# Stock Market Index Trading Algorithm Using Discrete Hidden Markov Models and Technical Analysis

Luis Andrade

***Abstract*— This work presents an innovative approach to algorithmic stock market index trading by means of a combination of discrete Hidden Markov Models (DHMMs) using windows of daily and weekly data. The DHMMs are trained using the Baum-Welch algorithm, and the predictions are obtained with the aid of the Viterbi algorithm. In order to use the DHMMs the close price data of the stock index S&P 500 is transformed into two discrete values: drop and rise in relation to the closing price of the previous trading day. The Relative Strength Index (RSI) is used as a decision criteria to choose between the different DHMMs, and subsequently a price trend forecast is produced. Using these forecasts the algorithm is capable of autonomously trading in the stock market. The system was trained using S&P 500 index price data from January 2003 to January 2009, and it was tested from January 2009 to January 2017. The results were compared to a state of the art solution and two investment strategies: the Buy & Hold and a purely random strategy. The developed algorithm outperformed all three other approaches over the testing period, achieving a rate of return of 356%, which significantly exceeds the 199% return of the S&P 500 index.**

***Index Terms*— Financial time series, Forecasting, Hidden Markov Model, Stock Market, Technical Analysis, Trading Algorithm**

## I. Introduction

Financial markets play a critical role in the modern world as they generate transactions of large amounts of money. Lured by the great potential profits that can be obtained by investing in these markets, many players ranging from large financial institutions to small investors have become actively involved. However, these investments are subject to sizeable risks, including the risk of a complete loss of capital. The difference between making a hefty profit and going bankrupt can rely on making correct predictions which can generate proper investment decisions. As a result, many experts and academic researchers have dedicated a large amount of time and effort in creating models and tools that can predict future market trends with the highest possible degree of accuracy. However, this has proven to be a tough challenge due to the complex nature of financial markets, as they exhibit volatile behavior.

Taking advantage of the computing power available, many machine learning methods have been created and employed in order to deal with the prediction problem. Some methods, previously developed for use in other areas, were adapted to the prediction of financial time series. Such methods can then be incorporated into trading algorithms, which automatically trade in financial markets. These methods include genetic algorithms, artificial neural networks, reinforcement learning, support vector machines, and Hidden Markov models [1].

This work tackles the financial markets investment challenge recurring to the implementation of a machine learning model. The basic idea is to use several discrete HMMs in combination with technical indicators to provide accurate forecasting of financial time series. This will provide a solid foundation for the creation of an automatic trading algorithm that is capable of generating significant returns without taking excessive risks.

## II. Background

### A. Financial Markets

In a market economy, many decisions must be made in order to allocate the available resources. The price of these resources is mainly the consequence of the interaction between supply and demand. This work will focus on a specific kind of resources, called financial assets, which are mostly traded in financial markets. An asset is any resource with a tangible or intangible value in a trade. The price of a tangible asset is related to its physical characteristics, such as real estate and currencies. On the other hand, intangible assets simply represent future claims on benefits, having no dependency on the asset's physical properties. Financial assets are intangible assets since their value is derived from a contract. Examples of financial markets include the stock market, the bond market, and the foreign exchange market.

A solid market analysis is crucial in order to maximize the probability of success when investing in financial markets. Some investors chose to adopt a strategy called Buy & Hold, in which an investor simply buys an asset and sells it after some period of time. Buying a certain asset hoping that its price will rise is often referred to as opening a long position. In contrast, buying a certain asset hoping that its price will fall is often referred to as opening a short position.

Several market analysis tools have been developed. The two main market analysis approaches are the Fundamental Analysis and Technical Analysis. Both seek to aid the investor to predict market trends, but they are based on very different principles.

### B. Technical Analysis

Technical analysis relies on the analysis of charts in order to predict future market trends. This type of approach assumes that the historical performance of the market is an indication of future behavior and that price changes already incorporate all fundamental factors. This kind of analysis suggests that, through the analysis of technical indicators, certain patterns that have forecasting value can be detected [2]. Depending on the investment strategy, and due to the easily available real-time data, technical analysis can be used to make predictions within seconds.

Technical analysis is heuristic by nature, lacking a mathematical foundation. This results in a certain level of skepticism within the academic and financial communities. Graham [3] argues that an investor must avoid technical analysis, as doing so will be unprofitable in the long run. On the other hand, Kown and Kish [4], Gunasekarage and Power [5], and Chong and Ng [6] all achieve considerable returns when using technical trading methods.

The Relative Strength Index (RSI) is a momentum indicator that attempts to identify overbought and oversold conditions of a financial asset. The RSI tracks an asset´s losses and gains over a certain period of time and generates buy and sell signals accordingly. Since it measures the asset's price directional movements, the RSI is called a momentum oscillator. The RSI calculation is as follows [7]:

$$RSI = 100 - \frac{100}{1+RS} \tag{1}$$

where RS is the ratio of the Average Gain (AG) to the Average Loss (AL),

$$RS = \frac{AG}{AL} \tag{2}$$

$$AG = \frac{(Previous\ AG)(time\ period - 1) + Current\ Gain}{time\ period} \tag{3}$$

$$AL = \frac{(Previous\ AL)(time\ period - 1) + Current\ Loss}{time\ period} \tag{4}$$

This indicator outputs a value in the range of 0 to 100. If the value computed is greater than or equal to 70 the asset is considered to be overvalued and therefore should be sold. If the value is less than or equal to 30 the asset is considered to be undervalued and therefore should be bought [2]. An illustration of buying and selling opportunities identified by the RSI is illustrated by figure 1. As can be seen in the figure, the RSI (in blue) crosses above the sell mark of 70 (in green) days before the 1st of September, and after some stagnation the price of the S&P 500 (in red) starts to decrease. Additionally, shortly after the 13th of October the RSI crosses below the buy value of 30 (in purple), and the price of the S&P 500 switches from a downwards trend to an upwards trend.

