose revenue grows more frequently have, in average, a higher stock price increase than those that do not. Notice especially the difference between the variation of stock price of companies whose revenue grew X years in the last X years in comparison with the ones that only grew $X-1$ years in X. On average, the stock price of the former set of companies grew 4.586% more than the latter. This is, not only a very significant difference, but also suggests that one year of a decreasing revenue is enough to significantly affect the stock price of a company.

In 7 of the 9 values of X tested, the stock price of companies with revenue's increasing years higher than $X-1$ years grew more than the average of stock prices of the complete set of companies in the last X years, observed in the last row. However, for X inferior to 4 years, only companies with Y equal to X were able overcome the total average.

The same test was performed for the case of net income growth. The results are presented in Table VI. Unlike what was observed with the revenue, the relationship between the percentage change of stock prices and the number of net income's increasing years is not so evident.

First, the number of cases where the average of the stock prices change is negative is smaller, which suggests that the net income increasing less years does not have such a negative impact on the stock price as the revenue. On the other hand, it is noticeable that, in average, the number of years that the net income of a company must increase during the X years in order to be valued above the total average is lower. In 4 of the 9 cases, $X-3$ in X years presented higher results than the total average, while, in 2 of the 9 cases, $X-4$ in X were enough.

Despite the increase in the average percentage change of the stock prices not being as linear with the increase of Y as seen in the case of the revenue, this test confirmed, nonetheless, that investors value companies which are able to increase their net income successively every year.

2) Influence of revenue and net Income growth on stock price: We now analyze how the profitability of a company influences its stock price. Since this financial condition takes a very important role on how a company remains sustainable, it is expected that companies that had a longer streak of positive net income have a higher increase in their stock price.

Table VII shows the average percentage change of the stock price of every company that presented a positive net income during the previous X years. This percentage change is annualized and the comparison of the stock prices is done from 2018-12-28 in regard to X years prior.

There is a clear uptrend of the average stock price with the

TABLE V

AVERAGE OF THE ANNUAL PERCENTAGE CHANGE OF THE STOCK PRICE OF EVERY COMPANY WHOSE REVENUE GREW Y YEARS DURING THE LAST X YEARS. THE AVERAGE STOCK PRICE CHANGE OF EVERY COMPANY USED IN EACH EACH COLUMN IS PRESENTED IN THE LAST ROW.

		X								
		2	3	4	5	6	7	8	9	10
Y	0	-11.92	-3.701	-8.016	-8.022	-2.106	-5.156	-12.37	-9.652	-2.740
	1	-1.863	-0.669	-5.669	-2.615	4.917	8.710	-8.692	7.539	9.072
	2	5.681	5.745	-2.648	-1.623	2.761	4.675	4.944	-0.648	-1.159
	3	-	10.80	6.616	1.530	5.604	6.262	8.281	18.37	22.23
	4	-	-	10.76	9.763	9.135	8.520	5.039	5.990	9.958
	5	-	-	-	13.40	14.13	10.90	5.364	6.065	10.06
	6	-	-	-	-	17.96	15.47	9.713	9.202	11.99
	7	-	-	-	-	-	19.09	13.77	11.66	14.99
	8	-	-	-	-	-	-	17.42	14.90	15.26
	9	-	-	-	-	-	-	-	19.92	19.84
	10	-	-	-	-	-	-	-	-	24.61
	Total	2.757	6.927	4.381	6.991	11.91	13.06	11.46	13.52	16.80

TABLE VI

ANNUAL AVERAGE OF THE PERCENTAGE CHANGE OF THE STOCK PRICE OF EVERY COMPANY WHOSE NET INCOME GREW Y YEARS DURING THE LAST X YEARS.

		X								
		2	3	4	5	6	7	8	9	10
Y	0	-10.17	1.797	-13.44	-12.07	-5.190	-4.249	-10.44	-6.405	-2.740
	1	1.126	1.427	-3.104	-1.355	9.597	23.09	0.000	20.01	21.88
	2	8.511	7.013	1.276	1.060	4.135	4.322	19.33	38.35	50.03
	3	-	15.75	8.743	7.898	8.074	7.219	4.889	6.633	6.832
	4	-	-	12.83	11.24	13.36	12.34	6.683	5.785	7.498
	5	-	-	-	13.53	18.29	17.06	12.17	10.28	12.97
	6	-	-	-	-	17.37	18.04	15.56	13.51	15.78
	7	-	-	-	-	-	16.20	15.71	17.43	17.87
	8	-	-	-	-	-	-	15.78	17.62	20.18
	9	-	-	-	-	-	-	-	21.05	23.67
	10	-	-	-	-	-	-	-	-	24.49
	Total	2.757	6.927	4.381	6.991	11.91	13.06	11.46	13.52	16.80

TABLE VII

ANNUAL AVERAGE PRICE GROWTH OF COMPANIES THAT HAVE HAD THE LAST X YEARS WITH POSITIVE NET INCOME.

