Innovative Applications of O.R

Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and particle swarm optimization

Georgios Sermpinis, Konstantinos Theofilatos, Andreas Karathanasopoulos, Efstratios F. Georgopoulos, Christian Dunis

PII: S0377-2217(12)00766-7
DOI: http://dx.doi.org/10.1016/j.ejor.2012.10.020
Reference: EOR 11325

To appear in: *European Journal of Operational Research*

Received Date: 30 June 2012
Accepted Date: 19 October 2012

Please cite this article as: Sermpinis, G., Theofilatos, K., Karathanasopoulos, A., Georgopoulos, E.F., Dunis, C., Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and particle swarm optimization, *European Journal of Operational Research* (2012), doi: http://dx.doi.org/10.1016/j.ejor.2012.10.020

Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and particle swarm optimization

by
Georgios Sermpinis*
Konstantinos Theofilatos**
Andreas Karathanasopoulos***
Efstratios F. Georgopoulos****
Christian Dunis *****

October 2012

Abstract

The motivation for this paper is to introduce a hybrid Neural Network architecture of Particle Swarm Optimization and Adaptive Radial Basis Function (ARBF-PSO), a time varying leverage trading strategy based on Glosten, Jagannathan and Runkle (GJR) volatility forecasts and a Neural Network fitness function for financial forecasting purposes. This is done by benchmarking the ARBF-PSO results with those of three different Neural Networks architectures, a Nearest Neighbors algorithm (k-NN), an autoregressive moving average model (ARMA), a moving average convergence/divergence model (MACD) plus a naïve strategy. More specifically, the trading and statistical performance of all models is investigated in a forecast simulation of the EUR/USD, EUR/GBP and EUR/JPY ECB exchange rate fixing time series over the period January 1999 to March 2011 using the last two years for out-of-sample testing.

As it turns out, the ARBF-PSO architecture outperforms all other models in terms of statistical accuracy and trading efficiency for the three exchange rates.

Keywords

Adaptive Radial Basis Function, Partial Swarm Optimization, Forecasting, Quantitative Trading Strategies

* **Georgios Sermpinis** University of Glasgow Business School, University of Glasgow, Adam Smith Building, Glasgow, G12 8QQ, UK (E-mail: georgios.sermpinis@glasgow.ac.uk)
** **Konstantinos Theofilatos** Department of Computer Engineering & Informatics, University of Patras, 26500, Patras, Greece (E-mail: theofilk@ceid.upatras.gr)
*** **Andreas Karathanasopoulos** London Metropolitan Business School, London Metropolitan University, 31 Jewry Street, London, EC3N 2EY, UK (E-mail: a.karathanasopoulos@londonmet.ac.uk)
**** **Efstratios Georgopoulos**, Technological Educational Institute of Kalamata, 24100, Kalamata, Greece, (Email: sfg@teikal.gr)
***** **Christian Dunis** Horus Partners Wealth Management Group SA, Genève, Suisse and Liverpool Business School, JMU, John Foster Building, Liverpool L3 5UZ (E-mail: christian.dunis@hpwmg.com)

1. INTRODUCTION

Neural networks (NN) are an emergent technology with an increasing number of real-world applications including operational research (Lisboa and Vellido (2000) and Zhang *et. al.* (1998)). However their numerous limitations and contradictory empirical evidence around their forecasting power are often creating scepticism about their use among practitioners. This scepticism is further fuelled by the fact that the selection of each algorithm parameters and inputs is based more on trial and error and the practitioner's market knowledge rather than on some formal statistical procedure.

The motivation for this paper is to introduce in Operational Research a hybrid Neural Network architecture of Particle Swarm Optimization and Adaptive Radial Basis Function (ARBF-PSO), which try to overcome some of these limitations. More specifically our proposed architecture is fully adaptive something that decreases the numbers of parameters that the practitioner needs to experiment while on the other hand it increases the forecasting ability of the network. The proposed methodology is superior in comparison to the application of meta-heuristic methods (PSO, Genetic Algorithms, Swarm Fish Algorithm) that have been already presented in the literature (Nekoukar and Beheshti (2010) and Shen *et al.* (2011)) because it eradicates the risk of getting trapped into local optima and the final solution is assured to be optimal for a subset of the training set.

In our study we benchmark our proposed algorithm with a Multi-Layer Perceptron (MLP), a Recurrent Neural Network (RNN), a Psi Sigma Neural Network (PSI), a Nearest Neighbors algorithm (k-NN), an autoregressive moving average model (ARMA), a moving average convergence/divergence model (MACD) plus a naïve strategy in a forecasting and trading simulation of the EUR/USD, the EUR/GBP and the EUR/JPY European Central Bank (ECB) daily fixing. The main reason behind our decision to use the ECB daily fixings is that it is possible to leave orders with a bank and trade on that basis. It is therefore a tradable quantity which makes our trading simulation more realistic. We examine the exchange rates from their first trading day since the end of April 2011. Moreover, the nonlinearities and the high complexity of the exchange rates series make them perfect for a forecasting exercise. Nevertheless our proposed methodology can be applied to any forecasting task irrespective the nature of the series under study.

Moreover, we introduce a time-varying leverage trading strategy based on GJR (1993) model volatility forecasts and examine if its application can increase the trading efficiency of our models.

We also introduce a fitness function for our NNs that not only minimize the MSE of our forecasts but also increase their profitability. This is crucial in financial applications where statistical accuracy is not always synonymous with financial profitability of the derived forecasts.

As it turns out the ARBF-PSO algorithm does remarkably well and outperforms all other models in terms of statistical accuracy and trading efficiency for the time series and period under study. It seems that its adaptability and flexibility allows it to outperform in our forecasting competition compared with the more 'traditional' k-NN, MLP, RNN and PSI models. These results provide the first empirical evidence around the utility of the ARBF-PSO in finance and forecasting.

