Learning and Nonlinear Models – Revista da Sociedade Brasileira de Redes Neurais, Vol. 5, No. 2, pp. 137–154, 2007

SWARM-BASED HYBRID INTELLIGENT FORECASTING METHOD FOR FINANCIAL TIME SERIES PREDICTION

Ricardo de A. Araújo[1]
[1]Information Technology Department
[gm]^2 Intelligent Systems
Campinas, SP, Brazil
ricardo@gm2.com.br

Abstract – This paper presents a new approach, referred to as Swarm-based Hybrid Intelligent Forecasting (SHIF) method, for financial time series forecasting. It consists of a hybrid intelligent model composed of an Artificial Neural Network (ANN) and a Particle Swarm Optimizer (PSO), which determines the relevant time lags to represent a complex time series, as well as to evolve the structure and parameters of an ANN (pruning process) and its training algorithm (used to optimize the ANN weights supplied by the PSO). Initially, the proposed method chooses the best prediction model and posteriorly it uses a statistical behavioral test with a phase fix procedure to adjust time phase distortions (characterized by a one step delay of generated predictions regarding real time series values), where such distortions are commonly found in financial time series like. Furthermore, an experimental analysis is conducted with the proposed method using four real world financial time series, and the achieved results are discussed and compared, according to a group of five relevant performance metrics, to results found with previously models proposed in literature.

Keywords: Particle Swarm Optimization, Swarm-based Systems, Financial Time Series Forecasting, Hybrid Intelligent Models, Artificial Neural Networks.

1 INTRODUCTION

Forecasting systems have been widely used for decision making. In this sense, an accurate prediction can build the future, being a key element for the success of a given decision. Many efforts have been made to the development of forecasting models able to determine the future behavior of a given phenomenon based on its past and present data.

Several linear and nonlinear statistical models were proposed for such [1–5]. Among all, the popular linear statistical approach based on Auto Regressive Integrated Moving Average (ARIMA) models [1] is one of the most common choices for the time series forecasting, particularly in financial and economic applications (financial marketing, stock exchange, etc). However, ARIMA models are linear and introduces a limitation in the accuracy of the predictions generated. In addition, nonlinear statistical models used to overcome linear statistical models limitations usually involve high technical and mathematical complexities, and have similar performance to linear statistical models [6].

Alternately, many approaches based on artificial intelligence are used in financial and economical applications. However, in the last two decades, the most popular approach for nonlinear modeling of time series is based on Artificial Neural Networks (ANNs) [7]. In order to define a solution to a given problem, the ANNs require the definition of a parameters set. The ANN topology, the number of hidden processing units, the algorithm for ANN training (and its corresponding variables) are just some of those parameters that require definition. In addition, in the particular case of time series forecasting, another crucial element is determine the relevant time lags to represent the time series.

In this context, hybrid intelligent approaches have produced interesting results, as Leung et al. [8], which proposed an evolutionary algorithm to determine ANNs architecture and parameters, Matilla-García and Arguello [9], which presented a hybrid approach using Genetic Algorithms (GAs) [10, 11] and ANNs to the study of profitability in the Spanish Stock Market, Araújo et al. [12, 13], which presented a hybrid evolutionary morphological approach to design a Modular Morphological Neural Networks (MMNNs) [14] based on increasing and non-increasing translation invariant morphological operators and according to Matheron decomposition [15] and Banon and Barrera decomposition [16], and Ferreira et al. [7], which presented a new hybrid method, combining ANNs and GAs to build an optimal model used to financial time series forecasting. In this way, it is possible notice that some of these works have focussed on the evolution of the model parameters whereas others aimed at evolving the model architecture.

An interesting technique that can be applied to evolve prediction models is the Particle Swarm Optimizer (PSO) [17], which represents a stochastic optimization mechanism inspired by the self-organized behavior of bird flocks or the sociological behavior of a group of people. The PSO was introduced by Kennedy and Eberhart [18] and is commonly applied to optimization problems due to its high search power in state spaces, where it has been used to solve a wide number of complex problems, like ANN training [19–21] and function minimization [22, 23].

This work presents the Swarm-based Hybrid Intelligent Forecasting (SHIF) method for financial time series prediction. It is inspired on Takens Theorem [24] and consists of a hybrid intelligent model composed of an ANN (MultiLayer Perceptron (MLP) like) with a PSO search mechanism [17], which searches for (i) the minimum number of relevant time lags necessary to efficiently represent complex time series, (ii) the best evolved neural network structure in terms of the number of hidden processing units and network weights, and (iii) the appropriate algorithm for training the ANN model (RPROP [25], Levenberg Marquardt [26], Scaled Conjugate Gradient [27] and One Step Secant Conjugate Gradient [28]) that boosts the prediction model performance. After the model training, the proposed method selects the most fitted prediction model for the time series representation, and performs a behavioral statistical test with a phase fix procedure to adjust time phase distortions observed in financial time series (characterized by one step prediction delay – the random walk dilemma).

Furthermore, an experimental analysis is conducted with the proposed method using four financial time series (Apple Inc Stock Prices, Applied Materials Inc Stock Prices, General Electric Company Stock Prices and Hewlett-Packard Company Stock Prices). Five well-known performance metrics are used to assess the performance of the proposed method: Mean Square Error (MSE), Mean Absolute Percentage Error (MAPE), U of Theil Statistics (THEIL), Prediction Of Change In Direction (POCID) and Average Relative Variance (ARV). The experimental results shown a better performance of the proposed method when compared to the MLP networks and the previously introduced Time-delay Added Evolutionary Forecasting (TAEF) method [7].

This paper is organized as follows. In Section 2 are presented fundamentals of the time series forecasting problem, the random walk dilemma for financial time series, the TAEF method, the PSO and the concept of MLPs with link switches. The Section 3 describes the proposed SHIF method. The Section 4 shows the performance measures used to assess the performance of the proposed method. In the Section 5 is presented the simulations and the experimental results with the proposed method, as well as a comparison between these experimental results and the results given by both the MLP and TAEF models for four financial time series. At the end, in Section 6, it is presented the conclusions of this work.

2 FUNDAMENTALS

In this section will be presented the fundamentals and theoretical concepts necessary to comprehension of the proposed method.

2.1 The Time Series Forecasting Problem

A time series is a sequence of observations about a given phenomenon, where it is observed in discrete or continuous space. In this work all time series will be considered time discrete and equidistant.

Usually, a time series can be defined by

$$X_t = \{x_t \in \mathbb{R} \mid t = 1, 2, \ldots, N\}, \tag{1}$$

where t is the temporal index and N is the number of observations. The term X_t will be seen as a set of temporal observations of a given phenomenon, orderly sequenced and equally spaced.

The aim of prediction techniques applied to a given time series (X_t) is to provide a mechanism that allows, with certain accuracy, the prediction of the future values of X_t, given by X_{t+k}, $k = 1, 2, ...$, where k represents the prediction horizon. These prediction techniques will try to identify certain regular patterns present in the data set, creating a model capable of generating the next temporal patterns, where, in this context, a most relevant factor for an accurate prediction performance is the correct choice of the past window, or the time lags, considered for the representation of a given time series.

Box & Jenkins [1] shown that when there is a clear linear relationship among the historical data of a given time series, the functions of auto-correlation and partial auto-correlation are capable of identifying the relevant time lags to represent a time series, and such procedure is usually applied in linear models. However, when it uses a real world time series, or more specifically, a complex time series with all their dependencies on exogenous and uncontrollable variables, the relationship that involves the time series historical data is generally nonlinear, which makes the Box & Jenkins' analysis procedure of the time lags only a crude estimate.

In mathematical sense, such relationship involving time series historical data defines a d-dimensional phase space, where d is the minimum dimension capable of representing such relationship. Therefore, a d-dimensional phase space can be built so that it is possible to unfold its corresponding time series. Takens [24] proved that if d is sufficiently large, such phase space is homeomorphic to the phase space that generated the time series. Takens' Theorem [24] is the theoretical justification that it is possible to build a state space using the correct time lags, and if this space is correctly rebuilt, Takens' Theorem [24] also guarantees that the dynamics of this space is topologically identical to the dynamics of the real system state space.

The main problem in reconstructing the original state space is naturally the correct choice of the variable d, or more specifically, the correct choice of the important time lags necessary for the characterization of the system dynamics. Many proposed methods can be found in the literature for the definition of the lags [29–31]. Such methods are usually based on measures of conditional probabilities, which consider,

$$X_t = f(x_{t-1}, x_{t-2}, \ldots, x_{t-d}) + r_t, \tag{2}$$

where $f(x_{t-1}, x_{t-2}, \ldots, x_{t-d})$ is a possible mapping of the pasts values to the facts of the future (where x_{t-1} is the lag 1, x_{t-2} is the lag 2,. . . , x_{t-d} is the lag d) and r_t is a noise term.

However, in general, these tests found in the literature are based on the primary dependence among the variables and do not consider any possible induced dependencies. For example, if

$$f(x_{t-1}) = f(f(x_{t-2})), \tag{3}$$

it is said that x_{t-1} is the primary dependence, and the dependence induced on x_{t-2} is not considered (any variable that is not a primary dependence is denoted as irrelevant).

The method proposed in this paper, conversely, does not make any prior assumption about the dependencies between the variables. In other words, it does not discard any possible correlation that can exist among the time series parameters, even higher order correlations, since it carries out an iterative automatic search for solving the problem of finding the relevant time lags.

2.2 The Random Walk Dilemma

A naive prediction strategy is to define the last observation of a time series as the best prediction of its next future value ($X_{t+1} = X_t$). This kind of model is known as the Random Walk (RW) model [32], which is defined by

$$X_t = X_{t-1} + r_t, \tag{4}$$

or

$$\Delta X_t = X_t - X_{t-1} = r_t, \tag{5}$$

where X_t is the current observation, X_{t-1} is the immediate observation before X_t, and r_t is a noise term with a gaussian distribution of zero mean and standard deviation σ ($r_t \approx N(0, \sigma)$). In other words, the rate of time series change (ΔX_t) is a white noise.

The model above clearly implies that, as the information set consists of past time series data, the future data are unpredictable. On the average, the value X_t is indeed the best prediction of value X_{t-1}. This behavior is common in the finance market and in the economic theory and it is so-called random walk dilemma or random walk hypothesis [32].

The computational cost for time series forecasting using the random walk dilemma is extremely low. Therefore, any other prediction method more costly than a random walk model should have a very superior performance than a random walk model, otherwise its use is not interesting in the practice.

