Quantitative Estimation of Market Sentiment: a discussion of two alternatives

FILIPPO NERI
University of Malta
Department of Computer Information Systems
Msida
MALTA
filipponeri@yahoo.com

Abstract: Modeling and estimating the time series (the value in time) of assets traded in real financial markets is an intriguing challenge that has attracted researchers from many fields including Economics, Statistics, and more recently Computer Science thanks to the common availability of computational power that made possible the investigation of computational or simulative methods for modeling financial time series. In this paper, we discuss a computational simulation technique based on agent based modeling and learning to closely approximate the SP500 and DJIA indexes over many periods and under several experimental set ups. According to our modeling approach, the value in time of a financial asset emerges as an aggregate result of several independent investment decisions (to buy, to sell, or to hold) during a short period of time. We can therefore reproduce the process of value formation by computationally simulating the community of agents-investors. We will compare our system's performances with respect to other approaches on the same time series to provide empirical data about the effectiveness of different computational techniques. The main finding emerging from our work is that a simple architecture for a simulator combining agent based modeling and learning produces close approximations for the SP500 and DJIA time series. The approximation results are comparable to those observed when evaluating prediction rules learned by neural networks or particle swarm optimization. An additional characteristic of our modeling approach is that it can provide insights about the contribution of each agent to the process of value formation for a financial asset.

Key–Words: Agent based modeling, simulated annealing, financial markets, prediction of the SP500 and DJIA time series.

1 Introduction

Financial markets are a complex research challenge for scientists working Economics, Statistics, and Computer Science. Recently the common availability of computational power has supported the development of novel research area like Agent Based Modeling (ABM) and Agent-based Computational Economics (ACE). The interests of the two research areas are somewhat overlapping: ABM focuses on the study of systems, without any domain restrictions, by developing computational models of their elementary components and their relative interactions [1, 2, 3, 4, 5, 6, 7, 8, 3], while ACE focuses on studying economic processes modeled as societies of interacting agents [9, 10, 11]. Both researchers in ACE and ABM consider financial markets a difficult and intriguing testbed for investigating if and how computational agent based models can reproduce by simulation an observed market behavior [2, 9]. Research on this topic is actively challenged because even if a variety of approaches to agent based modeling of financial markets have been proposed, financial markets still remain substantially unpredictable. In this paper, we will focus our attention on developing an agent based model for financial markets able to learn the model of the financial time series at hand.

Research on modeling financial markets usually focuses on: either evaluating or learning trading strategies for some given financial instruments (commodities, bonds, shares, derivatives, etc.); or developing artificial markets, whose internal dynamics can be controlled, in order to study the formation of notable phenomena (price bubble formation, for instance) that are observable in real markets; or, finally, modeling the time series of the values/prices for some financial assets. With the only intent to provide some examples of the listed research approaches and without claiming to be exhaustive, we briefly mention some papers that fall into the previous categorization.

In [12], a society of trading agents uses genetic algorithms to learn trading strategies. Genetic algorithms are used to learn which of the many input

parameters, derived from financial technical analysis [13] (n-days moving averages, oscillators, n-days relative strength, and so on), are useful in identifying profitable investment decisions. Individual learning and societal learning (selection of the best traders in the society) are exploited. As the research focuses on trading, the paper does not report information about how well the trading agents can approximate the financial time series that have been traded, so there is no way to know how good is their system in estimating or tracking a specific asset.

A similar research approach is followed in [14], where a multi agent system based on learning classifiers can discover trading strategies that outperform simple investing strategies like "buy and hold" or "keeping the money in a saving account". With a similar philosophy, in [15] neural networks are used to learn trading strategies. In these works, as well, no information is given on how well the system can forecast the value of a financial asset.

Another interesting work in this category consists of using boosting and alternating decision tree to learn trading strategies that combined with a risk management system may provide a complete self adaptive automatic trading system [16]. The paper describes a complex trading system but yet it does not provided data on how well some asset time series are approximated.

In the second category of works, artificial markets are used to investigate how different interaction policies among the agents would produce phenomena that are also observable in real markets. In [17], an agent based artificial market allows to experiment with investment strategies, guiding each single agent, in order to observe how the value for a financial asset changes. One of their findings is that over confident investors contribute in making the market efficient[1]. In [18], an artificial market, the Santa Fe Institute (SFI) Artificial Stock Market, is used to empirically test hypothesis about how investors adapt their expectations in time and how this adjustment affect the rest of traders. As a last example of this type of research, in [11], investors, modeled into an agent based market, use either a trading strategy based on trend expectation for a given stock or a trading strategy based on basic fundamental analysis of the stock value. The resulting artificial market displays volatility clustering phenomena similar to those observed in real financial markets. The works following this research focus, report no application of their systems to the approximation of time series of real financial assets.

Finally the last group of works aim to closely approximate the time series of some real assets. This approach involves evaluating how good some equation based models or computational models are in estimating some given target time series. The model can be either developed by the researcher by hand or automatically build by exploiting maybe machine learning algorithms. Some instances of this approach are: [3] where an agent based market try to explain the behavior of the SP500 index in a short period of time; the exploited model is manually built. In [19], neural networks and genetic algorithms are used to learn and predict the time series of the SP500 and of the DJIA indexes. While in [20], a manually determined equation model, based on inflation indexes, is used to explain the behavior of the price time series for some stocks.