## C. Investment Metrics

Investment metrics are used in order for an investor to assess the quality and performance of his investments. These metrics can measure the return and risk, and two common metrics used are the Rate of Return and the Sharpe Ratio.

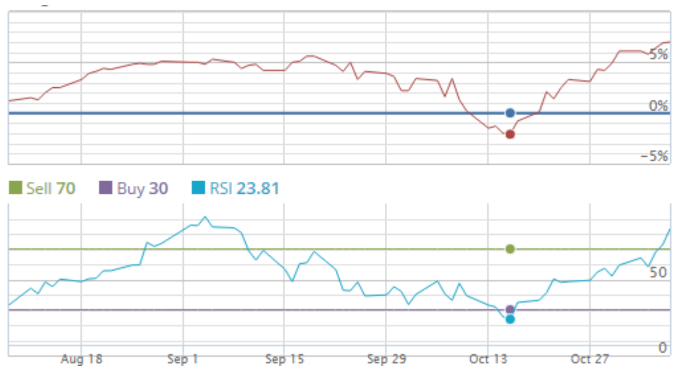


*Figure 1. Evolution of the S&P 500 index (in red) and the RSI (in light blue) from 8/08/2014 to 11/08/2014*

The Rate of Return (ROR) is used to measure the relative gain or loss of an investment over a particular period of time. In other words, the ROR is the earnings an asset generates in excess of its initial cost. To calculate the ROR one can use the following formula,

$$ROR = \frac{Final\ Asset\ Value - Initial\ Asset\ Value}{Initial\ Asset\ Value} \tag{5}$$

The ROR is often calculated over time periods of months or years. Naturally, a positive ROR is always desired as this means that profits have been made. The greater the ROR, the more profitable the investment is.

The Sharpe Ratio is a metric widely used to calculate risk-adjusted return (i.e., how much risk is involved in producing the specified return). It can be described as the mean return subtracted by the risk-free rate, and then divided by the standard deviation of the asset (which is a way to calculate volatility). To calculate the Sharpe Ratio one can use the following formula,

$$Sharpe\ Ratio = \frac{Mean\ return - Risk\ free\ rate}{Standard\ deviation\ of\ return} \tag{6}$$

An investment with a Sharpe Ratio of exactly zero is said to be risk-free. One such investment is to acquire U.S. Treasury bills since the expected return, by definition, is the risk-free rate. A negative Sharpe Ratio indicates that a risk-free investment would perform better than the asset being analyzed, and should therefore be avoided. Higher values of the Sharpe Ratio correspond to better risk-adjusted returns

## D. Hidden Markov Model

A Markov Chain is a stochastic process that satisfies the Markov property. This means that if a series of random variables constitutes a Markov Chain, the future state of each of those variables only depends directly on its current state. The Hidden Markov Model (HMM) can overcome some of the limitations faced by Markov Chains. As the name suggests, the state is not directly observable in this model. Nevertheless, the observations (which are directly observable, by definition) are tied to each state by a probability distribution. The observations can also be referred to as emissions. The HMM satisfies the Markov property, and so we have (7),

$$P(S_t|S_1, S_2, \dots, S_{t-1}) = P(S_t|S_{t-1}) \quad (7)$$

This means that each state S is only dependent on the state before it. A new observation *O(t)* is generated at every time instant and it is only dependent on the respective state at that time, *S(t)*. This can be written by (8),

$$P(O_t|O_1, O_2, \dots, O_{t-1}, S_1, S_2, \dots, S_t) = P(O_t|S_t) \quad (8)$$

A first order HMM (usually denoted as λ) can be completely characterized by the following [8] [9],

- A finite set of states **M**
- A finite (*discrete*) or infinite (*continuous*) set of observations **N**
- A state transition probability matrix ***A***

$$\boldsymbol{A} = \{a_{ij}|a_{ij} = P(S_t = j|S_{t-1} = i)\} \quad (9)$$

- A vector $\boldsymbol{\pi}$ of start probabilities

$$\boldsymbol{\pi} = \{\pi_i|\pi_i = P(S_1 = i)\} \quad (10)$$

- An observation emission probability distribution that characterizes each state

$$\{b_j(o_k)\big|b_j(o_k) = P(O_t = o_k|S_t = j)\} \quad (11)$$

In the case that there is a finite number of observations (i.e., the observations are discrete) then $b_j(o_k)$ is a discrete probability distribution and can thus be characterized by a probabilistic emission matrix ***B***, as given by (12),

$$\boldsymbol{B} = \{b_j(o_k)\big|b_j(o_k) = P(O_t = o_k|S_t = j)\} \quad (12)$$

Having a characterization of the HMM, $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$, three fundamental topics must be covered in for the model to be used to analyze and predict time series [7] [10],

1. Evaluation: given the model $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$, it is necessary to assess its ability to characterize the statistical properties of certain data using a quality measure $P(O|\lambda)$
2. Parameter estimation: consists in estimating the parameters of the model given an observation set
3. Decoding: consists in determining the most likely state sequence that, for a given model $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$, can generate the observation sequence

### *E. State of the Art*

Several different models have been developed over the years with the aim of predicting financial time series. Some of these models have even been incorporated into trading systems. This section reviews some of the most promising solutions, with special emphasis given to methods based on technical indicators and HMMs.

Hassan and Nath [11] propose a fusion model that combined an HMM, Artificial Neural Network (ANN), and Genetic Algorithm (GA) to forecast financial market behavior. The ANN was used to transform the actual observations, which were then fed into the HMM as an input vector. A GA tool was used to obtain the optimized initial parameter values of the HMM which, after the training, best fits with the transformed observation sequences. This process is executed until a possible best combination of ANN and optimized HMM is found. The system had the same performance as the Autoregressive Integrated Moving Average (ARIMA) model.

Bicego et al. [12] developed a novel approach for recognizing and forecasting brief sequences of time series relative to financial markets. The model explicitly and directly exploits the natural asymmetry present in the market by training two separate HMM models, one for the increase situation and one for the decrease.