However, if the time series phenomenon is driven by law with strong similarity to a random walk model, any model applied to this time series phenomenon will tend to have the same performance than a random walk model.

Assuming that an accurate prediction model is used to build an estimated value of X_t, denoted by $\widehat{X_t}$, the expected value ($E[\cdot]$) of the difference between $\widehat{X_t}$ and X_t must tends to zero,

$$E[\widehat{X_t} - X_t] \to 0. \tag{6}$$

If the time series generator phenomenon is supposed to have a strong random walk linear component and a very weak nonlinear component (denoted by $g(t)$), and assuming that $E[r_t] = 0$ and $E[r_t r_k] = 0$ ($\forall\ k \neq t$), the expected value of the difference between $\widehat{X_t}$ and X_t will be

$$\begin{aligned}
&E[\widehat{X_t} - (X_{t-1} + g(t) + r_t)] \to 0\\
&E[\widehat{X_t}] - E[X_{t-1}] - E[g(t)] - E[r_t] \to 0\\
&E[\widehat{X_t}] - E[X_{t-1}] - E[g(t)] \to 0\\
&E[\widehat{X_t}] \to E[X_{t-1}] + E[g(t)].
\end{aligned}$$

But $E[X_{t-1}] \gg E[g(t)]$, then $E[X_{t-1}] + E[g(t)] \simeq E[X_{t-1}]$ and

$$E[\widehat{X_t}] \to E[X_{t-1}]. \tag{7}$$

Therefore, in these conditions, to escape of random walk dilemma is a hard task. Indications of this behavior (strong linear random walk component and a weak nonlinear component) can be observed from time series lagplot graphics. For example, lagplot graphics where strong linear structures are dominant with respect to nonlinear structures [33], generally observed in the financial and economical time series.

2.3 The Time-delay Added Evolutionary Forecasting Method

The Time-delay Added Evolutionary Forecasting (TAEF) method [7] tries to reconstruct the phase space of a given time series by carrying out a search for the minimum dimensionality necessary to reproduce the generator phenomenon of the time series. The TAEF method is an intelligent hybrid system based on Artificial Neural Networks (ANNs) architectures trained and adjusted by a Modified Genetic Algorithm (MGA) which not only searches for the ANN parameters but also for the adequate embedded dimension represented in the time lags.

The scheme describing the TAEF algorithm is based on the iterative definition of the four main elements: (i) the underlying information necessary to predict the series (the minimum number of lags), (ii) the structure of the model capable of representing such underlying information for the purpose of prediction (the number of units in the ANN structure), (iii) the appropriate algorithm for training the model, and (iv) a behavior test to adjust time phase distortions that appear in some time series.

Following this principle, the important parameters defined by the algorithm are:

1. The number of time lags to represent the series;
2. The number of units in the ANN hidden layer;
3. The training algorithm for the ANN.

The TAEF method starts with the user defining a minimum initial fitness value (*MinFit*) which should be reached by at least one individual of the population in a given MGA round. The fitness function is defined as

$$\text{Fitness Function} = \frac{1}{1 + \text{MSE}} \tag{8}$$

where MSE is the Mean Squared Error of the ANN and will be formally defined in the Section 4.

In each MGA round, a population of M individuals is generated, each of them being represented by a chromosome (in the Ferreira's works [7] were used $M = 10$). Each individual is in fact a three-layer ANN where the first layer is defined by the number of time lags, the second layer is composed of a number of hidden processing units (sigmoidal units) and the third layer is composed by one processing unit (prediction horizon of one step ahead).

The stopping criteria for each one of the individual are the number of epochs (NEpochs), the increase in the validation error (Gl) and the decrease in the training error (Pt).

The best repetition (the more small validation error) is chosen to represent the best individual. Following this procedure, the MGA evolves towards an optimal or close to optimal fitness solution (which may not be the best solution yet), according to the stopping criteria: number of generations created (NGen) and fitness evolution of the best individual (BestFit).

After this point, when the MGA reaches a solution, the algorithm checks if the fitness of the best individual paired or overcame the initial value specified for the variable MinFit (minimum fitness requirement). If this is not the case, the value of MaxLags (maximum number of lags) is increased by one and the MGA procedure is repeated to search for a better solution.

However, if the fitness reached was satisfactory, then the algorithm checks the number of lags chosen for the best individual, places this value as MaxLags, sets MinFit with the fitness value reached by this individual, and repeats the whole MGA procedure. In this case, the fitness achieved by the best individual was better than the fitness previously set and, therefore, the model can possibly generate a solution of higher accuracy with the lags of the best individual (and with the MinFit reached by the best individual as the new target). If, however, the new value of MinFit is not reached in the next round, MaxLags gets again the same value defined for it just before the round that found the best individual, increased by one (the maximum number of lags is increased by one). The state space for the lag search is then increased by one to allow a wider search for the definition of the lag set. This procedure goes on until the stop condition is reached. After that, the TAEF method chooses the best model found among all the candidates.

After the best model is chosen, when training process is finished, a statistical test (t-test) is employed to check if the network representation has reached an "in-phase" matching (without a one step shift – the shape of the time series and the shape of the generated prediction has a time matching) or "out-of-phase" matching (with a one step shift – the shape of the time series and the shape of the generated prediction do not have a time matching). If this test accepts the "in-phase" matching hypothesis, the elected model is ready for practical use. Otherwise, the method carries out a new procedure to adjust the relative phase between the prediction and the actual time series. The validation patterns are presented to the ANN and the output of these patterns are re-arranged to create new inputs that are both presented to the same ANN and set as the output (prediction) target.

2.4 The Particle Swarm Optimizer

The Particle Swarm Optimizer (PSO) [17] is an optimization technique based on the social behavior (swarm) that a population of individuals (referred to as particles) adapts to its environment. In each epoch, all particles of the solutions population adjust their positions based on their own and neighbors experience, including the current velocities, the current positions and the best previous position experienced by each particle and its neighbors. If any particle discovers a promising solution, the swarm is guided to this new solution in order to explore more thoroughly the region found.

The swarm size is given by s. Each particle i ($1 \leq i \leq s$) represent a trial solution with $j = 1, 2, \ldots, n$ parameters. Each particle, at t time, has a current position $\overrightarrow{x}_i(t) = (x_{i,1}(t), x_{i,2}(t), \ldots, x_{i,j}(t), \ldots, x_{i,n}(t))$ in the search space, a current velocity $\overrightarrow{v}_i(t) = (v_{i,1}(t), v_{i,2}(t), \ldots, v_{i,j}(t), \ldots, v_{i,n}(t))$ and a personal best position $\overrightarrow{y}_i(t) = (y_{i,1}(t), y_{i,2}(t), \ldots, y_{i,j}(t), \ldots, y_{i,n}(t))$ in the search space. Assuming that, to solve a given problem, a cost function f need be minimized, then all swarm particles velocities are updated, at each iteration, by

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1 r_1[y_{i,j}(t) - x_{i,j}(t)] + c_2 r_2[\hat{y}_j(t) - x_{i,j}(t)], \tag{9}$$

where $\hat{y}_j(t)$ denotes the j parameter of the global best particle, c_1 and c_2 represent the acceleration coefficients, which control the velocity change of a particle in a single iteration, and $r_1 \sim U(0,1)$ and $r_2 \sim U(0,1)$ are elements from two uniform random sequences in the interval [0,1]. The term w is referred to as inertia weight, where its value is typically set to vary linearly from 1 to near 0 during the course of the PSO procedure.

In this way, the new particle position is updated by

$$\overrightarrow{x}_i(t+1) = \overrightarrow{x}_i(t) + \overrightarrow{v}_i(t+1). \tag{10}$$

The personal best particle position $\overrightarrow{y}_i$ and the global best particle swarm position $\overrightarrow{\hat{y}} = (\hat{y}_1(t), \hat{y}_2(t), \ldots, \hat{y}_j(t), \ldots, \hat{y}_n(t))$ (found by any particle during all previous iterations) are updated by equations (11) and (12), respectively.

$$\overrightarrow{y}_i(t+1) = \begin{cases} \overrightarrow{y}_i(t) & \text{if } f(\overrightarrow{x}_i(t+1)) \geq f(\overrightarrow{y}_i(t)), \\ \overrightarrow{x}_i(t+1) & \text{otherwise.} \end{cases}; \tag{11}$$

$$\overrightarrow{\hat{y}}(t+1) = \arg\min\left(f(\overrightarrow{y}_i(t+1))\right), \quad 1 \leq i \leq s, \tag{12}$$

where $\arg\min(\cdot)$ denotes the minimum argument and $f(\cdot)$ is a cost function.

The values of each component in every $\overrightarrow{v}_i$ vector can be clamped to the range $[-v_{max}, v_{max}]$ in order to reduce the likelihood of particles leaving the search space. This mechanism does not restrict the values of $\overrightarrow{x}_i$ in the range of $\overrightarrow{v}_i$, it only limits the maximum distance that a particle will move during each iteration [17].

2.5 MultiLayer Perceptron Networks with Link Switches

The MultiLayer Perceptron (MLP) networks have been widely used to solve many complex problems [34]. Commonly, the MLP topology definition process consists in to fix network structure (in terms of inputs, hidden processing units, outputs and connections), but Leung et al. [8] shown that a fixed ANN structure may not guarantee close to optimal or optimal performance within a given training period.

It is possible to verify that a very compact ANN may not provide satisfactory performance due to its limited generalization power. In addition, a very large ANN can have many connections (ANN weights), implying the increase of computational cost. Leung et al. [8] shown a way to perform a pruning procedure, which consists of a inclusion of switches between ANN connections (weights). This procedure offers the effective capacity to determine the most compact ANN, reducing the computational cost and the probability of the overfitting problem. The Figure 1 illustrates a general form of the MLP networks with link switches. In each ANN connection there is an on/off key, represented by a little square, giving the possibility of performing a pruning procedure [35–39], where the irrelevant connections can be removed, generating a compact ANN with high generalization power.

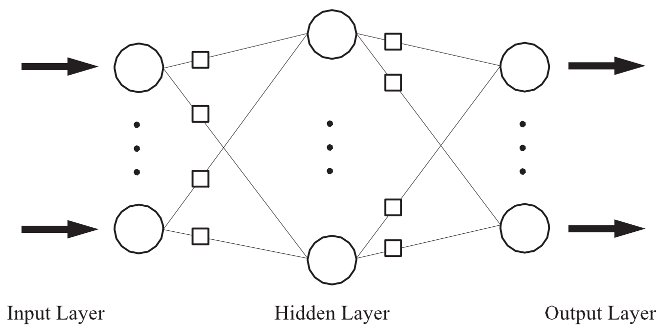

Figure 1: Multilayer perceptron networks with link switches.