X years	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Nº of companies	501	459	415	398	386	376	360	339	329	294
Average stock prices percentage change (%)	-2.301	0.272	1.534	1.336	2.741	5.510	7.032	6.572	8.184	10.48

increment of X. In addition, as the number of years increase, the number of companies that showed profitability decreases. This means that, although it is challenging to remain profitable for several consecutive years, in general, the companies that were able to do so, were rewarded with a higher stock price growth. A possible justification is as follows: a company that is consistently profitable provides more assurance that its stock price will not plunge unexpectedly and has a higher chance of keeping its increase, thus attracting more investors.

3) Definition of Financial Filters: Based on the results and conclusions taken from the previous tests, we define three financial filters to find companies with desirable financial conditions and, therefore, with a higher expectancy of stock price growth. The three conditions that we consider that have to be fulfilled to be considered as an investment by the *Trading Simulator* are:

- The company must have its revenue successively increasing for, at least, the past Y_1 years;
- The company must have its net income successively increasing for, at least, the past Y_2 years;
- The company must be profitable every year for, at least, the past Y_3 years.

The years $\{Y_1, Y_2, Y_3\}$ are a set of integer values that can be fine tuned by the user. Note that the set of values that provide the best results are not necessarily the same over the years. Ideally, the best set of values for a certain time period should be determined using an optimization method, in a similar process to the one done with the trading rules. However, the best values for each variable must be between 2 years and 7 years, since for a smaller number of years the financial filters would not be selective enough and because there is a very small number of companies with a revenue or net income that fulfill these requirements for more than 7 years. Therefore, the number of possible combinations of Y_1, Y_2 and Y_3 is small. For this reason, we decided instead to test experimentally several combinations of years using a *Buy&Hold* simulator that buys a share of every company that fulfills the financial conditions and does not sell them until the end of the simulation. We

found that the set $\{Y_1, Y_2, Y_3\} = \{5, 3, 3\}$ provides the best performance, for most market conditions.

As observed in Table V, on average, the increase of the stock price of companies whose revenue continuously increased for several years is larger than for companies whose revenue did not increase in at least one of the years. Taking this into account, we also implement a rule to sell every company from the portfolio whose revenue stops growing. This way we expect to remove companies whose stock price growth rate is likely to decrease and open space to other companies that might be under better financial conditions.

B. System performance

In this section, we evaluate the strategy derived from the experiments previously performed. We use both the Financial filters defined in Section IV-A and the Rule Combination system optimized with APSO. We apply the trading strategy to out-of-sample data and measure its performance in various market conditions. Additionally, the strategy is compared with the benchmark, the S&P500 index.

We start by stating the configurations of the system. APSO uses a swarm of 20 particles and runs for 50 iterations. The initial values of the acceleration coefficients, c_1 and c_2, are 2.0, and the maximum and minimum value of the inertia weight are, respectively, $\omega_{max} = 0.9$ and $\omega_{min} = 0.4$. Additionally, the maximum and minimum values of the elitist learning rate, used by the APSO, are $\sigma_{max} = 1.0$ and $\sigma_{min} = 0.1$, respectively. Regarding the investment settings, during the trading simulation the system cannot include more than 30 companies in the portfolio, and the transaction cost applied to each trade (either buying or selling) is 0.5%. The set of years that the Financial Filters uses is $\{Y_1, Y_2, Y_3\} = \{5, 3, 3\}$.

We first analyze the system performance in a total test period of three years, starting from 2015-01-15 and ending at 2018-01-18. We select this test period, because, during its course, the U.S. stock market goes through different behaviours, namely, times of high volatility followed by a clear uptrend, which allow us to observe how the system behaves in different types of market. During this experiment, we use a sliding window with a size of one year, which means that each year the APSO is executed again to find the most suitable parameters taking into account the data of the 2 previous years. Figure 5 presents the Return on Investment, in percentage, of the average results over 5 independent executions of the system and the Buy&Hold strategy of the S&P500.

We observe that the developed system is robust to the market volatility during the period between mid 2015 to mid 2016, capable of outperforming the index. At the same time, the trading rules were not able to effectively prevent the big sudden price falls that happened during this period. We also notice that, while the system increases its returns from mid 2016 to the end of the simulation, they are significantly inferior to the Buy&Hold of the S&P500 index. These results suggest that the system is more suitable as an investment application during periods of volatility and or when the market is downtrending than when it is uptrending.