Erlewin et al. [13] developed a multivariate HMM filtering process which analyses investment strategies. In particular, filtering techniques are used to aid an investor in his decision to allocate all of his investment fund to either growth or value stocks at a given time. For this purpose, the Russell 3000 Growth and the Russell 3000 Value indices were considered. The two indices were treated as a two-dimensional observation vector, with the mean levels and standard deviations at different time periods.

Hassan et al [11] introduce a new hybrid of HMM, Fuzzy Logic and Multi Objective Evolutionary Algorithm (MOEA) for building a fuzzy model to predict non-linear time series data. In this hybrid approach, the HMM´s log-likelihood score for each data pattern is used to rank the data and fuzzy rules are generated using the ranked data. A MOEA is used to find the range of trade-off solutions between the number of fuzzy rules and the prediction accuracy. The model was tested using 20 different stocks picked from the New York Stock Exchange (NYSE) and the Australian Security Exchange (ASX). The results demonstrate that the model is able to generate better results than other fuzzy models.

Gupta and Dhingra [10] present a Maximum a Posteriori (MAP) HMM approach for forecasting stock values for the next day using historical data. First, they consider the fractional change in stock value and the intra-day high and low values of the stock to train the continuous HMM. Then, the HMM is used to make a MAP decision over all the possible stock values for the next day.

Cheng and Li [14] propose an enhanced HMM-based forecasting model by developing a novel fuzzy smoothing method. A fuzzy time series is an ordered sequence with linguistic terms in time. The proposed model, referred to as *ps*HMM, can be generally applicable to the forecasting problem of fuzzy time series or traditional crisp time series. In the case of crisp time series, as happens with the stock market, a *fuzzification* process is needed. A smoothing method was developed to solve the zero-probability problem that differs from existing smoothing methods, which fail to consider the uncertainty that is characterized by fuzzy sets in the fuzzy time series. Basically, the proposed method searches for states (peaks) with higher probabilities than their neighboring states, and then shares a small portion of the probabilities with these neighbors since states are fuzzy in nature and should also be impacted by the probability mass. The model was tested using data from the Taiwan Weighted Stock Index (TWSI) and the National Association of Securities Dealers Automated Quotations (NASDAQ). The results suggest that, when compared to traditional HMM models, the psHMM provides statistically more accurate forecasting into the future. However, the need to incorporate a *fuzzification* process and smoothing brings additional complexity into the prediction problem.

Angelis and Paas [15] propose a framework to detect financial crises, pinpoint the end of a crisis in stock markets and support investment decision-making processes. This proposal is based on a HMM with 7 underlying states. By analysing weekly changes in the US stock market indexes over a period of 20 years, this study obtains an accurate detection of stable and turmoil periods and a probabilistic measure of switching between different stock market conditions.

Silva et al. [16] develop a portfolio composition system that tests investment models which incorporate a fundamental and technical approach, using financial ratios and technical indicators. A MOEA with two objectives, the return and the risk, is used to optimize the models. First, the best stocks are chosen based on the fundamental indicators and then, second, the technical indicators indicate when to buy or sell. Real world constraints such as transaction costs, long-only positions and quantity constraint for each asset were considered. This approach showed promising results, outperforming the benchmark index S&P 500.

Pinto et al. [17] construct a method to boost trading strategies performance using the Volatility Index (VIX) indicator together with a dual-objective evolutionary computation optimizer. A framework using a Multi-Objective GA (MOGA) in its core is used to optimize a set of trading strategies. The investigated framework is used to determine potential buy, sell or hold conditions in stock markets, aiming to yield high returns at a minimal risk. The VIX, indicators based on the VIX and other technical indicators are optimized to find the best investment strategy.

Alves, Neves & Horta [7] present a multi discrete HMM Approach. In this work three DHMMs are used simultaneously, each trained using a different sized window. Since the discrete version of the HMM is used, the authors recurred to a transformation of the input values into three distinct output values: rise, fall, and maintenance of the close price relatively to the previous day. The HMM was trained using the Baum-Welch Algorithm and tested using the Viterbi algorithm. Then, three training windows of 15, 30, and 90 days were chosen. As shown in figure 2, the different sized windows were able to capture different patterns. With the use of technical indicators and the three DHMMs, sub-models were developed that showed different characteristics and results between them. These models made use of technical indicators in order to trigger the different DHMMs. For instance, a change indicated by the RSI triggered the use of the 15 and 30 day DHMMs so that the model could better adapt to these market trend changes. The best models were combined, creating a supermodel able to adapt and respond to the demands of the Forex market. Each model outputs one of three signals depending whether the prediction is of an increase, decrease, or uncertainty of the market closing value. Subsequently, the most likely output is chosen from the five individual model outputs. The proposed method achieved good results as the final model recorded a gain of 26349 pips (price interest points) after 11 years (from 2002 to 2013) from the EUR/USD pair.

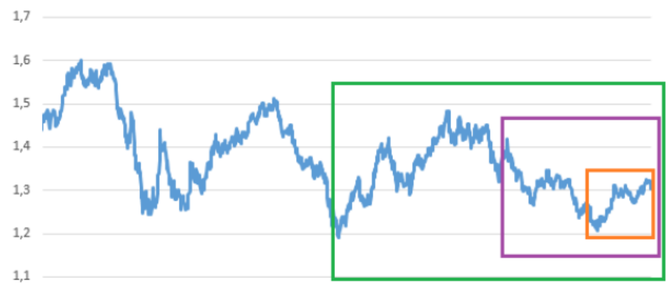


*Figure 2. 15, 30, and 90 day training windows for the DHMM [7]*

Tenyakov, Mamon & Davison [18] developed a zero delay HMM model that is able to aid investors in trading on the Forex market. The model immediately incorporates real-time data from fast trading environments. Using this data recursive filters for the Markov Chain are derived, and subsequently the model parameters are estimated. The various state of the art works are summarized in table I.