It is possible to define some particular descriptions of the three layer MLP architecture. The first architecture for MLP modeling ($\mathrm{NetMod} = 1$), considered in this paper, uses the sigmoidal activation function for all hidden processing units. The output processing unit uses a linear activation function where a sigmoidal function is applied to its bias. This architecture was employed by Leung et al. [8] for Sunspot time series forecasting (one step ahead prediction), where its output is given by:

$$y_k = \sum_{j=1}^{n_h} \delta(S^2_{jk}) W_{jk} Sig\left[\sum_{i=1}^{n_{in}} (\delta(S^1_{ij}) W_{ij} X_i(t) - \delta(S^1_j) b^1_j)\right] - \delta(S^2_k) Sig(b^2_k), \tag{13}$$

where $X_i(t)$ $(i = 1, 2, \ldots, n_{in})$ are the ANN input values, n_{in} denotes the number of ANN input and n_h is the number of hidden units. Whereas the prediction horizon is one step ahead, only one output unit is necessary ($k = 1$). Term $Sig(\cdot)$ represents a sigmoidal function defined by:

$$Sig(x) = \frac{1}{1 + exp(-x)}, \tag{14}$$

and δ denotes a step function, defined by,

$$\delta(x) = \begin{cases} 1, & \text{if } x \geq 0, \\ 0, & \text{otherwise.} \end{cases}. \tag{15}$$

The second MLP architecture considered ($\mathrm{NetMod} = 2$) uses the sigmoidal activation function for hidden processing units and linear activation function for output processing unit. Its MLP output is given by

$$y_k = \sum_{j=1}^{n_h} \delta(S^2_{jk}) W_{jk} Sig\left[\sum_{i=1}^{n_{in}} (\delta(S^1_{ij}) W_{ij} X_i(t) - \delta(S^1_j) b^1_j)\right] - \delta(S^2_k)(b^2_k). \tag{16}$$

Finally, the third MLP architecture employed here ($\mathrm{NetMod} = 3$) applies the sigmoidal activation function to all processing units. Its MLP output is given by

$$y_k = Sig\left\{\sum_{j=1}^{n_h} \delta(S^2_{jk})W_{jk}Sig\left[\sum_{i=1}^{n_{in}}(\delta(S^1_{ij})W_{ij}X_i(t) - \delta(S^1_j)b^1_j)\right] - \delta(S^2_k)(b^2_k)\right\}. \tag{17}$$

These three MLP modeling ($\mathrm{NetMod} = 1, 2, 3$) will be combined with the PSO system, and it will search by architecture that better describes the time series phenomenon.

3 THE PROPOSED SWARM-BASED HYBRID INTELLIGENT FORECASTING METHOD

The methodology proposed in this paper uses a swarm-based search mechanism to choice the architecture modeling, to train and to prune the MLP networks for financial time series forecasting. The main idea of this new methodology is based on TAEF method [7], where three elements necessary for building an accurate forecasting system are considered: (i) The underlying information necessary to predict the time series (the minimum number of time lags adequate for representing the time series), (ii) The structure of the model capable of representing such underlying information for the prediction purpose, and (iii) The appropriate algorithm for training the model. The search by the minimum number of time lags is important because the model must be as parsimonious as possible, avoiding the overfitting problem and decreasing the computacional cost.

According to these principles described above, the methodology proposed in this work, referred to as Swarm-based Hybrid Intelligent Forecasting (SHIF) method, consists of an intelligent hybrid system composed of a Particle Swarm Optimizer (PSO) (Section 2.4) and an ANN (Section 2.5), which determines the following important parameters: (i) The minimum number of time lags to represent the time series: initially, a maximum number of lags ($\mathrm{MaxLags}$) is user pre-defined, where the PSO can choose any subset of specific lags (particular time lags capable of a fine tuned time series characterization) contained in the interval [1,$\mathrm{MaxLags}$] for each population particle, (ii) The number of processing units in the ANN hidden layer: the maximum number of hidden processing units ($\mathrm{MaxHidden}$) is user pre-determined, where the PSO chooses, for each candidate particle, a number of processing units in the hidden layer in the interval [1,$\mathrm{MaxHidden}$], (iii) The network architecture (with pruning), network parameters configuration and the initial weights of the ANN, and (iv) The training algorithm for the ANN: RPROP [25], Levenberg Marquardt [26], Scaled Conjugate Gradient [27] and One Step Secant Conjugate Gradient [28] are candidates for the ANN training algorithm, where the PSO defines one of these algorithms for each particle in the population.

Such process offer the effective capacity to seek the most compact ANN model, also reducing computational cost and probability of model overfitting. Each PSO particle represents an ANN (three-layer MLP network), where the first layer is defined by the time lags number, the second layer is composed by the number of hidden processing units and the third layer is composed by one processing unit (prediction horizon of one step ahead). Figure 2 shows the proposed method scheme.

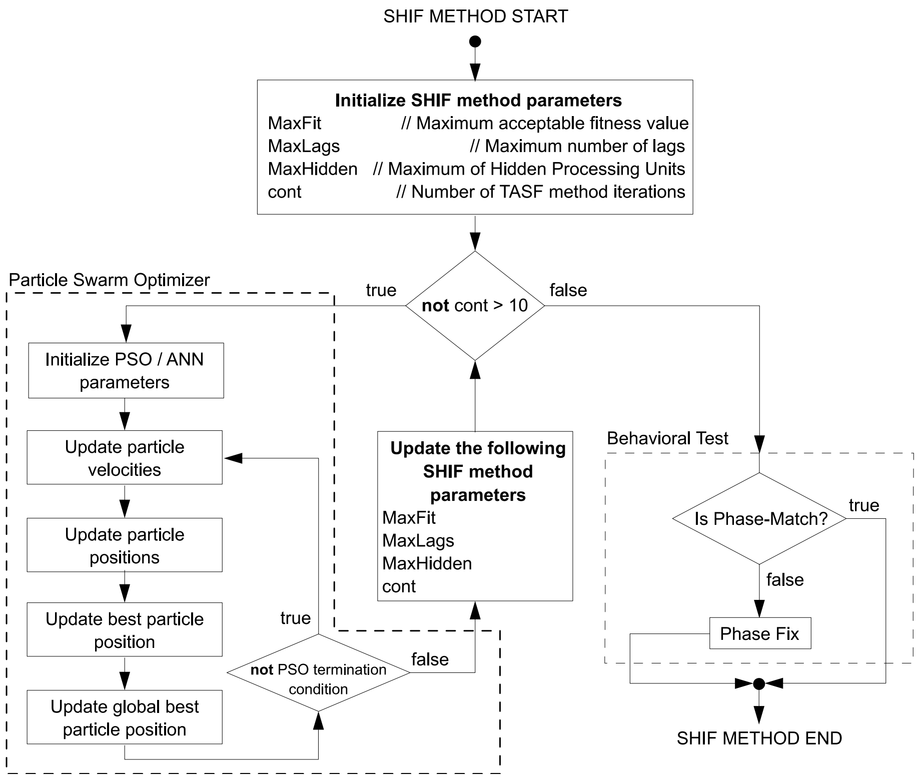
SHIF METHOD START
Initialize SHIF method parameters
MaxFit // Maximum acceptable fitness value
MaxLags // Maximum number of lags
MaxHidden // Maximum of Hidden Processing Units
cont // Number of TASF method iterations
Particle Swarm Optimizer
true
not cont > 10
false
Initialize PSO / ANN parameters
Update particle velocities
Update particle positions
Update best particle position
Update global best particle position
true
not PSO termination condition
false
Update the following SHIF method parameters
MaxFit
MaxLags
MaxHidden
cont
Behavioral Test
Is Phase-Match?
true
false
Phase Fix
SHIF METHOD END

Figure 2: The proposed method scheme.

Initially, it is necessary to define some parameters of the proposed method, which are the number of iterations of the algorithm (cont), the maximum number of time lags ($\mathrm{MaxLags}$), the maximum number of processing units in ANN hidden layer ($\mathrm{MaxHidden}$) and the maximum acceptable fitness function value (MaxFit), which should be reached by at least one particle of the population. As minimization problem, any particle with inferior fitness function values than MaxFit is considered an acceptable model. These parameters are initially user pre-defined and adjusted when the method evolves through the search for the most fitted forecasting model for each time series.

Most works found in the literature have the fitness function (or objective function) based on just one performance measure, like Mean Square Error (MSE). However, Clements et al. [40], since 1993, shown that the MSE measure has some limitations to available and to compare the prediction model performance. The information as the absolute percentual error, the accuracy in the future direction prediction and the relative gain regarding naive prediction models (like random walk models and mean prediction) are not well-described when it uses only MSE measure.

In order to provide a more robust forecasting model, it is defined a new fitness function, which is a combination of five well-known performance measures: Prediction Of Change In Direction (POCID), Mean Square Error (MSE), Mean Absolute Percentage Error (MAPE), Normalized Mean Square Error (NMSE) and Average Relative Variance (ARV), where all these measures will be formally defined in Section 4. The fitness function used here is given by (considering the PSO definition for minimization problems)

$$\mathrm{Fitness\,Function} = 100 - \frac{\mathrm{POCID}}{1 + \mathrm{MSE} + \mathrm{MAPE} + \mathrm{NMSE} + \mathrm{ARV}}. \tag{18}$$

Whereas there are linear and nonlinear metrics in the such fitness function and each one of these metrics can contribute of different forms for the evolution process, the Equation 18 was built of empirical form to have all information necessary to describe as better as possible the time series generator phenomenon.

After parameter initialization, the proposed method performs the PSO to search for the best ANN parameters. Once the PSO reaches its stop condition (defined later), it checks if the best particle (the more small validation error) obtained success with respect to the goal of acceptable maximum fitness value (MaxFit). If the best particle did not reach such goal, the $\mathrm{MaxLags}$ is increased by one, expanding the search space (i.e., the number of time lags) currently employed in the time series representation, and repeats the whole PSO procedure.

However, if the best individual reached or overcame such goal (acceptable maximum fitness value), the proposed method checks the number of lags chosen for the best individual, places this value as $\mathrm{MaxLags}$, sets MaxFit with the fitness value reached by this individual, and repeats the whole PSO procedure. In this case, the fitness reached by the best individual is less than the fitness previously set ($\mathrm{MaxLags}$) and, therefore, the model can possibly generate a solution of higher accuracy with the lags of the best individual (and with the $\mathrm{MaxLags}$ reached by the best individual as the new target). If the new value of MaxFit is not archived in the next round, $\mathrm{MaxLags}$ again receives the same value defined for it just before the round that found the best individual, increased by one. The idea here is that if the time lags found in the best individual were not capable of producing a lower fitness than the one previously found, this may be because some important lag (or lags) has been discarded. The state space for the lag search is then increased by one to allow a wider search for the definition of the lag set. This procedure goes on until the proposed method stop condition is reached.