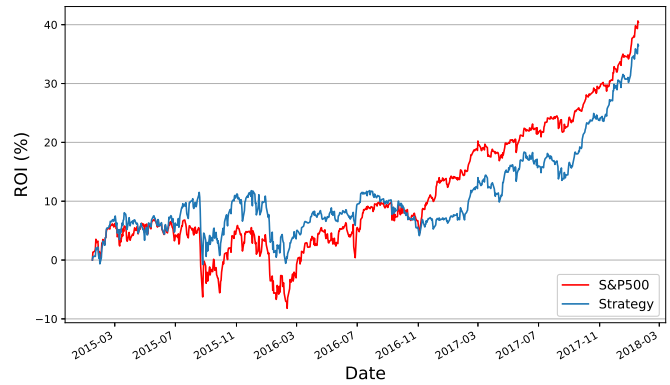

Fig. 5. System performance, in terms of the ROI (%), from 2015-01-15 to 2018-01-18. It presents the average of every execution of the system and the Buy&Hold of the S&P500 index.

An issue of the proposed approach is that its performance is highly dependent on the period where the rule optimization is performed. The system was optimized using the market conditions of the period between 2015-01-16 and 2017-01-17, when the market was volatile and the best strategy was probably not to buy any stock. However, the result of the training was applied during the period between 2017-01-18 and 2018-01-18, when there is a bull market. In other words, the system adapted its parameters to market conditions quite distinct to those of the market when it was tested, which may have significantly influence the system performance. We verify in [13] that if the conditions of the training and testing phases are similar then the performance of the system improves significantly regardless of the market conditions. We also consider that, by shortening the size of the training period, it may be possible to adjust more accurately the system optimization to the conditions of the test period. However, the size must not be too short, as over-fitting gets more likely to occur, leading the system to perform badly if the testing conditions change slightly with respect to the training ones.

We also evaluate the influence of the rule optimization method in the results by comparing the performance of the system with and without it. The system without rule optimization buys a stock of every company which fulfills the financial filters, and only sells them if the company's revenue stops growing. The results between these two strategies and the Buy&Hold of S&P500 index are presented in Figure 6. The trading system is executed in the same conditions and time periods as in the experiments of Figure 5.

We observe that, while both strategies are capable of outperforming the index during the volatile period, the system is capable of achieving higher returns during this period using rule optimization than without it. However, we also observe that when the uptrend is established (starting from 2016-03 until the end of the simulation), the system using only the financial filters performs better than system featuring rule optimization. Moreover, during this period, it presents better results than the S&P500 index itself.

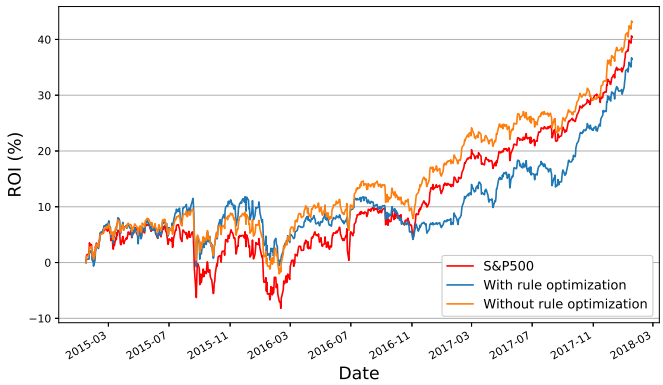

Fig. 6. System performance, in terms of the ROI (%), over the period between 2015-01-15 to 2018-01-18. It presents the average results of the system with and without rule optimization system, and the benchmark, the S&P500 index.

Table VIII presents an evaluation through several performance metrics of the system with and without the rule optimization system, and of the Buy&Hold strategy of the S&P500 index during the test period. The results show that the system with or without rule optimization has a MDD superior to the S&P500 index, which means that it is less susceptible to big price falls than the benchmark. As verified by the Sharpe and Sortino ratios, the system without rule optimization shows the best risk/reward ratio of the three methods, while the system with rule optimization is only slightly worst than the index in this regard. In respect to profitability, the system without rule optimization obtains the highest annual ROI.

TABLE VIII
EVALUATION METRICS OF THE BUY&HOLD STRATEGY OF THE S&P500 INDEX AND THE SYSTEM PERFORMANCE WITH AND WITHOUT THE RULE OPTIMIZATION SYSTEM BETWEEN 2015-01-15 AND 2018-01-18.