The research reported in this paper belong to the last category of investigation: in this paper, we will show how a Learning Financial Agent Based Simulator (L-FABS) can approximate the time series of the SP500 and DJIA indexes over many periods of time. We will study the behavior of the system under several experimental conditions and we will compare its performances with respect to other approaches on the same time series to provide empirical data about the effectiveness of different computational techniques in modeling financial time series.

The rest of the paper is organized as follows: Section 2 describes some the basic economics facts we have based the architecture of our system on; in Section 3, we describe the basic component of our system: a simple financial agent; in Sections 4 and 5, we describe the architecture of our financial agent based simulator and we show how learning can be exploited to approximate financial time series; in Section 6, an example of a learned model is explained; in Section 7, we discuss the differences among learning a deep and a shallow model; in Section 8, the experimental findings are commented, before drawing our conclusions, in Section 9.

2 Basic Economics Facts Considered in Designing L-FABS

When designing our system L-FABS (Learning Financial Agent Based Simulator) our intention was to keep its architecture as simple as possible in order to facilitate the interpretation of its results. We have achieve so by designing its architecture according to the following economics facts.

[1] A market is said efficient if any change in the fundamental drivers of the value formation process of any asset are quickly and correctly incorporated into the asset price. In plain words: the arrival of any good or bad news affecting an asset value produces an appropriate and rapid variation in the trading price of the asset.

We believe that any investment decision of each individual depends on two components:
a) his/her propensity to take some risks today, by buying a financial asset, in exchange for a future uncertain reward, when selling the asset. For instance: should one invest his/her savings in Google shares for the next 10 years hoping in an awesome return but accepting the risk of losing everything in case of the company's default, or is it better to put the savings into government bonds which will not default but will return a relatively modest income even after 10 years? In the end the investment decision is affected by an investor's propensity to risk something for an uncertain future reward. We will model the risk/reward profile of an investor using a risk/reward propensity rate in the system.

b) the common consensus about the future behavior of the asset. If it is true that each investor takes investment decisions independently from the others, it is also true that public knowledge about the economic outlook, as diffused for instance by financial news and economic reports, will influence the investment decision of every one operating in the market. In details, if the economic outlook looks negative, people will tend to sell some of their stocks and to buy instead government bonds or physical gold to deal with a possible drop in stock prices due to reduced future profits. If the economic future looks positive, investors tend do the opposite. In the paper, we call market sentiment, or just sentiment, the economy outlook and model it accordingly in our system.

Let's move now to explaining the architecture of our system.

3 The Simple Financial Agent Based Simulator

In this section, we will present a Simple Financial Agent Based Simulator, S-FABS, that we will employ to simulate an individual investor. S-FABS implements the simple decision making process underlying the investment decision of buying, selling or holding an asset. A Financial Agent will decide to buy some asset with probability P(X<BuyThreshold), it will hold to its assets with probability P(BuyThreshold<X<SellThreshold), and it will sell some of its assets with probability P(SellThreshold<X). As it can be seen in the algorithm, the probabilities are dependent on the current level of sentiment about the economy. In fact the probability to buy assets increases along with the sentiment, while the probability to sell assets decreases when the sentiment is raising. Finally, the amount of assets to be bought or sold is set to 2% of the total assets of the investors. The value 2% has been chosen as it represents twice the average daily variation of the timeseries we have investigated. We verified empirically in earlier works [4, 5] that it provides reasonable simulation results and has been kept constant since then. It might be changed in case the time series under investigation displays an average daily variation higher than 1%.

Given a Financial Agent, a simulation of the financial market can be obtained by creating several Financial Agents, each one with its own status in terms of own assets, invested assets and risk/reward propensity, and then performing a sequence of investment rounds where each agent decides if buying, selling, or holding taking into account the current Sentiment value. At the end of each round, it is possible to measure the percentage of invested assets, this percentage can then be used as an estimated for one data point of the target time serie. If the simulation is repeated for a number of rounds, S-FABS can output a estimate for a full time serie. S-FABS take as input the vector of risk/rewards propensity rates for each Financial Agent. During each round, the risk/rewards propensity rates are used in combination with the current value of the economic sentiment by each Financial Agent. We note that our approach does not impose any restriction on how the market mood is determined. For our research we implement the Sentiment function as follows to keep it as simple and as general as possible:

```
function Sentiment(time, mavgDays)
  begin=time-1
  end=time-mavgDays
  if (MAVG(PredictedData, begin, end) <
      MAVG(RealData, begin, end))
      then return(0.65)
      else return(0.30)
```

The variables RealData and PredictedData give access to the time series of the real and predicted values. The values 0.65 and 0.30 are constants empirically determined in an earlier work [21] and never modified since. According to our definition of the Sentiment function, the outlook of the real market is considered bullish if the moving average[2] (MAVG) of the predicted time serie is lower than the moving average of the real data. If this is the case, the Sentiment value is

[2]The MAVG function is defined as:

$$\text{MAVG(index,t,n)} = \frac{\sum_{k=0}^{n-1} index(t-k)}{n}$$

set to an high value so that a bullish mood is communicated to the Financial Agents. The opposite happens if the predictions of the system have been higher than the real data.

Following our philosophy of keeping things simple, we decided to exploit only two parameter settings when invoking the Sentiment function: in the first setting, the Sentiment function is called as $Sentiment(round, 1)$: this means that only the latest available real asset value is used to estimate the market mood. With the second parameter setting, the function is called as $Sentiment(round, 5)$: the average of the latest week of real index data is used when estimating the market mood. We will identify in the following the two options in estimating the market sentiment as S1 or S5 respectively. We are aware that there are other alternatives for trying to identify the market outlook including, for instance, employing technical indicators from financial technical analysis [13], or using complex text analysis systems able to digest financial news appearing over the Web as proposed, for instance, in [22].