After model training (at the end of proposed method iterations), the proposed method uses the phase fix procedure presented by Ferreira [7], where a two step procedure is introduced to adjust time phase distortions observed ("out-of-phase" matching) in financial time series. Ferreira [7] shows that the representations of some time series (natural phenomena) were developed by the model with a very close approximation between the actual and the predicted time series (referred to as "in-phase" matching), whereas the predictions of other time series (mostly financial time series) were always presented with a one step delay regarding the original data (referred to as "out-of-phase" matching).

The proposed proposed method uses the statistical test (t-test) to check if the ANN model representation has reached an in-phase or out-of-phase matching (in the same way of TAEF method [7]). This is conducted by comparing the outputs of the prediction model with the actual series, making use only of the validation data set. This comparison is a simple hypothesis test, where the null hypothesis is that the prediction corresponds to in-phase matching and the alternative hypothesis is that the prediction does not correspond to in-phase matching (or correspond to out-of-phase matching). If this test accepts the in-phase matching hypothesis, the elected model is ready for practical use. Otherwise, the proposed method performs a new procedure to adjust the relative phase between the prediction and the actual time series. The phase fix procedure has two steps (described in the Figure 3): (i) the validation patterns are presented to the ANN and the output of these patterns are re-arranged to create new inputs patterns (reconstructed patterns), and (ii) these reconstructed patterns are represented to the same ANN and the output set as the prediction target. This procedure of phase adjustment considers that the ANN does not a random walk model, it just shows a behavior characteristic of a random walk model: the $t + 1$ prediction is taken as the t value (Random Walk Dilemma – Section 2.2). If the ANN was like a random walk model, the phase adjust procedure would not work.

The termination conditions for the PSO are:

1. The maximum iteration number (I_{PSO}): $I_{PSO} = 1000$;

2. The increase in the validation error or generalization loss (Gl) [41]: $Gl > 5\%$;

3. The decrease in the training error process training (Pt) [41]: $Pt \leq 10^{-6}$.

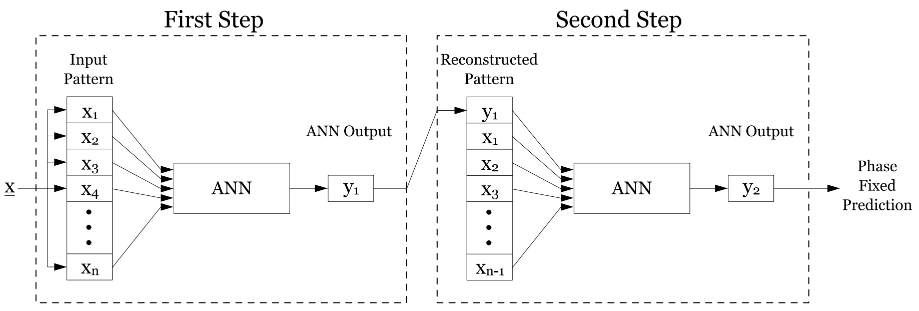

Figure 3: Phase fix procedure.

Each particle of the PSO population is an ANN represented by the data structure with the following components (ANN parameters):

- W_{ij}: weights of connections between the input layer and the hidden layer
- S_{ij}: switches of connections between the input layer and the hidden layer
- W_{jk}: weights of connections between the hidden layer and the output layer;
- S_{jk}: switches of connections between the hidden layer and the output layer;
- b_j^1: bias of the hidden layer;
- S_j^1: switches of the hidden layer bias;
- b_k^2: bias of the output layer;
- S_k^2: switches of the output layer bias;
- NetMod: ANN model.
- $\mathrm{NHidden}$: the number of processing units in the ANN hidden layer;
- $\mathrm{ANNTrainAlg}$: the ANN training algorithm;
- NLags: a vector, where each position has a real-valued codification, which is used to determine if a specific time lag will be used ($\mathrm{NLags}_i \geq 0$) or not ($\mathrm{NLags}_i < 0$).

4 PERFORMANCE EVALUATION

Many performance evaluation criteria are found in literature. However, most of the existing literature on time series prediction frequently employ only one performance criterion for prediction evaluation. The most widely used performance criterion is the Mean Squared Error (MSE), given by

$$\mathrm{MSE} = \frac{1}{N}\sum_{j=1}^{N}(\mathrm{target}_j - \mathrm{output}_j)^2, \tag{19}$$

where N is the number of patterns, target_j is the desired output for pattern j and output_j is the predicted value for pattern j.

The MSE measure may be used to drive the prediction model in the training process, but it cannot be considered alone as a conclusive measure for comparison of different prediction models [40]. For this reason, other performance criteria should be considered for allowing a more robust performance evaluation.

A measure that presents accurately identifying model deviations is the Mean Absolute Percentage Error (MAPE), given by

$$\mathrm{MAPE} = \frac{1}{N}\sum_{j=1}^{N}\left|\frac{\mathrm{target}_j - \mathrm{output}_j}{x_j}\right|, \tag{20}$$

where x_j is the time series value at point j.

The random walk dilemma can be used as a naive predictor ($X_{t+1} = X_t$), commonly applied to financial time series prediction. Thus, a way to evaluate the model regarding a random walk model is using the Normalized Mean Squared Error (NMSE) or U of Theil Statistic (THEIL) [42], which associates the model performance with a random walk model, and given by

$$\mathrm{THEIL} = \frac{\sum_{j}^{N}(\mathrm{target}_j - \mathrm{output}_j)^2}{\sum_{j}^{N}(\mathrm{target}_j - \mathrm{target}_{j-1})^2}, \tag{21}$$

where, if the THEIL is equal to 1, the predictor has the same performance than a random model. If the THEIL is greater than 1, then the predictor has a performance worse than a random walk model, and if the THEIL is less than 1, the predictor is better than a random walk model. In the perfect model, the THEIL tends to zero.

Another interesting measure maps the accuracy in the future direction prediction of the time series or, more specifically, the ability of the method to predict if the future series value (prediction target) will increase or decrease with respect to the previous value. This metric is known as the Prediction Of Change In Direction (POCID) [7], and is given by

$$\mathrm{POCID} = \frac{100}{N}\sum_{j=1}^{N} D_j, \tag{22}$$

where

$$D_j = \begin{cases} 1, & \text{if } (\mathrm{target}_j - \mathrm{target}_{j-1})(\mathrm{output}_j - \mathrm{output}_{j-1}) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

The last measure used associates the model performance with the mean of the time series. The measure is the Average Relative Variance (ARV), and given by

$$\mathrm{ARV} = \frac{\sum_{j=1}^{N}(\mathrm{target}_j - \mathrm{output}_j)^2}{\sum_{j=1}^{N}(\mathrm{output}_j - \overline{\mathrm{target}})^2}, \tag{24}$$

where, $\overline{\mathrm{target}}$ is the mean of the time series. If the ARV is equal to 1, the predictor has the same performance of the time series average prediction. If the ARV is greater than 1, then the predictor has a performance worse than the time series average prediction, and if the ARV is less than 1, the predictor is better than the time series average prediction. In the ideal model, ARV tends to zero.

5 SIMULATIONS AND EXPERIMENTAL RESULTS

A set of four real world financial time series (Apple Inc Stock Prices, Applied Materials Inc Stock Prices, General Electric Company Stock Prices and Hewlett-Packard Company Stock Prices) were used as a test bed for evaluation of the proposed method. All time series investigated corresponds to the daily records of Nasdaq Stock Market from June 23th 2000 to June 22th 2007, constituting a database of 1,758 points. These time series were normalized to lie within the range $[0, 1]$ (because the ANN processing units use sigmoidal activation function) and divided in three sets according to Prechelt [41]: training set (50% of the data points – 879 points), validation set (25% of the data points – 440 points) and test set (25% of the data points – 439 points).

For all the experiments, the following initialization system parameters were used: $\mathrm{cont} = 10$ (number of SHIF method iterations), $\mathrm{MaxFit} = 60$ (maximum acceptable fitness value), $\mathrm{MaxLags} = 4$ (initial time lag maximum dimension) and $\mathrm{MaxHidden} = 10$ (maximum number of the hidden processing unit). The PSO parameters are the same for all experiments. The maximum number of PSO iterations is $I_{PSO} = 1000$. The acceleration coefficients ($c1$ and $c2$) are set to 2.05, the inertia weight (w) is set to 0.9, the v_{max} is set to 2.048 and the terms r_1 and r_2 are random numbers in the range $[0, 1]$. Such values are based on values suggested by VandenBergh and Engelbrecht [17]. The PSO population is composed of 10 particles, where each particle is an ANN with the maximum architecture: $10 - 10 - 1$ (MLP network), which denotes ten units in the input layer, ten units in the hidden layer and one unit in the output layer (prediction horizon of one step ahead). After the ANN weight initialization, each PSO particle is trained by one of the four algorithms below: Levenberg Marquardt (LM) [26], RPROP [25], Scaled Conjugate Gradient [27] and One Step Secant [28], for a maximum period of 10^3 epochs. The termination conditions for the ANN training are the maximum number of epochs, the increase in the validation error or generalization loss beyond 5% ($Gl \gtrsim 5\%$) and the decrease in the training error or process training under 10^{-6} ($Pt < 10^{-6}$).

The simulation experiments involving the SHIF model were conducted with and without the phase fix procedure, referred to as SHIF out-of-phase model and SHIF in-phase model, respectively. These two procedures (in-phase and out-of-phase) were used to study the possible performance improvement, in terms of fitness function and of the phase fix procedure, applied to SHIF model. For each time series, ten experiments were executed and the instance with the more small validation fitness function is chosen to represent the prediction model (minimization problem).

In order to establish a performance study, results previously published in the literature with the TAEF method [7] under the same conditions are employed for comparison of results. In addition, experiments with MultiLayer Perceptron (MLP) networks were used for comparison with the SHIF method. In all of the experiments, ten random initializations for each model were carried out, where the experiment with the more small validation fitness function is chosen to represent the prediction model. The Levenberg Marquardt Algorithm [26] was employed for training the MLP network for a maximum period of 10^3 epochs. The termination conditions for the MLP training are equal to the termination criteria for the ANN training in the PSO system (Epochs, Gl and Pt). For all the series, the best ANN initialization was elected to represent the prediction model. The statistical behavioral test, for phase fix procedure, was also applied to all the MLP and TAEF models in order to guarantee a fair comparison among the models.