## III. System Architecture

### A. Introduction

. The overall model is incorporated into a trading algorithm which is capable of autonomously trading in the stock market. The ultimate result is a trading system which can predict stock market index price trends, and is thus able to generate significant returns while keeping the risk to acceptable levels. The discrete version of the HMM is used, as opposed to the continuous one. The continuous HMM attempts to predict the exact value of the next data point, which can be very challenging when dealing with financial time series. A small error in prediction can even give wrong trend information. On the other hand, the DHMM only concerns itself with the prediction of discrete values, which can translate into fall, maintenance, and rise of the price when dealing with financial time series. For the system developed along this work it suffices to predict price directions, and therefore the DHMM was chosen.

The system takes in the daily closing prices of stock market indices as input, and everyday new buy, sell, and hold decisions will be made. Any buy and/or sell orders will then be placed at market open. Specifically, shares of the S&P 500 index are considered. The large daily volume and the general robustness of companies listed in this index makes this financial asset an attractive choice

### B. Algorithm Architecture

A global overview of the system´s architecture is given by

*Table I. Comparison of the different state of the art works*

| Work | Model | Financial Application | Markets / assets tested | Period tested | Results |
|---|---|---|---|---|---|
| **Tenyakov, Mamon & Davison (2016) [11]** | Zero delay HMM | Forex Market Forecasting | JPY/GBP and JPY/USD pairs | July 2012 | Outperforms traditional HMM, and random strategy |
| **Alves, Neves & Horta (2015) [7]** | Multi Discrete HMM with technical indicators | Forex Market forecasting | EUR/USD pair | 2002-2013 | A gain of 26349 pips after 11 years |
| **Pinto, Neves & Horta (2015) [12]** | MOGA with technical indicators | Stock Market forecasting | NASDAQ | 2006-2014 | Return of higher than 10% annual |
| **Silva et al. (2014) [13]** | MOEA using financial ratios and technical indicators | Portfolio composition stock market | S&P 500 | 2010-2014 | Best chromosome achieved returns of 50.24% |
| **Angelis and Paas (2013) [14]** | HMM | Stock Market Index forecasting | S&P 500 | April 2010 – August 2010 | Net profit of 1.08% |
| **Gupta and Dhingra (2012) [10]** | HMM and MAP | Stock Market forecasting | 4 stocks from American companies | August 2002-September 2011 | Average MAP of 1.13, outperforming ARIMA, ANN, and HMM-fuzzy model |
| **Cheng & Li (2012) [15]** | HMM-based with a smoothing fuzzy model | Stock Market forecasting | TWSI and NASDAQ | January 2004-December 2006 | Statistically more accurate predictions than traditional HMM models |
| **Hassan et al. (2011) [16]** | MOEA and HMM-fuzzy model | Stock Market forecasting | NYSE and ASX | August 2007 | Better than other fuzzy models |
| **Erlewin et al. (2009) [17]** | HMM multivariate process | Stock Market Index forecasting | Russel 3000 Growth and Russel 3000 Value indices | 1995-2008 | Returns 21.4% higher than indices |
| **Bicego et al. (2008) [18]** | 2 separate HMMs | Stock Market Index forecasting | Dow Jones Index | November 1995 - February 2001 | Forecasting error rate of 49% |
| **Hassan et al. (2007) [19]** | A fusion of HMM, ANN, and GA | Stock Market Forecasting | 3 stocks from the IT sector | February 2003-January 2005 | Same performance as the ARIMA model |

the diagram in figure 3. As can be seen in the diagram, the algorithm interacts with two external entities: the user and the stock market. The user is responsible for selecting the investing period to be used by the algorithm. Note that the investing period does not need to have a pre-defined ending date by the user, as the algorithm could simply run until it is told to stop. The algorithm will query the stock market on a daily basis in order to fetch price data, and new buy and sell orders will be placed when appropriate. The inner blocks of the algorithm consist of the prediction core and two other modules. The data module is responsible for fetching the necessary data and processing it. The investment module is responsible for placing buy and sell orders in the stock market. The prediction core is responsible for predicting market trends and generating appropriate buy, sell, and hold signals. Essentially, it takes as input the processed data from the data module and outputs a signal to the investment module. This module can be further subdivided into two submodules: the DHMMs and the RSI. All of these blocks are further explained in the following sections.

## C. Prediction Core

The prediction core is the essence of the algorithm, having the task of predicting future trends. Depending on the prediction, one of four signals will be generated: Strong Buy, Hold, Sell, or Strong Sell. For this, a combination of three DHMMs, and the RSI, is used. More specifically, two daily DHMMs containing windows of 30 and 60 days are used, along with a weekly DHMM of 30 weeks. The RSI is used as the decision criteria to switch between the two daily DHMMs and the weekly DHMM. This is illustrated by figure 4.