A final point about the Financial Agents in S-FABS. In our study, we employ four types or classes of Financial Agents to capture the richness in investment decisions and in size of financial transactions that occur in real financial markets. The four types of investors we try to model are: individual investors (and the likes), banks (and the likes), hedge funds (and the likes), and central banks (and the likes). They differ in term of the size of the assets they can invest in financial markets and for their risk/reward appetite. In addition, their numerical presence is also different. Here are the values we used during our simulations:

Investor type	Total Assets (in millions)	Percentage
Individual	0.1	30
Funds	100	20
Banks	1000	49
Central Banks	10000	1

Note that the numbers in the table are only a rule-of-thumb approximation for the average asset size and number of investors operating globally on the financial markets. Specific reasons for choosing four types of investors and for setting their parameters are: empirical findings from previous works[21], common view about who operates in the markets, the desire to keep the model simple while preserving its intrinsic diversity, and personal conversations with investment managers which provided useful insights about the size of assets available for investment to different type of investors.

As the reader may have certainly noted, we did not list the risk/reward propensity rate for each investor type as this parameter is either manually set up by the experimenter or is object of learning as we explain in the following section. All the experiments reported in this paper have been made after learning the risk/rewards propensity rates from the data.

4 Learning in S-FABS

In this section, we describe how a learning capability can be added to S-FABS so that it may find the best model for a given timeserie. In particular, learning in S-FABS consists of finding the vector of risk/reward propensity rates that approximates a given time serie with a minimum error. As in our agent based model, we have four types of agents, the vector of risk/reward propensity rates, that we are interested in learning, contains four values. This learning setting allows for combining S-FABS, the simulation engine, with any of the many machine learning algorithms able to find a vector of values that minimizes a given error function. Examples of suitable machine learning algorithms include genetic algorithms [23], decision trees [24], neural networks [25], simulated annealing [26] and many others. Given the possibility to select any of these learning methods, we decided to use simulated annealing because the author has research experience in evolutionary computation and he is aware that evolutionary learning produces good results [8]. For the error function to be minimized, we need to select one that can evaluate how well two time series are similar: the Mean Squared Error and the Mean Average Percentage Error will do the work. They are commonly used in Statistics when two data samplings have to be compared. Just as a reminder, the Mean Squared Error is defined as:

$$MSE(X,Y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2$$

and the Mean Average Percentage Error is defined as:

$$MAPE(X,Y) = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{x_i - y_i}{x_i}\right|$$

Given two time series X and Y, the lower the values for MSE and MAPE, the closer the two are. MSE provide an absolute measure for the error while MAPE returns the error measured in percentage terms. The squared root of MSE is known as the Standard Error. In our research, the MSE or Standard Error are useful error measures when comparing time series for the same target time series as they have the same data range and a measure of error in absolute terms makes sense. Instead MAPE, that returns an error expressed as a percentage, is more suitable when comparing time series

having different target data as they usually have different value ranges and an absolute error measure would not be helpful in determining their relative accuracy.

In the next section, we will briefly describe how Simulated Annealing works and we will show how Simulated Annealing and S-FABS could be integrated to create a learning system which we will call Learning-FABS or for short L-FABS.

5 Combining Simulated Annealing and S-FABS

Simulated annealing[3] [26] is a searching algorithm that iteratively search for the minimum of a given function E (Energy or Error function) by generating at each step random candidate points for the minimum in the proximity of the current candidate minimum point which represents the "center" of the exploration. At the end of each step, the Simulated Annealing may move the "center" of its exploration at a newly generated point with a given probability.

Simulated Annealing, as a search process, can also be understood as walking along a random trajectory from one point in the domain of the energy function E to another point in search for the function minimum while the overall Temperature of the process is lowering.

All that said, a possible template for the Simulated Annealing algorithm is:

$SimulatedAnnealing()$
  $s = randomstate();$
  $e = E(s)$
  $T = 1; \alpha = 0.95$
  $k = 0; kmax = 100;$ emax=0
  $while(k < kmax) and (e > emax) do$
    $s_n = neighbour(s)$
    $e_n = E(s_n)$
    $if(e_n < e) then$
      $s_{best} = s_n; e_{best} = e_n$
    $endif$
    $if(random(1) < P(e_n, e, T)) then$
      $s = s_n; e = e_n$
    $endif$
    $k = k + 1$
    $T = T \times \alpha$
  $endwhile$
  $return(best_{state})$

Simulated Annealing starts by setting the current state s to a random state, the energy for the state is calculated and stored in e. Now the main cycle performs the following tasks: first a novel point s_n in the proximity of s is determined, its energy is calculated and stored in e_n. If the energy of the new state e_n is lower than e, that of the current state, then the new state and its energy value are saved in the s_{best} and e_{best} variables. Then a decision is made if moving the current state s to the new state s_n and thus moving the search to another region with probability dependent on the function $P(e_n, e, T)$. The probability in the test introduces a stochastic factor in the direction of the trajectory followed to find the minimum of the energy function E and helps the Simulated Annealing in escaping local minima.

In order to integrate the described Simulated Annealing schema in our system, the detailed definition for the following elements has to be given. A state s represent a vector of four numbers, in the (0,1) interval, representing the risk/reward propensity rates of the four agent types in S-FABS.