It is worth to mention that the results with ARIMA models were not investigated in this comparative analysis since Ferreira [7] shown that MLP network obtained better results than ARIMA models.

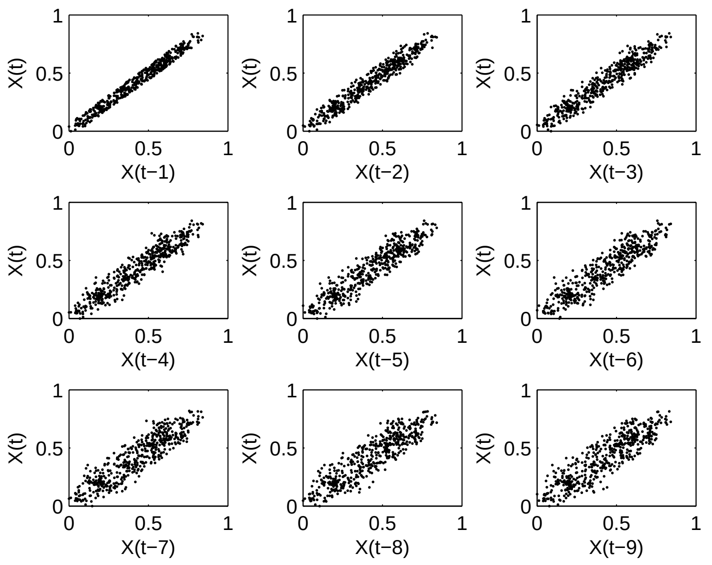

Figure 4: A typical Random Walk Lagplot.

Besides, in order to analyze time lag relations in the studied time series, the graphical methodology proposed by [33, 43], referred to as lagplot [43] or phase portrait [33], was employed. This consists of dispersion graph relating the different time lags of the time series (X_t vs X_{t-1}, X_t vs X_{t-2}, X_t vs X_{t-3}, $\ldots$), and allow observations of possible relative strong relationships in any time lags (when a structure appearance is shown in the graph). Although such technique is very limited, since it depends on human interpretation of the graphs, its simplicity is a strong argument for its utilization [7]. For example, a time series like a random walk model will have a Lagplot just only linear structures, as illustrated in the Figure 4.

5.1 Apple Inc Stock Prices Series

The Figure 5 shows the Apple Inc Stock Prices lagplot, where it is seen that for all the time lags of this series there is a clear nonlinear relationship among the lags.

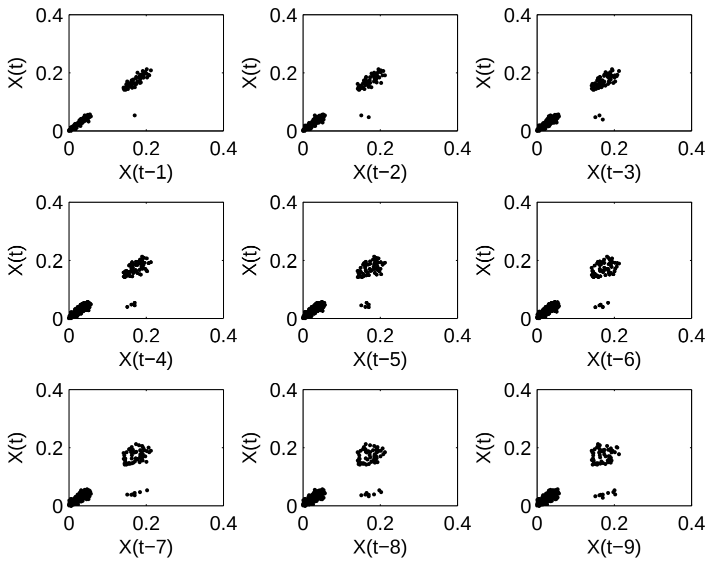

Figure 5: Apple Inc Stock Prices series lagplot.

For the prediction of the Apple Inc Stock Prices series, the proposed model automatically chose the lags 2, 3, 8 and 10 as the relevant lags for the time series representation, defined the second ANN model (Equation 16) with architecture 4-6-1, in which $\mathrm{NetMod} = 2$, $\mathrm{NLags} = 4$ and $\mathrm{NHidden} = 6$, chose the Scaled Conjugate Gradient algorithm for ANN training and classified the model as "out-of-phase" matching. Table 1 shows the results (with respect to the test set) for all the performance measures for the MLP, TAEF and SHIF models.

Table 1: Forecasting performance for the Apple Inc Stock Prices series.

Performance	MLP		TAEF		SHIF	
Measures	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase
MSE	2.8102e-4	2.4082e-4	2.7712e-4	5.6002e-5	2.1419e-4	2.5576e-5
MAPE	2.0674e-2	2.1511e-2	2.0921e-2	9.9201e-3	1.9942e-2	8.0102e-3
NMSE	1.3904	1.1935	1.3750	0.2717	1.0635	0.1267
ARV	1.4342e-2	1.2290e-2	1.4143e-2	2.9030e-3	1.0931e-2	1.3258e-3
POCID(%)	47.21	47.36	47.36	97.78	47.36	99.54
Fitness Function	80.5376	78.7389	80.3514	23.8817	77.3894	12.3816

According to Table 1, the prediction of SHIF out-of-phase model (with the phase fix procedure) obtained a performance much better (in terms of fitness function – 87.6184) than the MLP out-of-phase model (21.2611), and a slightly better result than the TAEF out-of-phase model (76.1183). The obtained NMSE value (0.1267) indicated that the SHIF out-of-phase model had a much better performance than a random walk like model [32] and the MLP out-of-phase model (1.1935), and a slightly better result than the TAEF out-of-phase model (0.2717). The POCID measure (99.54%) shows that the SHIF out-of-phase model had a much better performance than a "coin-tossing" experiment and the MLP out-of-phase model (47.36%), and a slightly better performance than the TAEF out-of-phase model (97.78%).

The Figure 6 shows the actual Apple Inc Stock Prices (solid line) and the predicted values generated by the SHIF model (dashed line with mark point) for the last 100 points of the test set for both cases: in-phase matching and out-of-phase matching.

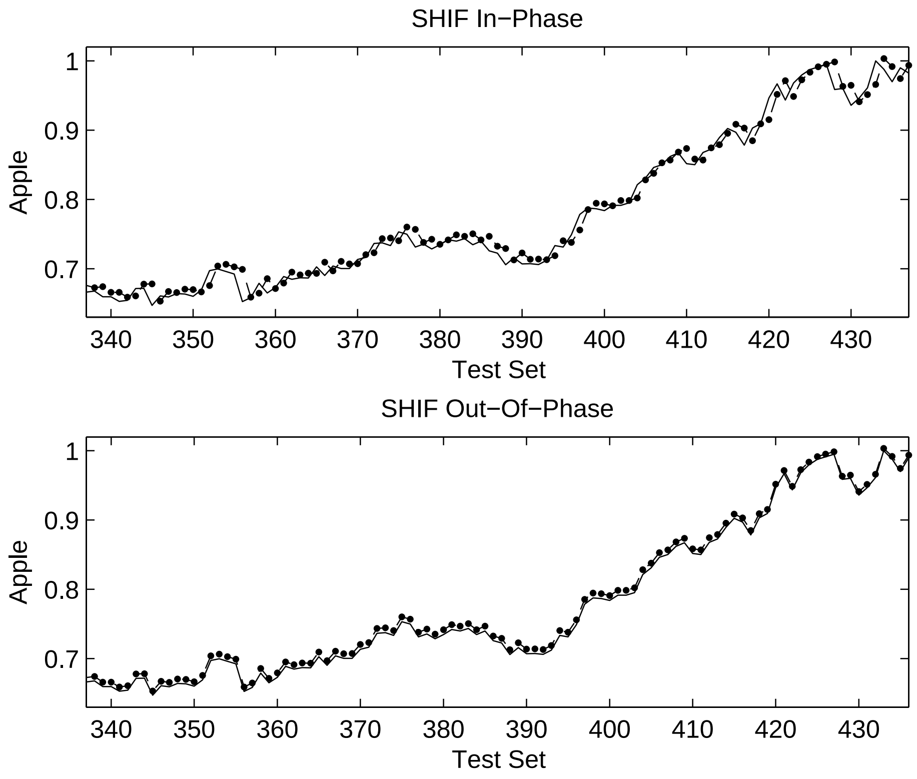

Figure 6: Prediction results for the Apple Inc Stock Prices series (test set). The superior graphic is the in-phase matching model and the inferior graphic is the out-of-phase matching model, where for both cases the solid line is the real value and the dashed line with mark point is the prediction value.

5.2 Applied Materials Inc Stock Prices Series

The Figure 7 shows the Applied Materials Inc Stock Prices lagplot, where it is possible to notice that there is a possible nonlinear relationship among all time lags embedded in the linear structure.

The proposed model selected, for the prediction of the Applied Materials Inc Stock Prices series, the time lags 2, 6 and 10, the second ANN model (Equation 16) with architecture 3-10-1, where $\mathrm{NetMod} = 2$, $\mathrm{NLags} = 3$ and $\mathrm{NHidden} = 10$, the

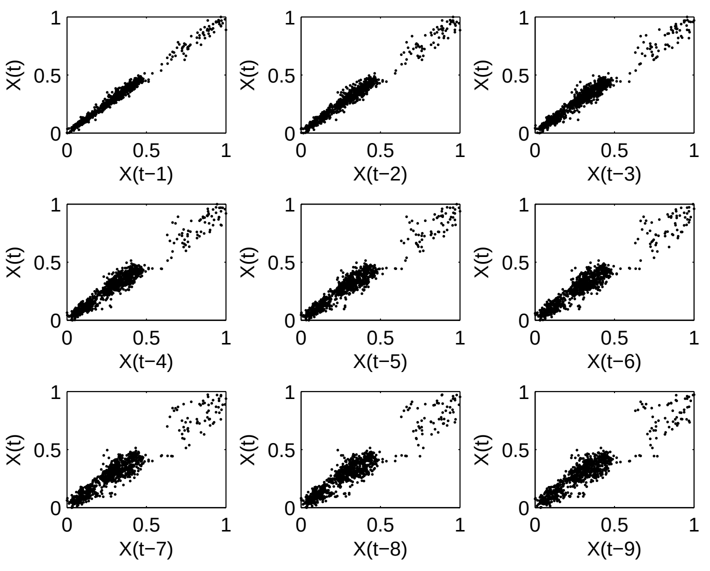

Figure 7: Applied Materials Inc Stock Prices series lagplot.