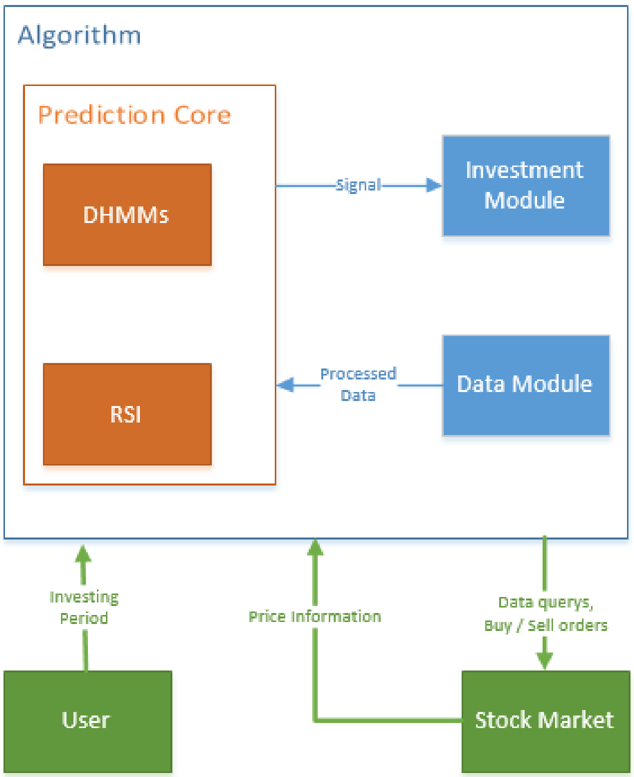


*Figure 3. Overall diagram of the algorithm*

As can be seen in the figure, when the value of the RSI crosses above 70 (and thus the stock index is overbought) the two daily DHMMs are used. In this scenario, since there will be a likely decrease in price, the algorithm can evaluate whether short positions should be taken on a daily basis. Once the value of the RSI drops below 30 (and thus the stock index is oversold) the prediction core switches to using the weekly DHMM. Since the price is likely to rise in this scenario, the weekly DHMM will put emphasis on long positions. Note that the weekly DHMM will still forecast price decreases, but such forecasts will be interpreted differently from those of the daily DHMMs.

## D. DHMM Architecture

The evaluation is done using the Forward-Backward algorithm and the window of discrete values received from the data module. Subsequently, the parameter estimation is done using the Baum-Welch algorithm. This means that the initial parameters of the DHMM can be generated randomly. When the parameters have finished being estimated, the decoding takes place using the Viterbi algorithm and the same window of discrete values.

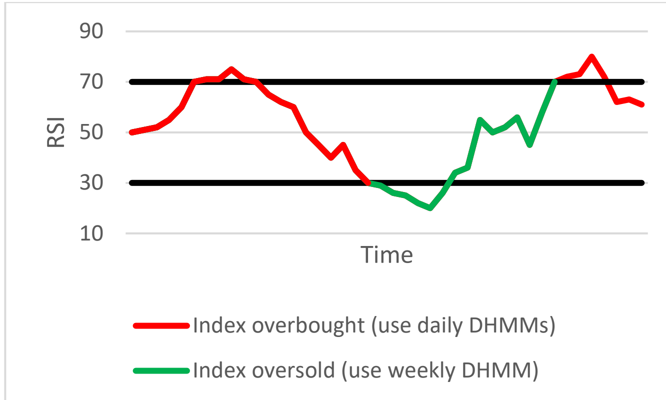


*Figure 4. Switching between the daily DHMMs and the weekly DHMM according to the RSI*

The Viterbi algorithm, along with a manipulation of equations, is the chosen option to forecast the direction of the market close price. The procedure implemented in this work, although developed from scratch, was based on [7], and can be described as follows

1. Obtain $\delta_t(i)$ using the Viterbi algorithm. $\delta_t(i)$ contains the most likely state sequence ending in state $s_i$ taking into account the model parameters $\lambda = \{\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}\}$ and the sequence of observations $O_1, O_2, \dots, O_t$, as given by (13),

$$\delta_t(i) = \max_{s_1, s_2, \dots, s_{t-1}} P(O_1, O_2, \dots, O_t, s_1, s_2, \dots, s_{t-1}, s_t = i|\lambda) \quad (13)$$

2. Determine the most probable state at time $t = T + 1$ by using $\delta_t$. This is achieved through a manipulation of the Viterbi algorithm equations. A new matrix $\varphi_i(O_{T+1})$ is created, which contains the probability of transitioning to state $s_i$ for each observation observed at time $T + 1$,

$$\varphi_i(O_{T+1}) = \max_i\{\delta_T(i)a_{ij}\}b_j(O_{T+1}) \quad (14)$$

3. Obtain $s_T^*$ from (15), the most probable state in T,

$$s_T^* = arg\max_j \delta_T(j) \quad (15)$$

4. Compute the new variable $\psi_t(j)$, the so-called backward pointer, which contains the optimal predecessor state for each $\delta_t(i)$.

$$\psi_t(j) = arg\max_i\{\delta_{t-1}(i)a_{ij}\} \quad (16)$$

5. Obtain the predecessor state using (17),

$$Predecessor = \psi_T(j = s_T^*) \quad (17)$$

6. Use the most likely predecessor state to extract the most probable observation from $\varphi_{predecessor}(O_{T+1})$ at time $T+1$,

$$Forecast = arg \max_i \ \varphi_{predecessor}(O_{T+1}) \quad (18)$$

From (18) it is possible to obtain the forecast for the next day.

### E. Investment Module

The investment module decides when to place buy and sell orders. This is done using the signal generated by the prediction core and the current state of the algorithm. The algorithm has three states, as illustrated by the state diagram of figure 6. As can be seen the three possible states are Out of the Market, Long position, and Short position. Initially, the algorithm is out of the market, and will stay that way until a Strong Buy, Strong Sell, or Sell signal is generated by the prediction core. A Strong Buy signal will set the state to long position. A Strong Sell signal will set the state to short position. A Sell signal will set the state to out of market in case the current state is long position, and it will set the state to short position otherwise. A Hold signal will not change the state. In case the investing period ends, the algorithm returns to its initial state, which is out of market.

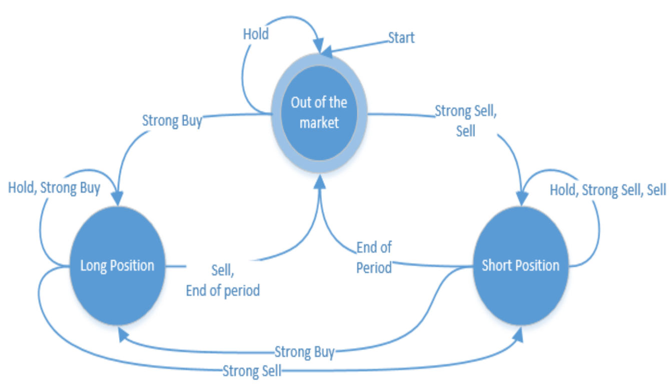


*Figure 6. State diagram of the algorithm*

## IV. Results

### A. Costs and constraints

In order to make the testing conditions as close as possible to reality, slippage and commission costs have been considered. The slippage can be calculated using (19).

$$Slippage = p\left(\frac{order\ size}{total\ volume}\right)^2 \quad (19)$$

Where the square of the ratio of the order size relatively to the total volume of the stock index is multiplied by p, the price impact constant. In this thesis p was set to 0.1, as this is considered a realistic value [20].