The *energy function E* to be minimized can be either defined as the MSE, the Standard Error or the MAPE. Note that evaluating $E(s')$ requires to run a simulation with risk/reward propensity rates set at s': Simple-FABS(s').

The $neighbor(s)$ function has to return a state in proximity of its argument. We implemented it as a function that for each value of the vector s randomly adds or decreases it by 0.1 $\times$ random(1).

Coming to $P(e_n, e, T)$, if the energy of the new state, e_n, is lower than the energy of the current state, e, than returns 1; else returns $\exp(\frac{e-e_n}{T})$. In other terms, even if the new state has higher energy than the current state, there is a non zero probability that the simulated annealing will move its exploration center to it. The parameters α and T are set to 0.95 and to 1 according to Kirkpatrick et al. [26].

Finally we recall that adapting the Simulated Annealing template to a specific learning setting is similar to the need of defining domain specific fitness function and cross over operators when using genetic algorithms as shown for instance in [23].

6 Example of a financial model learned by L-FABS

Before reporting about the experimental evaluation of L-FABS, we would like to show an example of

[3] Simulated Annealing owns its name to the imitation, from a computational perspective, of the physical phenomena of heating and liquefying a material (perhaps a metal) and then cooling it down in a controlled way in order to allow for the molecules to move slowly from higher energy states to lower energy states in order to achieve a crystallization without defects of the annealed material.

a learned model for a given time series. As already said, given a financial time series, L-FABS can learn the vector of risk/reward propensity rates that better approximate the original time series when they are given in input to S-FABS. The vector of risk/reward propensity rates then completely identify, in our setting, the financial model for the given time series. Here is an example of a such a model:

Investor type	Risk/Reward propensity
Individual	0.22
Banks	0.565
Funds	0.237
Central Banks	0.513

The shown rates are just an example of a financial model learned by L-FABS. This particular model would imply that the behavior of the time series can be simulated or explained by assuming that the agents representing private Banks and Central Banks tend to be more active investors in the financial market (they have a propensity to buy of over 50%) than individual investors or funds. Also the example shows how the model learned by L-FABS can used not only for simulation and prediction purposes but it can also open up a qualitative discussion about how the various types of investors contribute in producing the observed behavior in the time serie. However being this discussion of qualitative nature, we will not investigate it any further in this work.

7 Shallow and Deep Learning of a Time serie

In this section, we want to spend few words on the difference between deep and shallow models of a domain [6] in order to put in the right context the following experimental findings. Deep models of a domain are explains and are based on structural or causal relationships among domain elements which are not directly observable in the data but whose peculiar interactions generate the data itself. Shallow models, instead, are based on the "superficial" recognition of patterns in the data and do not contain information about the underlying mechanics that generated the patterns. Our system, L-FABS, predicts a time series by exploiting a functional approximation of how a financial market works or, said in another terms, by simulating the investment decisions made by many individual investors. Therefore L-FABS learns and exploits a deep model of the domain. This is different from the most common machine learning systems, like decision trees or neural network, that acquire classification rules based on pattern recognition on the latest values of the target financial time serie.

In the following experimentation, we will test the prediction capability of the deep model learned by L-FABS by using or not using all the knowledge about the real values of the time series up to the current time t when forecasting the future values from time t+1 and ahead.

The fact that L-FABS exploits a deep model of the domain when making predictions, opens up two learning scenarios to experiment with that are not available to shallow model learners:
$FullKnowledge(FK)$ - all the information in the time serie, up to the current time, is exploited during learning. This implies that, at the beginning of each simulation round and if needed, the correct value of the time serie at time t-1 is set as the value of the estimated time serie at time t-1.
$LimitedKnowledge(LK)$ - only the starting point of the time serie, t=0, is given to L-FABS in order to initialize the simulation, but all the rest of the real time serie values are not known. In this case, the simulation model in L-FABS will move from one predicted value for the time serie to the next without using the correct value of the time serie at time t-1 in order to estimate a value for time t.

Learning scenario Full Knowledge corresponds to the learning setting used by other learning approaches when predicting a time series. Learning scenario Limited Knowledge, instead, produces forecasts based on a "free" evolution of the modeled market without any correction during the simulation. Also as learning scenario Limited Knowledge uses less information than Full Knowledge, it corresponds to a more difficult learning task.

We are considering the Limited Knowledge scenario because this learning setting can provide insights about the "robustness" of the deep model acquired by L-FABS. By robustness we mean the feature of the model to have intrinsically captured the law of evolution in time of the time serie which can be measured by the maintenance of accuracy in forecasts as the system move far from the last known point of origin.

8 Empirical evaluation of L-FABS

In order to empirically evaluate L-FABS, we need to select some financial time series as datasets to work with. As usual with machine learning systems, we will train L-FABS on a part of the dataset, the learning set, and then we will use the remaining part of the dataset as test set to assess the performances of the learned model. The selected datasets are:

Dataset 1 - learning set: SP500 from 3 Jan 1994 to 17 Dec 2003 and test set: SP500 from 18 Dec 2003 to 23 Oct 2006.
Dataset 3 - learning set: SP500 from 3 Jan 2008 to 31 Dec 2008 and test set: SP500 from 2 Jan 2009 to 20 Aug 2010.
The reason for selecting Datasets 1 and 2 is that they have been used to test other learning algorithms and are therefore useful to compare our system with other ones. Dataset 3 and 4 instead are recent and smaller datasets which can show the performances of L-FABS with fewer learning data and on turbulent markets such as the recent ones at the time of writing. All the datasets contain the daily closing values of the indexes and have been freely acquired from the finance section of yahoo.com.