Levenberg Marquardt algorithm for ANN training and classified the model as "out-of-phase" matching. The results (regarding test set) for all the performance measures for the MLP, TAEF and SHIF models are presented in Table 2.

Table 2: Forecasting Performance for the Applied Materials Inc Stock Prices series.

Performance	MLP		TAEF		SHIF	
Measures	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase
MSE	5.9464e-5	5.9457e-5	5.9572e-5	1.9235e-7	5.9977e-5	2.7779e-8
MAPE	2.7723e-2	2.7718e-2	2.7680e-2	1.7111e-3	2.7771e-2	5.5186e-4
NMSE	0.9893	0.9892	0.9919	3.2271e-3	0.9985	4.6592e-4
ARV	5.1830e-2	5.1824e-2	5.1925e-2	1.6868e-4	5.2278e-2	2.4360e-5
POCID(%)	53.51	54.00	52.36	97.01	53.54	99.16
Fitness Function	74.1662	73.8979	74.7244	3.4829	74.2424	0.9432

It is verified that, according to Table 2, the prediction of SHIF out-of-phase model (with the phase fix procedure) obtained a performance much better (in terms of fitness function – 99.0568) than the MLP out-of-phase model (26.1021), and a slightly better result than the TAEF out-of-phase model (96.5171). Again, the NMSE value (4.6592e-4) and the POCID measure (99.16%) indicates, respectively, that the SHIF out-of-phase model had a much better performance than a random walk like model [32] and it had a much better performance than a "coin-tossing" experiment. It is important to mention that the proposed model also had a much better performance than MLP out-of-phase model (NMSE = 0.9892 and POCID = 54.00%), and a slightly better result than the TAEF out-of-phase model (NMSE = 3.2271e-3 and POCID = 97.01%).

The Figure 8 shows the actual Applied Materials Inc Stock Prices (solid line) and the predicted values generated by the SHIF model (dashed line with mark point) for the last 100 points of the test set, where the predictions generated by the SHIF out-of-phase model is overlapped to real values, being not possible to distinguish the real values of the predicted values by visual analysis in the out-of-phase matching model.

5.3 General Electric Company Stock Prices Series

The Figure 9 shows the General Electric Company Stock Prices lagplot, where there is a clear linear relationship among the lags, but, with the increase in the time lag degree, the appearance of the structure towards graph center and towards the upper corner on the right hand side of the graph can indicate a nonlinear relationship among the lags.

The proposed model for General Electric Company Stock Prices series prediction automatically defined the lags 2 and 3 as the most fitted lags for the time series representation, chose the second ANN model (Equation 16) with architecture 2-1-1, where $\mathrm{NetMod} = 2$, $\mathrm{NLags} = 2$ and $\mathrm{NHidden} = 1$, selected the Levenberg Marquardt algorithm for ANN training and determined the model as "out-of-phase" matching. Table 3 presents the results (subject of the test set) for all the performance measures for the MLP, TAEF and SHIF models.

According to Table 3, it is seen that the prediction of SHIF out-of-phase model (with the phase fix procedure) obtained a performance much better (in terms of fitness function – 96.3542) than the MLP out-of-phase model (22.7730), and a slightly

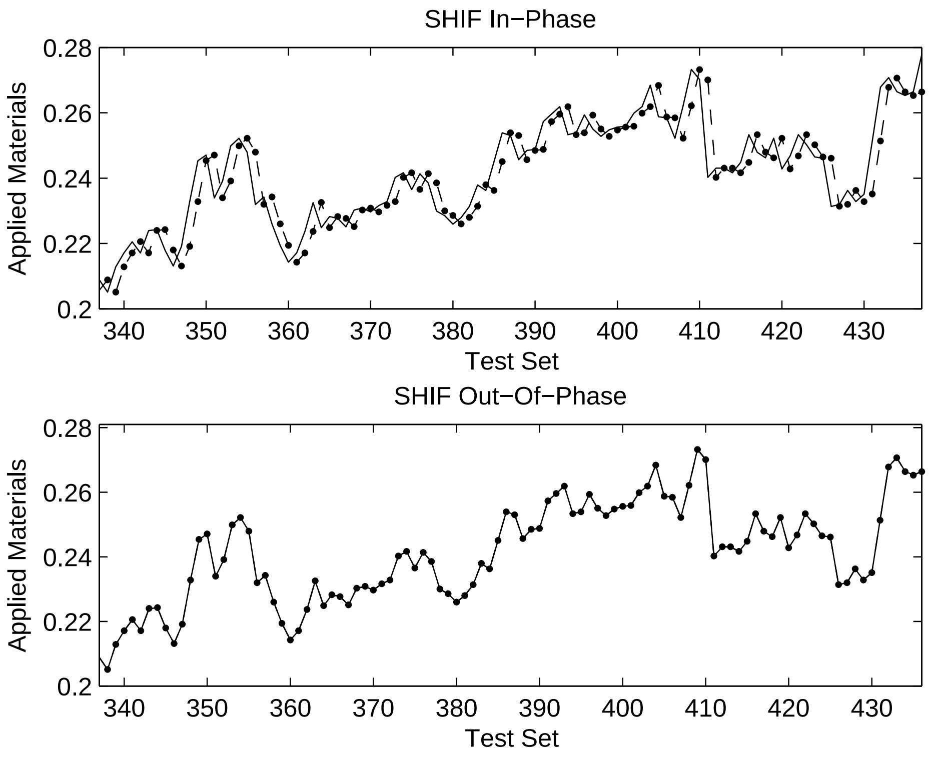

Figure 8: Prediction results for the Applied Materials Inc Stock Prices series (test set). The superior graphic is the in-phase matching model and the inferior graphic is the out-of-phase matching model, where for both cases the solid line is the real value of the time series and the dashed line with mark point is the prediction value.

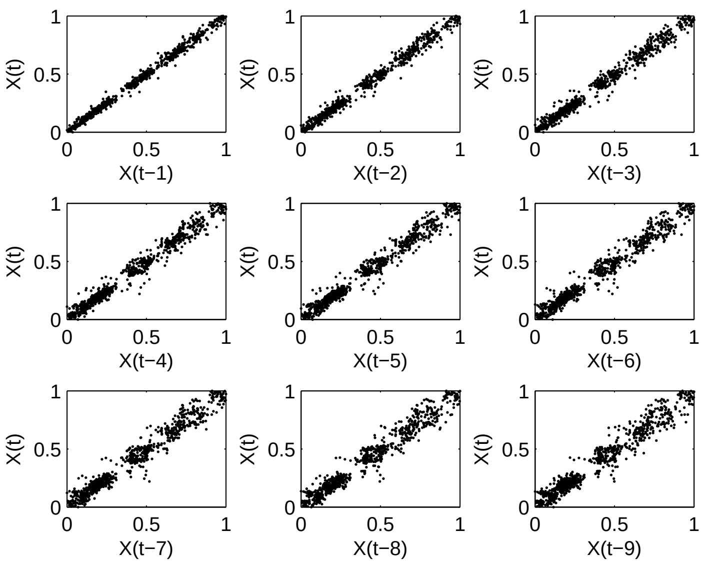

Figure 9: General Electric Company Stock Prices series lagplot.

better result than the TAEF out-of-phase model (90.9583). More one time, the achieved NMSE value (3.2911e-3) suggest that the SHIF out-of-phase model had a much better performance than a random walk like model [32] and the MLP out-of-phase model (1.0111), and a slightly better result than the TAEF out-of-phase model (0.0105). Also, the obtained POCID measure (96.78%) indicates that the SHIF out-of-phase model had a much better performance than a "coin-tossing" experiment and the MLP out-of-phase model (47.14%), and a slightly better performance than the TAEF out-of-phase model (91.97%).

The Figure 10 exhibits the actual General Electric Company Stock Prices (solid line) and the predicted values generated by the SHIF model (dashed line with mark point) for the last 100 points of the test set for the in-phase and out-of-phase matching

Table 3: Forecasting Performance for the General Electric Company Stock Prices series.

Performance	MLP		TAEF		SHIF	
Measures	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase
MSE	6.0449e-5	6.0758e-5	5.9292e-5	6.3877e-7	6.0183e-5	1.9482e-7
MAPE	1.7424e-2	1.7509e-2	1.7371e-2	1.8367e-3	1.7412e-2	9.9381e-4
NMSE	1.0086	1.0111	0.9848	0.0105	1.0030	3.2911e-3
ARV	4.1112e-2	4.1323e-2	4.0326e-2	4.3862e-4	4.0932e-2	1.3378e-4
POCID(%)	46.68	47.14	45.53	91.97	45.99	96.78
Fitness Function	77.4187	77.2270	77.7093	9.1902	77.6900	3.6458

models.

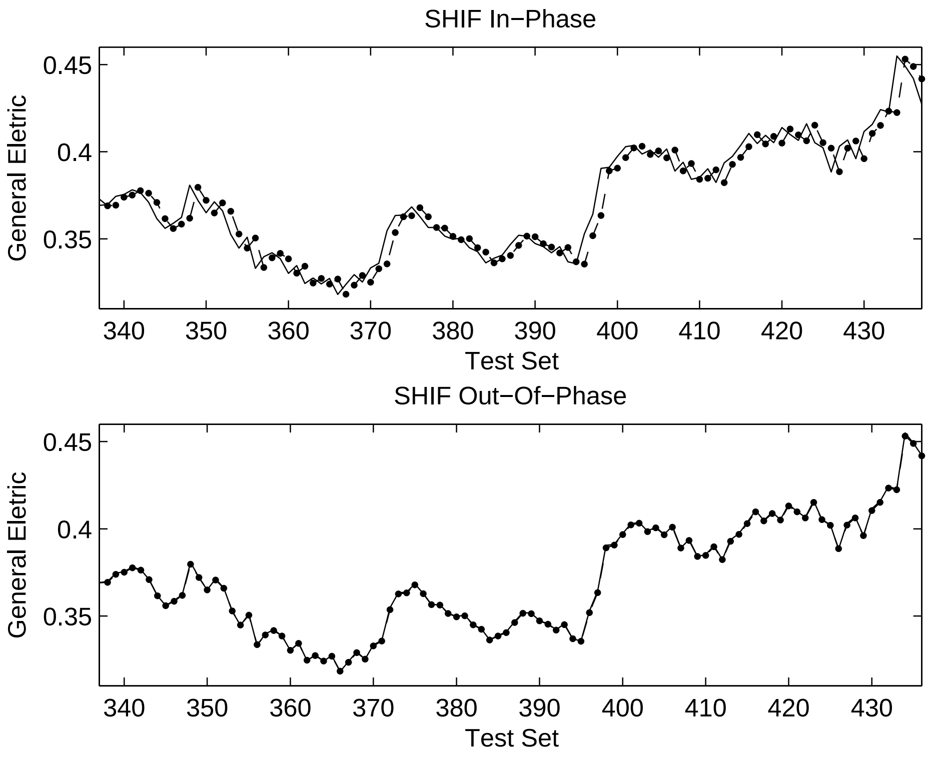

Figure 10: Prediction results for the General Electric Company Stock Prices series (test set). The superior graphic is the in-phase matching model and the inferior graphic is the out-of-phase matching model, where for both cases the solid line is the real value of the time series and the dashed line with mark point is the prediction value.