It is also important to take into consideration the order size, as one cannot trade more than the market volume. In fact, usually it is only possible to trade a fraction of the total volume. Thus, if the algorithm places an order that cannot be executed due to insufficient market volume, the order will remain open until it can either be processed or the market closes for the day. If the order is not able to be executed during the respective market session, it will be cancelled. For this thesis, a volume limit of 2.5% of the total volume traded every minute was imposed, a commonly accepted value [20].

Commissions are costs that an investor must take in order to access the market. For this thesis commissions of $0.0075 per share were used, with a $1 minimum cost per trade.

### B. Weekly Window size case study

Firstly, different sized weekly windows were tested. Price values of the S&P 500 index were used over a period of 6 years from 11/01/2003 to 11/01/2009.

*Table II. Weekly window size comparison*

| Window Size | ROR | Error | Sharpe |
|---|---|---|---|
| **5** | -27,5% | 52% | -0,39 |
| **10** | 13,6% | 48% | 0,23 |
| **15** | 26,8% | 48% | 0,42 |
| **20** | 15,7% | 47% | 0,25 |
| **25** | 47,1% | 47% | 0,55 |
| **30** | 50,4% | 47% | 0,59 |
| **35** | 14,8% | 47% | 0,23 |

As can be seen by table II, the highest ROR was achieved by the 30 week window at 50.4%, followed by the 25 and 15 week windows with 47.1% and 26.8%, respectively. It can also be noted that for windows smaller than 10 and bigger than 30 weeks the ROR decreases sharply. As for the Sharpe ratio, the values were quite low for the most part, ranging from -0.39 to 0.59.

### C. Multi Daily Window Sizes case study

Having analysed the best training windows to use with single daily DHMM systems, multi daily DHMMs with different training windows were combined with the goal of improving the performance of the system. More specifically, double DHMM systems were considered from the 11th of January of 2003 to the 11th of August of 2004. These double DHMM systems only generated buy or sell signals in case the two DHMMs produced a unanimous forecast, otherwise the system would take no action. The results are shown in figure 7. The figure is color-coded, having green as the best performance, yellow as intermediate, and red as the worst performance. Note that the case where the window sizes of the two DHMMs are the same is identical to having a single DHMM system. As can be seen, the combination that obtained the highest ROR (30.1%) was the combination of 30 and 60 day window DHMMs. Although this result outperforms the original weekly DHMM for this specific time period, it underperforms when compared to the best single daily DHMM system (the 60 day window).

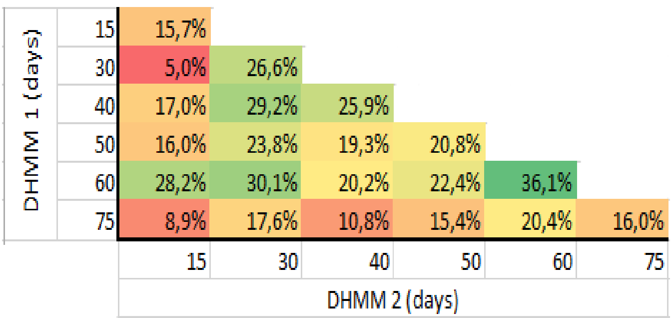


*Figure 5. ROR comparison of the different DHMM combinations*

### D. Technical Indicator Case Study

Having found out the best weekly, single daily, and double daily DHMM systems, it was time to fuse them into a single algorithm. The ultimate goal was to create an overall system that could better identify both short and long term trends. In order to do this, the momentum technical indicator RSI was considered, as it can help identify trend shifts. The criteria used by the RSI to select the different DHMMs is depicted by figure 6.

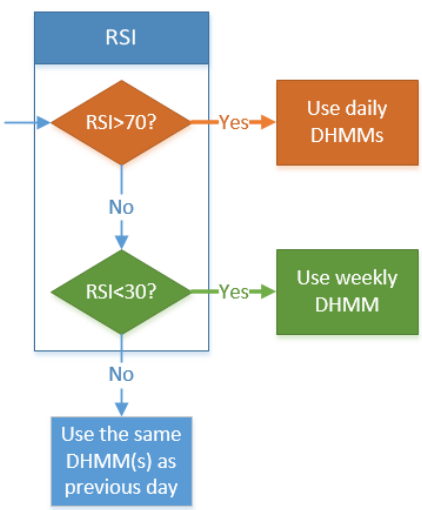


*Figure 6. Decision criteria for the RSI*

The results are presented in table III. It can be seen that the best performing overall system, in terms of ROR, is a fusion of the RSI, the 30 week DHMM, and the double DHMM system with a 30 and a 60 day window combination. This system achieved a ROR of 81.8%, which is significantly better than the system originally considered. The same is true for the Sharpe ratio, which was 0.82. The error rate was of 47%.

### E. Observation types case study

It was also important to determine the number and type of

*Table III. Fusion of the 30 week DHMM with daily DHMM(s) using the RSI*

| Technical Indicator | Daily DHMM(s) | ROR | Error | Sharpe |
|---|---|---|---|---|
| **RSI** | 60 | 76.0% | 47% | 0.85 |
| | 30, 60 | 81.8% | 47% | 0.82 |

observations to be used by the DHMMs. Traditionally, the works involving discrete financial time series prediction use three types of observations: drop, maintenance, and rise of the price relatively to the previous day. In this case study, four different approaches were considered, consisting of either three or five observations. In the case of three observations, the standard fall, maintenance, and rise of the price were considered. In the case of five observations, a very strong increase and very strong decrease in price were also considered, in case the fall or rise was over 2%. In addition to this, two types of maintenance were considered: strict and weak. A strict maintenance is a price change of exactly zero. A weak maintenance is a price change no greater than 0.5%. These four approaches were used by the best system developed in the previous case study (a fusion of the RSI, the 30 week DHMM, and the double DHMM system with a 30 and a 60 day window). The results are presented in table IV.