In the performed experiments, the Sentiment value will be estimated using Sentiment(round,1), case identified with S1, or calling Sentiment(round,5), case identified with S2. We recall that in case S1, only the previous day value of the real index is used, while in case S2, the average of the latest previous five days is used, as explained in Section 3. In term of information available to S-FABS, we will explore both the $FullKnowledge$ learning scenario, the information about the correct value at time t is used when making a prediction for t+1, and the $LimitedKnowledge$ setting, only information about the the first point in the time serie is used in making a prediction.

Finally, we will evaluate S-FABS when estimating the next value of the time serie (the value of the next trading day) and the seven days ahead value of the time serie. The seven days ahead prediction has been selected as it is the most far ahead prediction made by other learning systems and thus can serve as an interesting comparison data.

The following reported results are averaged over 10 runs of the same experimental setting and the shown forecast errors are measured on test sets. In table 1, the performances of L-FABS are shown when run on Datasets 1 and 3 above described. The columns stand for: "Knowledge" identifies the learning setting as with $FullKnowledge$ or $LimitedKnowledge$, "Day to predict" indicates the number of days ahead for which a prediction of the time serie is made, "Sentiment" indicates if the Sentiment index is calculated with modality S1 or S5, and, finally, the measured errors on the test set are reported in terms of the MAPE and Standard Error. The results in table 1 show lower errors both in terms of the MAPE and of the Standard Error when the $FullKnowledge$ learning setting is selected instead of the $PartialKnowledge$ one. The empirical finding suggests that using all the information available in the time series up to the current time, before making a prediction, produces better forecasts.

However the performances obtained under the $PartialKnowledge$ learning set up are not as bad as one could expect considering that the system would only know about the starting point of the time serie. This finding may suggest that L-FABS is able to learn a model for the time serie that has captured its intrinsic dynamics. Moreover, from table 1, it appears that the predictions of the next day values are more accurate than the predictions made for the seven days ahead values. This result confirms the intuitive experience that the farther a prediction is moved into the future, the less accurate it will be.

Another finding is that the error rates obtained in the learning scenario $FullKnowledge$ on Datasets 1 and 2 are lower than those obtained on Dataset 3 and 4. The relationship is similar but reversed when considering the learning scenario $PartialKnowledge$. A possible interpretation is that during the period of time represented in Datasets 1 and 2, both the SP500 and DJIA time series had a behavior respecting short term trend. Thus the information about the real value for the time series at time t can help when forecasting at time t+1 and farther. An opposite relationships has been found on Datasets 3 and 4. During this period of time, the SP500 and DJIA time series have behaved more erratically (with an higher volatility), so knowing the latest real value of the time serie cannot help in reducing the forecasting error. The ability of L-FABS to learn a deep model of the domain appears here quite useful in dealing with time series (market periods) with high volatility.

For the sake of completeness, we also show the graph of time series as predicted by L-FABS in fig. 1 which is an exemplar of the results obtained over several runs of L-FABS. The predicted time series are compared with the actual ones. As it can be seen from the graphs, the solid line (actual) and the dotted line (predicted) are very close confirming the error figures that have been reported in the tables.

9 SP500, VIX and a Way to calculate the market sentiment.

Let us consider in this section alternative ways to estimate the market sentiment other then using the moving average of previous days close as described earlier. In order to do this, let us introduce a new type of financial asset: index options. Options are derivative financial assets, derivative as their price depends on the price of an underlying (primary) asset, that provide the buyer with some rights and the seller with some obligations. Those rights can be exercised and

Experimental results on Dataset 1

Knowledge	Day to predict	Sentiment	MAPE	StdErr
PK	1	S1	0.70	58.08
PK	1	S5	1.06	57.84
PK	7	S1	1.46	59.88
PK	7	S5	1.65	58.73
FK	1	S1	0.76	16.55
FK	1	S5	0.70	17.31
FK	7	S1	1.46	25.16
FK	7	S5	1.42	24.29

Experimental results on Dataset 3

Knowledge	Day to predict	Sentiment	MAPE	StdErr
PK	1	S1	1.66	32.41
PK	1	S5	2.27	34.20
PK	7	S1	3.45	42.22
PK	7	S5	4.23	49.11
FK	1	S1	1.27	19.53
FK	1	S5	1.23	20.05
FK	7	S1	3.00	36.77
FK	7	S5	3.01	36.75

Table 1: Experimental findings on Datasets 1 and 3 relative to different periods of the SP500 and DJIA time series.

those obligations must be respected some time in the future or for some interval of time in the future depending on the type of option. The two main types of options are Calls and Puts. A Call option gives the right to buy an asset at a given price, a Put option give the right to sell an asset at a given price. Index options are options expressed on market indexes.

As the price of an options includes some financial action/decision that has to take place in the future, it is thought that the pricing of options includes the expectation of the market of what will happen to the underlying asset. So for instance if the underlying asset is expected to rise, then the call option on the asset would become more expensive over time than the relative put option on the same asset.