It is verified that, in Figure 10, the predictions generated by the SHIF out-of-phase model for the General Electric Company Stock Prices series is overlapped to real values. Therefore, it is not possible, through figure analysis, to identify the difference between the real values of the predicted values.

5.4 Hewlett Packard Company Stock Prices Series

The Figure 11 shows the Hewlett Packard Company Stock Prices lagplot, where there is a clear linear relationship among the lags. On the other hand, with the increase in the time lag degree, the appearance of the structure towards graph center and towards the upper corner on the right hand side of the graph can indicate a nonlinear relationship among the lags.

The parameters automatically defined by the proposed model for the prediction of the Hewlett Packard Company Stock Prices series were the lags 2, 3 and 4 as the most fitted lags for the time series representation, the second ANN model (Equation 16) with architecture 3-9-1, in which $\mathrm{NetMod} = 2$, $\mathrm{NLags} = 3$ and $\mathrm{NHidden} = 9$, the Scaled Conjugate Gradient algorithm for ANN training and the model as "out-of-phase" matching. In Table 4 it is possible to show the results (concerning to the test set) for all the performance measures for the MLP, TAEF and SHIF models.

Regarding Table 4, it is verified that the prediction of SHIF out-of-phase model (with the phase fix procedure) obtained a performance much better (in terms of fitness function – 96.6886) than the MLP out-of-phase model (23.9768), and a slightly better result than the TAEF out-of-phase model (88.4729). Once again, the POCID measure (98.39%) shows that the SHIF out-of-phase model had a much better performance than a "coin-tossing" experiment and the MLP out-of-phase model (49.36%), and

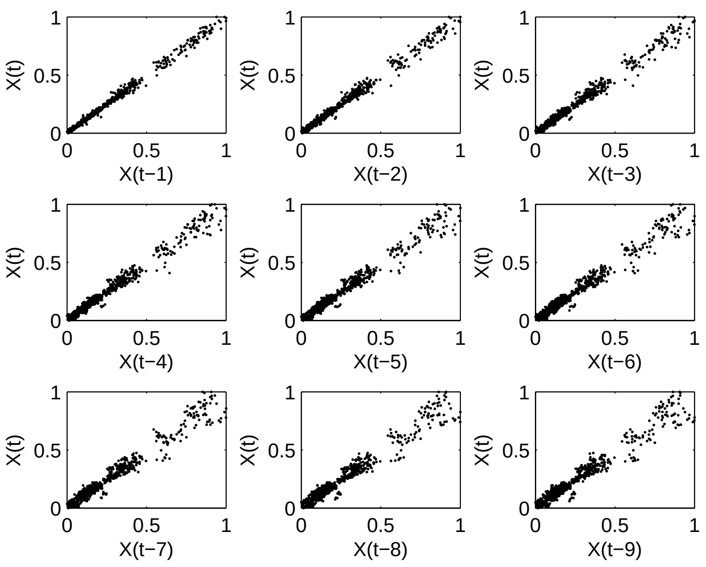

Figure 11: Hewlett Packard Company Stock Prices series lagplot.

Table 4: Forecasting Performance for the Hewlett Packard Company Stock Prices series.

Performance	MLP		TAEF		SHIF	
Measures	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase	In-Phase	Out-Of-Phase
MSE	8.3194e-5	8.2013e-5	8.0441e-5	1.6988e-6	8.2028e-5	1.1907e-6
MAPE	1.6409e-2	1.6384e-2	1.6069e-2	2.4381e-3	1.6198e-2	2.5625e-3
NMSE	1.0477	1.0331	1.0124	0.0213	1.0327	0.0149
ARV	9.2214e-3	9.0904e-3	8.9162e-3	1.8921e-4	9.0921e-3	1.3262e-4
POCID(%)	47.59	49.36	49.19	90.59	47.36	98.39
Fitness Function	77.0475	76.0232	75.8573	11.5271	76.9882	3.3114

a slightly better performance than the TAEF out-of-phase model (90.59%). Again, the obtained NMSE value (0.0149) indicated that the SHIF out-of-phase model had a much better performance than a random walk like model [32] and the MLP out-of-phase model (1.0331), and a slightly better result than the TAEF out-of-phase model (0.0213).

The Figure 12 shows the actual Hewlett Packard Company Stock Prices (solid line) and the predicted values generated by the SHIF model (dashed line with mark point) for the last 100 points of the test set for in-phase and out-of-phase matching models, where the predictions generated by the SHIF out-of-phase model is overlapped to real values. In this way, it is not perceptible, through figure analysis, the difference between the real values of the predicted values.

In general, all generated prediction models using the phase fix procedure to adjust time phase distortions shown forecasting performance much better than the MLP model, and slightly better than the TAEF model. The SHIF method was able to adjust the time phase distortions of all analyzed time series (the prediction generated by the out-of-phase matching hypothesis is not delayed with respect to the original data), while the MLP model was not able to adjust the time phase. This corroborates with the assumption made by Ferreira [7], where he discusses that the success of the phase fix procedure is strongly dependent on an accurate adjustment of the prediction model parameters and on the model itself used for forecasting.

6 CONCLUSION

The Swarm-based Hybrid Intelligent Forecasting (SHIF) method to overcome the random walk dilemma for stock market forecasting, where the predicted values of a given financial time series are shifted one step ahead of the real values, was proposed in this paper. It performs a swarm-based search for the minimum dimension necessary to determine the characteristic phase space that generates the financial time series phenomenon. The proposed SHIF method is inspired on Takens Theorem [24] and consists of an intelligent hybrid system composed of a Particle Swarm Optimizer (PSO) [17] and an Artificial Neural Network (ANN), which searches for (i) the minimum number of relevant time lags necessary to efficiently represent complex time series, (ii) the best evolved neural network structure in terms of the number of hidden processing units and network weights (training and pruning process), and (iii) the appropriate algorithm for training the model (RPROP [25], Levenberg Marquardt [26], Scaled Conjugate Gradient [27] and One Step Secant Conjugate Gradient [28]) that boosts the prediction performance. After training process, the proposed SHIF method chooses the most fitted prediction model for the time series representation, and performs a behavioral statistical test [7] in the attempt to adjust forecasting time phase distortions that frequently appear in financial time

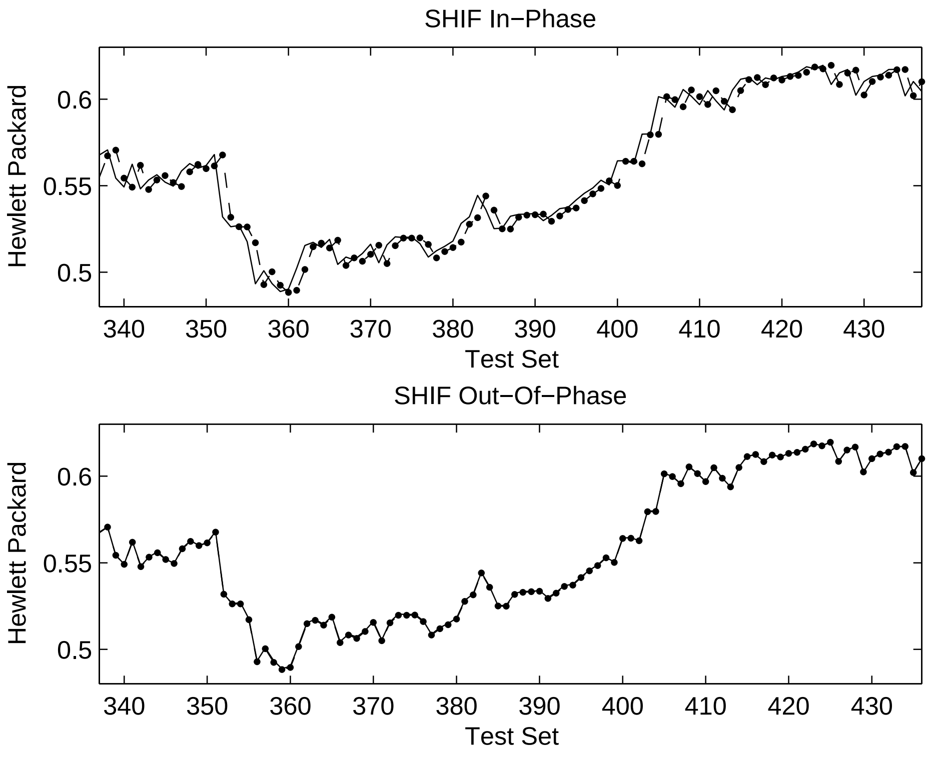

Figure 12: Prediction results for the Hewlett Packard Company Stock Prices series (test set). The superior graphic is the in-phase matching model and the inferior graphic is the out-of-phase matching model, where for both cases the solid line is the real value of the time series and the dashed line with mark point is the prediction value.

series.

Five different metrics were used to measure the performance of the proposed SHIF method for financial time series forecasting, where a new empirical fitness function was built in order to improve the description of the time series phenomenon as better as possible. The five different evaluations measures used to compose this fitness function can have different contributions to final prediction result, where a more sophisticated analysis will must be done to determine the optimal combination of such metrics.

The results were collected with four real world time series from the financial stock market (Apple Inc Stock Prices, Applied Materials Inc Stock Prices, General Electric Company Stock Prices and Hewlett-Packard Company Stock Prices). The experimental results demonstrated a consistent much better performance of the proposed model when compared to the MLP networks, and better performance than the TAEF method model [7], which had previously shown boosted performance enhancement when compared to other results found in the literature [1–5, 9, 12, 13, 44, 45].