*Table IV. Different observation approaches used by the algorithm and corresponding results*

| Observation types | ROR | Error |
|---|---|---|
| **Rise, strict maintenance, decrease** | 80.3% | 47% |
| **Rise, weak maintenance, decrease** | 37.2% | 47% |
| **Strong rise, rise, strict maintenance, decrease, strong decrease** | 0.1% | 48% |
| **Strong rise, rise, weak maintenance, decrease, strong decrease** | 11.1% | 48% |

One Upon inspection of table IV one can see that approach 1 delivers the best results. The type of observations of this approach fall, strict maintenance, and rise of the price. These results, however, are worse than the best results achieved by the previous case study. These findings suggest that incorporating a “strict maintenance” observation into a trading algorithm deteriorates the quality of the results, perhaps due to having an extra observation that seldom happens in the real world [7]. The weak maintenance approach also struggled to achieve significant ROR. The five observation system also proved disappointing, achieving low ROR when compared to the other

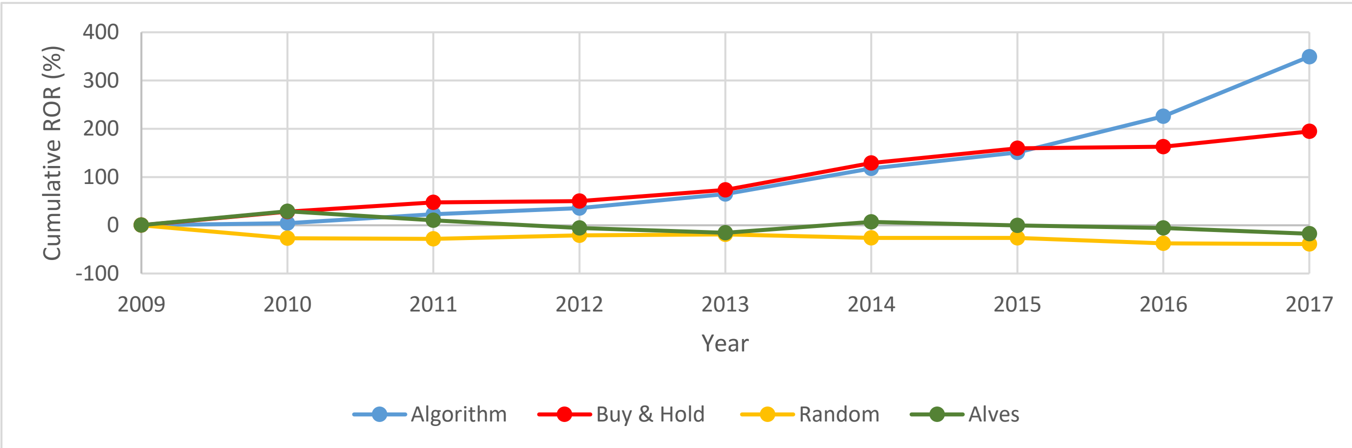


*Figure 8. Cumulative ROR of the four approaches over the testing period*

observation systems. The error rate for these cases was also higher, at 48%. This may be due to the fact that the higher number of observations increases the complexity of the observation sequence, making it more difficult to produce accurate forecasts using the DHMMs.

### *F. Testing*

Once the training was complete, it was time to test the algorithm using out of sample data. To do this, S&P 500 price data over a period of eight years from 12/01/2009 to 12/01/2017 was considered. The algorithm was then compared to a state of the art solution, which was replicated during this thesis, and two other investment strategies: the Buy & Hold and a purely random strategy. The chosen state of the art solution was the system developed by Alves et. al [7] since this particular solution is the most similar to the proposed algorithm. The purely random strategy randomly places buy and sell orders every day. Figure 7 shows the ROR of the different approaches obtained in each year of the testing period.

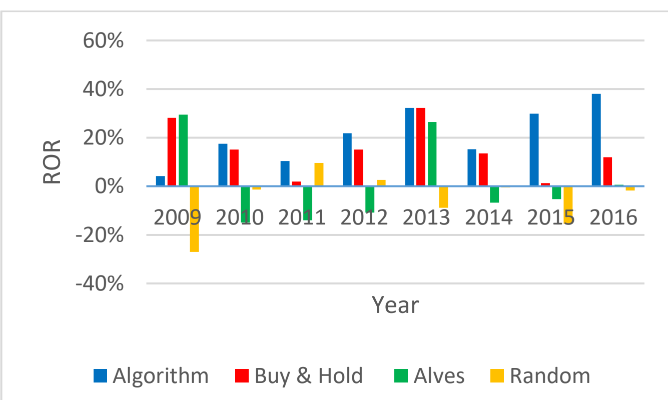


*Figure 7. Testing results of the different approaches*

Upon inspection of the bar chart of figure 7 one can conclude that, with the exception of 2009 and 2013, the algorithm outperformed all other approaches. This is especially evident in the last two years. It is also noteworthy that the algorithm makes a profit every year. The Buy & Hold strategy also performed fairly well, since there was a long term upwards tendency of the S&P 500 index. The approach developed by Alves et. al was volatile, outperforming all other approaches in the beginning, but then falling behind. The random strategy's returns were also volatile, and in the majority of the years (5 in total) it suffered losses. The evolution of the cumulative ROR over the testing period is given by the graph in figure 8.