Given this short introduction to options and index options, consider the graphs in fig. 2. The first graphs represent a the behavior of the SP500 index from July 3, 2007 until August 20, 2012. The second graph reports the VIX index. The VIX value represents how expensive are, at a given point in time, short term options on the SP500. Thus an high value of VIX means that the price of short term options is increasing, a lower value, they are decreasing. This can also be read as if the VIX is high, the market is expecting the SP500 to make a significant move and then they buy options to protect their position. Vice versa if the VIX is low, the market is assuming that the SP500 will not move and thus is not interested in buying any protection.

The above consideration could be applied to our research as follow. As we need to estimate the market Sentiment value for the near future, one approach explored in the paper has been to use a function based on moving averages and previous day closes to estimate the current market mood. However, another approach would be to convert the current VIX value into a Sentiment value for our system.

In this case, the resulting sentiment value would be a measure of the increase in price in ”insurances”/options, paid in the market to protect about future moves.

Let’s us consider the third graph in the figure. The financial community tends to think that high value of the VIX will happen when the market is about to move or is moving lower very quickly. Thus using the VIX we could also estimate the direction of the move and not only the expected amplitude.

10 Experimental comparison of L-FABS to other systems

For providing an exhaustive evaluation of L-FBAS, we will compare its performances with respect to those obtained with alternative approaches. We will

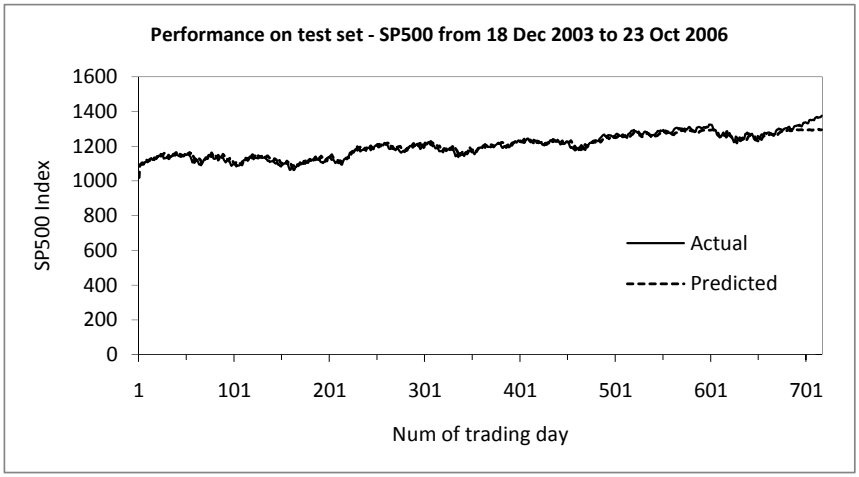

Figure 1: The actual and predicted SP500 time series when testing L-FABS on Dataset 2 with settings: FullKnowledge, S1, Day to predict: 1.

Time serie	Day to predict	MAPE PSO	MAPE MLP	MAPE L-FABS
SP500	1	0.66	1.00	0.70
SP500	7	1.46	3.11	1.42

Table 2: Experimental results averaged on 10 runs for time series SP500 (Dataset 1) and DJIA (Dataset 2).

therefore consider how well the same time series can be modeled by a particle swarm optimization algorithm (an evolutive learning method), a multi layer perceptron (a simple neural network), and L-FABS. We selected those methods for comparisons as they have been used to model time series of market indexes by other researchers [19] so they can act as an useful benchmark.

Other approaches to modeling financial markets or trading strategies, like those cited at the beginning of the paper, could not be used because full details about their experimental set ups was not available. This is understandable considering that the focus of their study was not the minimization of an error rate but the integration of some modeling concepts into a trading system to maximize the trading results, like in [16]. Or, in other cases, the focus of the research was on trying to understand the sensitivity of the system to parameter variations and how the internal system mechanics would work in artificial context as, for instance, in [18].

In table 2, we compare the prediction errors, as given by the MAPE measure, for the SP500 and DJIA time series, corresponding to Datasets 1 and 2 of the previous section, by using a Particle Swarm Optimization algorithm (PSO) [27] and a Multi-Layer Perceptron (MLP) [28], whose parameters have been set up as in [19], with our agent based approach L-FABS. The results for L-FABS are relative to a $FullKnowledge$ learning scenario and a Sentiment determined as in case S5. This learning set up has been chosen so that L-FABS can use the information present in the time series as it has been done when setting up PSO and MLP. Also note that no tuning of L-FABS has been made so that it could outperform the other learning systems, just the setting used in the previous experiments has been used. The reason is that we are interested in evaluating the performances of L-FABS across a set of different experimental conditions without tuning its parameters for each specific experiments.

Before commenting on the error figures, we want to point out how elaborate are the inputs given to PSO and MLP to obtain the reported prediction rates. Any learning/prediction in PSO or MLP requires: the real value of the time serie at the current time t plus a number of financial technical indicators[4] such as: EMA10, EMEA20, EMEA30, ADO, and a few others[5].

At a first glance, the results in Table 2, shows that the forecasting errors of L-FABS are in between those

[4]Financial technical indicators are useful in forecasting financial time series according to the philosophy of financial technical analysis used by technical traders. For a primer on technical analysis and indicators, any introductory book on the topic will do: for instance [13].

[5]As an example, EMA10 stands for the Exponential Moving Average calculated over 10 previous periods and is defined as EMA(t,j=10) = value(t-1) x a + EMA(t-1,j-1)*(1-a) where value(t) is the value at time t of the time series under consideration and the smoothing factor a is defined as a = 2 / (j + 1).