	S&P500	**W/ rule optimization**	**W/o rule optimization**
ROI	40.42%	37.60%	43.12%
Sharpe Ratio	0.987	0.973	1.189
Sortino Ratio	1.062	1.031	1.359
Information Ratio	-	-0.139	0.112
Maximum Drawdown	-14.16%	-11.48%	-10.60%
Annualized ROI	11.98%	11.23%	12.69%

Although the performed experiments suggest that applying a strategy based only on the Financial Filters and companies' revenue growth is a more viable way of achieving good results, it is very important to consider that this strategy may require investing on a large number of companies, since we did not develop a method to select the best companies from the set of companies that fulfilled the financial conditions of the filters. For example, during the simulation, this strategy invested in a total of 99 companies. Investing in such a large number of companies requires considerable capital, which may not be viable for many investors. Additionally, since this strategy does not feature a method to automatically select the best moments to buy and sell stocks, these decisions would have to be performed manually by the investor. This means that it is more time expensive and requires more attention to each stock, which is not a simple task since the number of companies is large. With the Rule optimization system, by selecting the ones whose stock ranked the best, the number of companies can be reduced to a smaller and more manageable set.

V. CONCLUSIONS

In this work, we developed a system to dynamically manage a stock portfolio. Combining technical and fundamental analysis, we created an investment strategy that seeks companies growing financially and whose stock is predicted to increase. In addition, Particle Swarm Optimization adjusts the parameters of the strategy to the market conditions, improving the portfolio's profitability. To conclude, we consider that the proposed trading strategy is a viable approach to stock investing, and, despite the overall performance of the system being slightly inferior to the S&P500 index, its robustness to market volatility and capacity to avoid big price drops, makes it a better option for investors with higher risk adversity.

REFERENCES

[1] Yong Hu, Kang Liu, Xiangzhou Zhang, Lijun Su, EWT Ngai, and Mei Liu. Application of evolutionary computation for rule discovery in stock algorithmic trading: A literature review. *Applied Soft Computing*, 36: 534–551, 2015.

[2] Liad Wagman. Stock portfolio evaluation: An application of genetic-programming-based technical analysis. *Genetic Algorithms and Genetic Programming at Stanford*, 2003:213–220, 2003.

[3] António Silva, Rui Neves, and Nuno Horta. A hybrid approach to portfolio composition based on fundamental and technical indicators. *Expert Systems with Applications*, 42(4):2036–2048, 2015.

[4] António Gorgulho, Rui Neves, and Nuno Horta. Applying a ga kernel on optimizing technical analysis rules for stock picking and portfolio composition. *Expert systems with Applications*, 38(11):14072–14085, 2011.

[5] Antonio C Briza and Prospero C Naval Jr. Stock trading system based on the multi-objective particle swarm optimization of technical indicators on end-of-day market data. *Applied Soft Computing*, 11(1):1191–1201, 2011.

[6] Fei Wang, LH Philip, and David W Cheung. Combining technical trading rules using particle swarm optimization. *Expert Systems with applications*, 41(6):3016–3026, 2014.

[7] Jing J Liang, Shang Zhigang, and Li Zhihui. Coevolutionary comprehensive learning particle swarm optimizer. In *IEEE congress on evolutionary computation*, pages 1–8. IEEE, 2010.

[8] Xiaojia Liu, Haizhong An, Lijun Wang, and Xiaoliang Jia. An integrated approach to optimize moving average rules in the eua futures market based on particle swarm optimization and genetic algorithms. *Applied energy*, 185:1778–1787, 2017.

[9] Rodrigo C Brasileiro, Victor LF Souza, Bruno JT Fernandes, and Adriano LI Oliveira. Automatic method for stock trading combining technical analysis and the artificial bee colony algorithm. In *2013 IEEE Congress on Evolutionary Computation*, pages 1810–1817. IEEE, 2013.

[10] Zhi-Hui Zhan, Jun Zhang, Yun Li, and Henry Shu-Hung Chung. Adaptive particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(6):1362–1381, 2009.

[11] Xiaohui Hu and Russell Eberhart. Solving constrained nonlinear optimization problems with particle swarm optimization. In *Proceedings of the sixth world multiconference on systemics, cybernetics and informatics*, volume 5, pages 203–206. Citeseer, 2002.

[12] Konstantinos E Parsopoulos, Michael N Vrahatis, et al. Particle swarm optimization method for constrained optimization problems. *Intelligent Technologies–Theory and Application: New Trends in Intelligent Technologies*, 76(1):214–220, 2002.

[13] Renato Afonso Henriques. Portfolio composition using fundamental analysis and rule optimization. Master's thesis, Instituto Superior Técnico, 2019.

[14] Yahoo! Yahoo! Finance. https://finance.yahoo.com/. Accessed: 2019-05-04.

[15] Wharton Research Data Services. https://wrds-www.wharton.upenn.edu/. Accessed: 2019-05-04.