*Table V. Comparison of the algorithm's performance against a state of the art solution and two investment strategies*

| Approach | Testing Period ROR | Average Annual ROR | Error | Sharpe |
|---|---|---|---|---|
| **Algorithm** | 356% | 21% | 46% | 1.28 |
| **Buy & Hold** | 199% | 15% | [1] | 0.87 |
| **Alves et. al [7]** | -17% | 1% | 49% | -0.05 |
| **Random** | -40% | -5% | 50% | -0.37 |

As can be seen by table V, the algorithm outperforms the other approaches both in terms of the cumulative ROR and the average annual ROR, which were 356% and 21%, respectively. In addition, the algorithm achieved an error rate of 46%, which was the lowest out of all the approaches. Finally, the algorithm also achieved the best Sharpe ratio value, which was 1.28. This is considered a good risk-adjusted return, and thus one can conclude that the algorithm's strategy's hefty profits are not simply due to overly high exposure to risk. The second best performance was that of the Buy & Hold strategy[1], with a cumulative ROR of 199%, average annual ROR of 15%, and a Sharpe value of 0.87. After came the solution developed by Alves et.al. with a cumulative ROR of -17%, average annual ROR of 1%, 49% error rate, and a Sharpe value of -0.05. It is noteworthy that the poor performance exhibited by this approach may be due to the fact that the system was optimized for the Forex market, which differs from the stock market.

[1] The error rate of the Buy & Hold strategy is meaningless since no predictions are made

Nevertheless, it is still significantly better than applying a purely random strategy, which achieved a cumulative ROR of -40%, average annual ROR of -5%, 50% error rate, and a Sharpe value of -0.37.

## V. Conclusions

This work presents a novel approach to stock market index algorithmic trading. It does so by predicting stock market index price trends using the discrete Hidden Markov Model and the technical indicator RSI. The financial time series was transformed into a discrete sequence of two values: rise and fall of the price in relation to the previous trading day.

One of the great innovations of this method is the combination of DHMMs with windows of different time frequencies: weekly and daily. In case the stock index price is overbought, as identified by the RSI, two daily DHMMs are used in order to profit from the likely drop in price in the short term. When the index is oversold, the system switches to using a weekly DHMM in order to take advantage of the longer term trends. Using the weekly version of the DHMM mitigates shorter term noise, allowing the system to focus on longer term trends. Tests using price data from the S&P 500 index were conducted over a period of eight years from January 2009 to January 2017. The results demonstrated the validity of the proposed solution, as it outperformed the Buy & Hold strategy, a methodology proposed by [7], and a purely random strategy. Therefore, the main conclusion that can be drawn by analysing the results is that implementing the proposed solution turned out to be a great choice. Although the testing period included times of uncertainty and volatile behaviour in the stock market, the algorithm was able to achieve profits every year.

## VI. References


[1] J. Patel, S. Shah, P. Thakkar and K. Kotecha, "Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques," *Expert Systems with Applications*, 2014.

[2] S. B. Achelis, "Technical Analysis From A-To-Z," Vision Books, 2000.

[3] B. Graham and J. Zweig, The Intelligent Investor,, Rev Sub Edition, HarperBusiness, 2006.

[4] K.-Y. Kwon and J. R. Kish, "Technical trading strategies and return predictability: NYSE," in *Applied Financial Economics*, 2002.

[5] A. Gunasekarage and D. M. Power, "The profitability of moving average trading rules in South Asian stock markets.," in *Emerging Markets Review 2*, 2001, pp. 17-33.

[6] T. T.-L. Chong and W. K. Ng, “Technical analysis and the London stock exchange: Testing the MACD and RSI rules using the FT30,” in *Appl. Econ. Lett.*, 2008, pp. 1111-1114.

[7] J. Alves, R. Neves and N. Horta, "Forex Market Prediction Using Multi Discrete HMM Models," 2015.

[8] L. E. Baum and T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains," *The Annals of Mathematical Statistics*, pp. 1554-1563, 1966.

[9] D. Ramage, "Hidden Markov Models Fundamentals," 2007.

[10] A. Gupta and B. Dhingra, "Stock Market Prediction Using Hidden Markov Models," 2012.

[11] A. Tenyakov, R. Mamon and M. Davison, "Modelling high-frequency FX rate dynamics: A zero-delay multi-dimensional HMM-based approach," *Knowledge-Based Systems*, vol. 101, pp. 142-155, 2016.

[12] J. Pinto, R. Neves and N. Horta, "Boosting Trading Strategies performance using VIX indicator together with a dual-objective Evolutionary Computation optimizer," *Expert Systems with Applications*, vol. 42, 2015.

[13] A. Silva, R. Neves and N. Horta, "A hybrid approach to portfolio composition based on fundamental and technical indicators," *Expert Systems with Applications*, vol. 42, pp. 2036-2048, 2015.

[14] L. D. Angelis and L. J. Paas, "A dynamic analysis of stock markets using a hidden Markov model," *Journal of Applied Statistics*, 2013.

[15] Y.-C. Cheng and S.-T. Li, "Fuzzy Time Series Forecasting With a Probabilistic Smoothing Hidden Markov Model," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 2, 2012.

[16] M. Hassan, B. Nath, M. Kirley and J. Kamruzzaman, "A hybrid of multiobjective Evolutionary Algorithm and HMM-Fuzzy model for time series prediction," *Neurocomputing*, vol. 81, pp. 1-11, 2012.

[17] C. Erlewein, R. Mamon and M. Davison, "An examination of HMM-based investment strategies for asset allocation," *Applied Stochastic Models in Business and Industry*, vol. 27, pp. 204-221, 2009.

[18] M. Bicego, E. Grosso and E. Otranto, "A Hidden Markov Model Approach to Classify and Predict the Sign of Financial Local Trends," *Structural, Syntactic, and Statistical Pattern Recognition*, vol. 5342, pp. 852-861, 2008.

[19] M. R. Hassan, B. Nath and M. Kirley, "A fusion model of HMM, ANN and GA for stock market forecasting," *Expert Systems with applications*, vol. 33, pp. 171-180, 2007.

[20] A. Frino and T. Oetomo, "Slippage in futures markets: Evidence from the Sydney Futures Exchange," *Journal of Futures Markets*, vol. 25, no. 12, pp. 1129-1146, 2005.

[21] G. A. Fink, "Markov Models for Pattern Recognition," in *From Theory to Applications*, Dortmund, Springer, 1998, pp. 61-92.