It was also observed that the SHIF model obtained a much better performance than a random walk model [32] for the analyzed financial time series, overcoming the random walk dilemma (where the predicted values are shifted one step ahead of the original values).

The out-of-phase behavior reached by the models appears mostly when the time series is a financial series [7]. Although the out-of-phase behavior is characterized by a prediction shift (delay) with respect to original data (and this is a random walk like behavior), the models generated by the SHIF method are not random walk models (although they behave with a characteristic similar behavior). This observation is supported by the fact that the phase fix procedure in the SHIF model was able to correct the one step delay distortion. If the models generated by the SHIF method were reals random walk models, the phase fix procedure would generate similar results of the original predictions, because in the random walk model the $t + 1$ value is always the t value (Section 2.2).

Such facts can be observed with respect to relationship between the linear and nonlinear structures indicated by lagplot analysis, where for time series with clear nonlinear structure embedded in time lags, the proposed method can reach a so close estimation of real time series values, overcoming the random walk dilemma. But, if such nonlinear structures have a small amplitude when compared with the random walk component the SHIF method cannot reach same improved performance, has a slight decreasing in prediction performance results when compared to other analyzed times series. However, even with this limiting factor, the SHIF method was still capable of to adjust the time phase distortion, being able to break down the random walk dilemma.

While the SHIF model was able to adjust the time-phase delay, the MLP models were not capable to produce such correction behavior although the same procedure was applied to all the models. A feasible explanation for such phenomenon is that the phase fix procedure will depend on the information complexity contained in the time series data and the ability to accurately define the best prediction model parameters to estimate the real time series values, in other words, the success of the phase fix procedure is strongly dependent on an accurate adjustment of the prediction model parameters and on the model itself used for forecasting.

Finally, the results shown that the phase fix procedure was able to correct more efficiently the prediction time phase of the proposed SHIF model when compared to TAEF model. Further studies are being developed to better formalize and explain the properties of the SHIF model and to determine possible limitations of the method with other financial time series with components such as trends, seasonalities, impulses, steps and other nonlinearities. Also, further studies, in terms of risk and financial return, are being developed in order to determine the additional economical benefits, for an investor, with the use of the SHIF method.

References

[1] G. E. P. Box, G. M. Jenkins and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice Hall, New Jersey, third edition, 1994.

[2] T. S. Rao and M. M. Gabr. *Introduction to Bispectral Analysis and Bilinear Time Series Models*, volume 24 of *Lecture Notes in Statistics*. Springer, Berlin, 1984.

[3] T. Ozaki. *Nonlinear Time Series Models and Dynamical Systems*, volume 5 of *Hand Book of Statistics*. North-Holland, Amsterdam, 1985.

[4] M. B. Priestley. *Non-Linear and Non-stationary Time Series Analysis*. Academic Press, 1988.

[5] D. E. Rumelhart and J. L. McCleland. *Parallel Distributed Processing, Explorations in the Microstructure of Cognition*, volume 1 & 2. MIT Press, 1987.

[6] M. P. Clements, P. H. Franses and N. R. Swanson. "Forecasting economic and financial time-series with non-linear models". *International Journal of Forecasting*, vol. 20, no. 2, pp. 169–183, 2004.

[7] T. A. E. Ferreira, G. C. Vasconcelos and P. J. L. Adeodato. "A New Intelligent System Methodology for Time Series Forecasting with Artificial Neural Networks". In *Neural Processing Letters*, volume 28, pp. 113–129, 2008.

[8] F. H. F. Leung, H. K. Lam, S. H. Ling and P. K. S. Tam. "Tuning of the Structure and Parameters of the Neural Network Using an Improved Genetic Algorithm". *IEEE Transactions on Neural Networks*, vol. 14, no. 1, pp. 79–88, January 2003.

[9] M. Matilla-García and C. Argüello. "A hybrid approach based on neural networks and genetic algorithms to the study of profitability in the Spanish Stock Market". *Applied Economics Letters*, vol. 12, no. 5, pp. 303–308, April 2005.

[10] J. H. Holland. *Adaptation in Natual and Artificial Systems*. University of Michigan Press, Michigan, 1975.

[11] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[12] R. A. Araújo, F. Madeiro, R. P. Sousa, L. F. C. Pessoa and T. A. E. Ferreira. "An Evolutionary Morphological Approach for Financial Time Series Forecasting". In *Proceedings of the IEEE Congress on Evolutionary Computation*, Vancouver, Canada, 2006.

[13] R. A. Araújo, R. P. Sousa and T. A. E. Ferreira. "An Intelligent Hybrid Approach for Designing Increasing Translation Invariant Morphological Operators for Time Series Forecasting". In *ISNN (2)*, volume 4492 PART II of *Lecture Notes in Computer Science*, pp. 602–611. Springer-Verlag, 2007.

[14] R. P. Sousa. "Design of translation invariant operators via neural network training". Ph.D. thesis, UFPB, Campina Grande, Brazil, 2000.

[15] G. Matheron. *Random Sets and Integral Geometry*. Wiley, New York, 1975.

[16] G. J. F. Banon and J. Barrera. "Minimal Representation for Translation Invariant Set Mappings by Mathematical Morphology". *SIAM J. Appl. Math.*, vol. 51, no. 6, pp. 1782–1798, 1991.

[17] F. vandenBergh and A. P. Engelbrecht. "A Cooperative Approach to Particle Swarm Optimization." *IEEE Trans. Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[18] R. C. Eberhart and J. Kennedy. "A new optimizer using particle swarm theory". In *Proceedings of the Int. Symp. Micro Machine and Human Science*, Nagoya, Japan, 1995.

[19] R. C. Eberhart and X. Hu. "Human tremor analysis using particle swarm optimization". In *Proceedings of the IEEE Congress on Evolutionary Computation*, Washington, USA, 1999.

[20] A. P. Engelbrecht and A. Ismail. "Training product unit neural networks". In *Stability Control: Theory Appl.*, volume 2, pp. 59–74, 1999.

[21] F. van den Bergh and A. P. Engelbrecht. "Cooperative learning in neural networks using particle swarm optimizers". In *South African Comput. J.*, volume 26, pp. 84–90, 2000.

[22] Y. Shi and R. C. Eberhart. "A modified particle swarm optimizer". In *Proceedings of the IEEE Congress on Evolutionary Computation*, Anchorage, AK, 1998.

[23] Y. Shi and R. C. Eberhart. "Empirical study of particle swarm optimization". In *Proceedings of the IEEE Congress on Evolutionary Computation*, Washington, USA, 1999.

[24] F. Takens. "Detecting Strange Attractor in Turbulence". In *Dynamical Systems and Turbulence*, edited by A. Dold and B. Eckmann, volume 898 of *Lecture Notes in Mathematics*, pp. 366–381, New York, 1980. Springer-Verlag.

[25] M. Riedmiller and H. Braun. "A direct adaptive method for faster backpropagation learning: The RPROP algorithm". In *Proceedings of the IEEE Int. Conf. on Neural Networks (ICNN)*, pp. 586–591, San Francisco, 1993.

[26] M. Hagan and M. Menhaj. "Training feedforward networks with the Marquardt algorithm". *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989–993, November 1994.

[27] M. F. Moller. "A scaled conjugate gradient algorithm for fast supervised learning". *Neural Networks*, vol. 6, pp. 525–533, 1993.

[28] R. Battiti. "One Step Secant Conjugate Gradient". *Neural Computation*, vol. 4, pp. 141–166, 1992.

[29] R. Savit and M. Green. "Time Series and Dependent Variables". *Physica D*, vol. 50, pp. 95–116, 1991.

[30] H. Pi and C. Peterson. "Finding the Embedding Dimension and Variable Dependences in Time Series". *Neural Computation*, vol. 6, pp. 509–520, 1994.

[31] N. Tanaka, H. Okamoto and M. Naito. "Estimating the Active Dimension of the Dynamics in a Time Series Based on a Information Criterion". *Physica D*, vol. 158, pp. 19–31, 2001.

[32] T. C. Mills. *The Econometric Modelling of Financial Time Series*. Cambridge University Press, Cambridge, 2003.

[33] H. Kantz and T. Schreiber. *Nonlinear Time Series analysis*. Cambridge University Press, New York, NY, USA, second edition, 2003.

[34] S. Haykin. *Neural networks: A comprehensive foundation*. Prentice Hall, New Jersey, 1998.

[35] J. Sietsma and R. Dow. "Neural net pruning – Why and how?" In *Proceedings of the IEEE International Conference on Neural Networks*, volume 1, pp. 325–333, 1988.

[36] E. D. Karnin. "A simple procedure for pruning back-propagation trained neural networks". In *Proceedings of the IEEE International Conference on Neural networks*, volume 1, pp. 239–245, 1990.

[37] A. S. Weigend, D. E. Rumelhart and B. A. Huberman. "Generalization by Weight-Elimination with Application to Forecasting". In *Advances in Neural Information Processing Systems – NIPS*, volume 3, pp. 875–882, 1991.

[38] R. Reed. "Pruning algorithms - A survey". *IEEE Transaction on Neural Networks*, vol. 4, no. 5, pp. 740–747, 1993.

[39] M. Crottel, B. Girard, Y. Girard, M. Mangeas and C. Muller. "Neural modeling for time series: a statistical stepwise method for weight elimination". *IEEE Transaction on Neural Networks*, vol. 6, no. 6, pp. 1355–1364, 1995.

[40] M. P. Clements and D. F. Hendry. "On the Limitations of Comparing Mean Square Forecast Errors". *Journal of Forecasting*, vol. 12, no. 8, pp. 617–637, Dec. 1993.

[41] L. Prechelt. "Proben1: A set of Neural Network Benchmark Problems and Benchmarking Rules". Technical Report 21/94, 1994.

[42] T. H. Hann and E. Steurer. "Much ado about nothing? Exchange rate forecasting: Neural networks vs. linear models using monthly and weekly data". *Neurocomputing*, vol. 10, pp. 323–339, 1996.

[43] D. B. Percival and A. T. Walden. *Spectral Analysis for Physical Applications – Multitaper and Conventional Univariate Techniques*. Cambridge University Press, New York, 1998.

[44] J. Binner, G. Kendall and A. Gazely. "Evolving Neural Networks with Evolutionary Strategies: A New Application to Divisia Money". *Advances in Econometrics*, vol. 19, pp. 127–143, 2004.

[45] R. A. Araújo, G. C. Vasconcelos and T. A. E. Ferreira. "Hybrid Differential Evolutionary System for Financial Time Series Forecasting". In *Proceedings of the IEEE Congress on Evolutionary Computation*, Singapore, 2007.