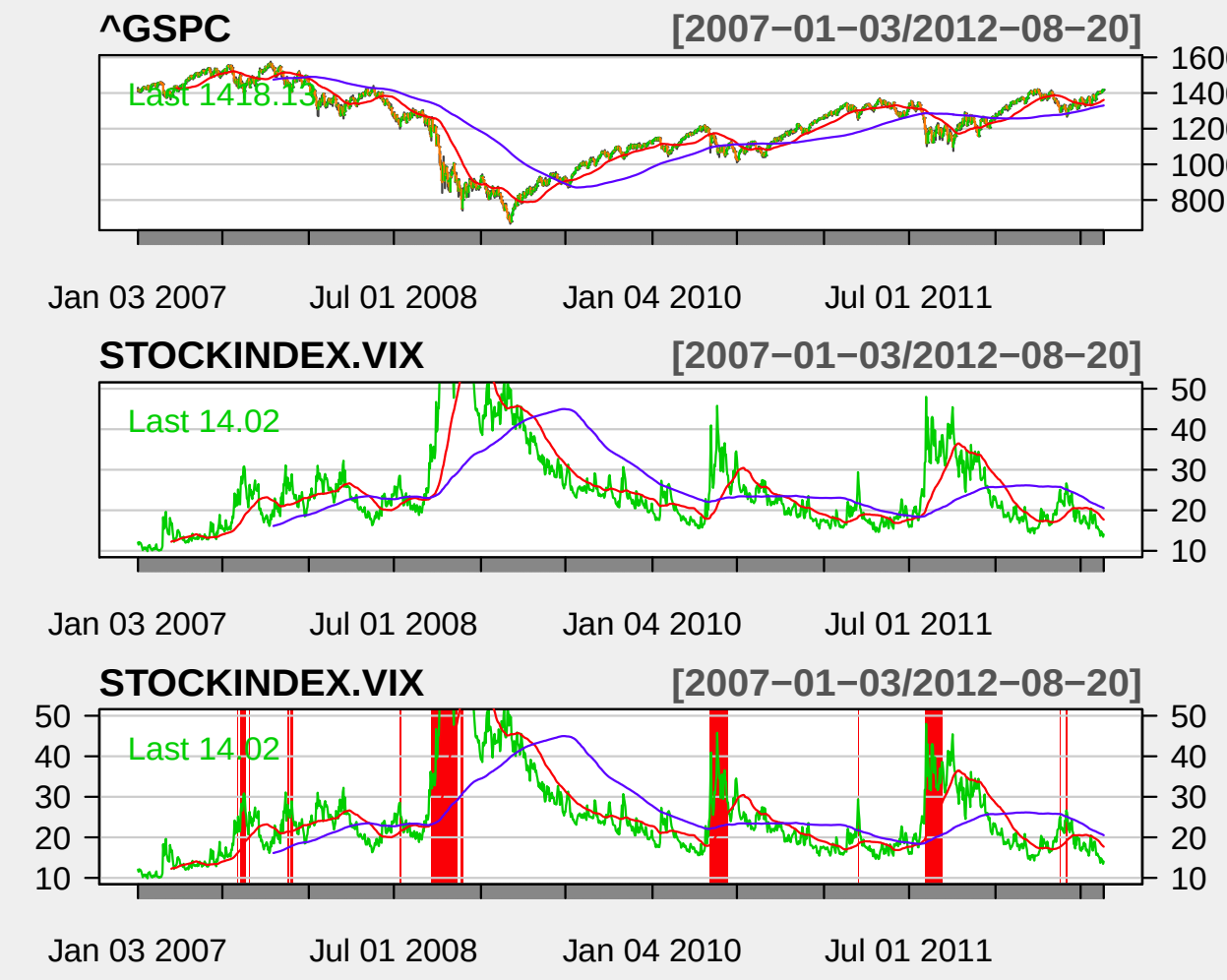

Figure 2: SP500, VIX and another way to calculate the market sentiment.

of PSO and MLP then its performances are comparable to others state of the art forecasters. However, and this is an important point, no complex knowledge representation (features) is necessary to L-FABS to obtain the reported results, just the plain time serie, while for the other systems a number of complex data representation, the technical indicators, have to be given (and finely tuned) so that they can produce the reported results.

From the experimental findings, it appears that the forecasting error increases when farther into the future the prediction is. This holds for all the systems. This observation can be interpreted as evidence that the information contained in the time series up to the current time has a decreasing usefulness in predicting future values the farther we move ahead in time. This finding also confirms what we expect to happen in real world financial markets. Otherwise, statisticians or machine learning scientists would all have become rich long time ago by trading stocks.

An additional consequence for the latest observation, is that even if we use information rich input data, like those built on technical indicators that provide a way to summarize the past behavior of the time series, the predictive performances of the models would still decrease the farther into the future the time to be predicted is.

11 Conclusions

In the paper we have discussed how an agent based modeling techniques, combined with an evolutionary learning algorithm, resulted in a learning simulator, L-FABS, able to acquire the model of financial time series. We have empirically evaluated the system, under several parameter settings, on several time series extracted from the historical evolution of the SP500 and DJIA indexes since 1993. Also we compared the performances of our system with respect to other approaches on the same time series. The main results of our work are that: a) a simple simulator combin-

ing agent based modeling and simulated annealing can closely approximate the SP500 and DJIA time series; b) the observed approximation errors are comparable to those observed in other approaches based on neural networks or particle swarm optimization; c) finally, our approach learn models that can provide information about the contribution of each individual agent to the process of value formation for a financial asset.

References:

[1] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences*, vol. 99, pp. 7280–7287, May 2002.

[2] J. M. Epstein and R. Axtell, *Growing artificial societies: social science from the bottom up*. Washington, DC, USA: The Brookings Institution, 1996.

[3] F. Neri, "Empirical investigation of word-of-mouth phenomena in markets: a software agent approach," *WSEAS Transaction on Computers*, vol. 4, no. 8, pp. 987–994, 2005.

[4] F. Neri, "A comparative study of a financial agent based simulator across learning scenarios," in *ADMI 2011 Agent and Data Mining Interaction, workshop of the Autonomous Agents and Multi Agents Systems conference (AAMAS 2011), Cao L. et al. Eds.*, vol. LNAI 7103, pp. 86–97, Springer, 2012.

[5] F. Neri, "Learning and predicting financial time series by combining natural computation and agent simulation," in *EvoApplications (2) Di Chio et al. Eds.*, vol. LNCS 6625, pp. 111–119, Springer, 2011.

[6] F. Neri, "Agent based modeling under partial and full knowledge learning settings to simulate financial markets," *AI Communications*, vol. in press, 2012.

[7] F. Neri, "Pirr: a methodology for distributed network management in mobile networks," *WSEAS Transaction on Information Science and Applications*, vol. 5, no. 3, pp. 306–311, 2008.

[8] F. Neri, "Traffic packet based intrusion detection: decision trees and generic based learning evaluation," *WSEAS Transaction on Computers*, vol. 4, no. 9, pp. 1017–1024, 2005.

[9] B. Lebaron, "Agent based computational finance: Suggested readings and early research," *Journal of Economic Dynamics and Control*, vol. 24, pp. 679–702, 1998.

[10] L. Tesfatsion, "Agent-based computational economics: Growing economies from the bottom up," *Artif. Life*, vol. 8, no. 1, pp. 55–82, 2002.

[11] A. O. I. Hoffmann, S. A. Delre, J. H. von Eije, and W. Jager, "Artificial multi-agent stock markets: Simple strategies, complex outcomes," in *Advances in Artificial Economics, volume 584 of Lecture Notes in Economics and Mathematical Systems*, pp. 167–176, Springer-Verlag, 2006.

[12] G. Kendall and Y. Su, "A multi-agent based simulated stock market - testing on different types of stocks," in *Congress on Evolutionary Computation CEC03*, pp. 2298–2305, 2003.

[13] C. Kirkpatrick and J. Dahlquist, *Technical Analysis: The Complete Resource for Financial Market Technicians*. FT Press, 2006.

[14] S. Schulenburg, P. Ross, and S. Bridge, "An adaptive agent based economic model," in *Learning Classifier Systems: From Foundations to Applications, volume 1813 of Lecture Notes in Artificial Intelligence*, pp. 265–284, Springer-Verlag, 2000.

[15] M. A. H. Dempster, T. W. Payne, Y. Romahi, and G. W. P. Thompson, "Computational learning techniques for intraday fx trading using popular technical indicators," *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 744–754, 2001.

[16] G. Creamer and Y. Freund, "Automated trading with boosting and expert weighting," *Quantitative Finance*, vol. 4, no. 10, pp. 401–420, 2010.

[17] H. Takahashi and T. Takano, "Analyzing the influence of overconfident investors on financial markets through agent-based model," in *Intelligent Data Engineering and Automated Learning - IDEAL 2007, volume 4881 of Lecture Notes in Computer Science*, pp. 1042–1052, Springer-Verlag, 2007.

[18] W. B. Arthur, J. H. Holland, B. LeBaron, R. Palmer, and P. Taylorm, "Asset pricing under endogenous expectation in an artificial stock market," in *The Economy as an Evolving Complex System II*, pp. 15–44, Santa Fe Institute Studies in the Sciences of Complexity Lecture Notes, 1997.

[19] R. Majhi, G. Sahoo, A. Panda, and A. Choubey, "Prediction of sp500 and djia stock indices using particle swarm optimization techniques," in *Congress on Evolutionary Computation 2008*, pp. 1276–1282, IEEE press, 2008.

[20] I. Kitov, "Predicting conocophillips and exxon mobil stock price," *Journal of Applied Research in Finance*, vol. 2, pp. 129–134, 2009.

[21] F. Neri, "Using software agents to simulate how investors' greed and fear emotions explain the behavior of a financial market," in *WSEAS Conference ICOSSE 09, Genoa, Italy*, pp. 241–245, 2009.

[22] B. Wuthrich, V. Cho, S. Leung, D. Permunetilleke, K. Sankaran, J. Zhang, and W. Lam, "Daily stock market forecast from textual web data," in *IEEE International Conference on Systems, Man, Cybernetics*, pp. 2720–2725, 1998.

[23] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, Ma: Addison-Wesley, 1989.

[24] J. R. Quinlan, *C4.5: Programs for Machine Learning*. California: Morgan Kaufmann, 1993.

[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations*, pp. 318–362, Cambridge, MA, USA: MIT Press, 1986.

[26] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[27] J. Kennedy and R. Eberhard, "Particle swarm optimization," in *Int. Conf. on Neural Networks*, pp. 1942–1948, IEEE press, 1995.

[28] J. Zirilli, *Financial prediction using Neural Networks*. International Thompson Computer Press, 1997.