The rest of the paper is organised as follows. In section 2 we present some relevant recent applications in forecasting and section 3 describes the dataset used for this research and its characteristics. An overview of the proposed model and the NN and statistical benchmarks is given in section 4. Section 5 gives the empirical results of all the models considered and investigates the possibility of improving their performance with the introduction of a sophisticated trading strategy while section 6 provides some concluding remarks.

2. LITERATURE REVIEW

Developing high accuracy techniques for predicting time series is a very crucial problem for scientists and decision makers. The traditional statistical methods seem to fail to capture the discontinuities, the nonlinearities and the high complexity of datasets such as financial time series. Complex machine learning techniques like Artificial Neural Networks (NNs) provide enough learning capacity and are more likely to capture the complex non-linear models which are dominant in the financial markets but their parameter tuning remains difficult and generalization problems exist (Donaldson and Kamstra (1996) and Lisboa and Vellido (2000)).

The main objective of this paper is to introduce a novel hybrid method which is able to overcome the difficulties in tuning the parameters of artificial neural networks. For this purpose among the various neural network techniques, we use the Radial Basis Function Neural Networks (RBFNN) which has proven experimentally to outperform the more classical NNs architectures (Broomhead and Lowe (1988)). The hybrid method combines the RBFNNs with Particle Swarm Optimization (PSO) algorithm, a state-of-the art heuristic optimization technique (Kennedy and Eberhart (1995)) in a way that optimizes the neural networks parameters,

structure and training procedure. Our proposed methodology is an extension of the algorithm proposed by Ding *et. al.* (2005) for forecasting purposes.

The proposed methodology has not been applied in science yet. However, two approaches have been recently proposed for the optimization of RBF Neural Networks and their application in financial time-series forecasting. Nekoukar and Beheshti (2010) propose the application of a modified PSO (using hunter particles to increase diversity) for training Radial Basis Functions. This methodology was applied for the prediction of the price of Iranian stock time-series. Despite the high prediction accuracy of the derived model, this hybrid technique does not provide any method for optimizing the structure of the RBF network. Moreover, the applied PSO algorithm uses constant parameters, which requires an extra time-consuming optimization step. Shen *et al.* (2011) introduce a novel hybrid technique which applies an Artificial Fish Swarm algorithm to train Radial Basis Function Neural Networks for modeling the Shanghai Composite Indices. The prediction results are extremely good, but the artificial fish swarm algorithm is not used for the optimization of the RBF network's structure and it requires some parameters to be tuned via a time consuming trial and error approach. Compared to a simple genetic algorithm and a simple PSO method which are also used to train Radial Basis Function Neural Networks, the Artificial Fish Swarm algorithm produces a slightly higher prediction error but the authors believe that being a new intelligent algorithm it has room for improvement and development. Both of these methods use Mean Square Error as a fitness function and they are not specialized for the prediction of financial time series contrary to our proposed methodology.

Several scientists have applied other NNs algorithms to the task of forecasting financial series with ambiguous empirical evidence. Fulcher *et. al.* (2006) apply Higher Order Neural Networks in forecasting the AUD/USD exchange rate with a 90% accuracy. Panda and Narasimhan (2007) use a single hidden layer feedforward NN to produce statistical accurate forecasts of the INR/USD exchange rate having several linear autoregressive models as benchmarks while Andreou *et. al.* (2008) use NNs to forecast and trade European options with disappointing results. On the other hand, Kiani and Kastens (2008) forecast the GBP/USD, the CAD/USD and the JPY/USD exchange rates with feedforward and recurrent NNs having as benchmarks several ARMA models. In their application, NNs outperform in statistical terms their ARMA benchmarks in forecasting the GBP/USD and USD/JPY but not in forecasting the USD/CAD exchange rate. Yang *et. al.* (2008) employ a NN and other regression techniques to examine the potential martingale behaviour of Euro exchange rates in the context of out-of-sample forecasts. The overall evidence indicates that, while martingale behaviour cannot be rejected for Euro

exchange rates with major currencies such as the Japanese yen, British pound, and US dollar, there is nonlinear predictability in terms of economic criteria with respect to several smaller currencies (such as the Australian dollar, the Canadian dollar and the Swiss franc). Bekiros and Georgoutsos (2008) forecast and trade successfully the NASDAQ index with RNNs and Yang *et. al.* (2010) study the predictability of eighteen stock indexes with NNs and linear models. Their models demonstrate low predictability when the data snooping bias was considered. On the same year while on the same year Huck (2010) combines NNs with a multi-criteria decision making method in a S&P 100 stock pair trading application with good results. Adeodato *et. al.* (2011) won the NN3 Forecasting Competition problem with an innovative approach based on the use of median for combining MLP forecasts and Matias and Reboredo (2012) forecast successfully with NNs and other nonlinear models intraday stock market returns. In a forecasting competition, Dunis *et. al.* (2010 and 2011) and Sermpinis *et. al.* (2012) compare several Higher Order NNs and autoregressive models in forecasting and trading the EUR exchange rates. Their results demonstrate the forecasting superiority of a class of NNs, the Psi Sigma, which are able to capture higher order correlation within their dataset. Bekiros (2010) introduced a promising hybrid neurofuzzy system which forecast accurately the direction of the market for 10 of the most prominent stock indices of U.S.A, Europe and Southeast Asia and Dhamija and Bhalla (2011) apply several variants of the MLP and RBF networks to the task of forecasting five different exchange rates with good results. On the same year Wang *et. al.* (2011) forecast successfully the Shenzhen Integrated Index and the Dow Jones Industrial Average Index with a hybrid NN model. Compared to the above mentioned studies, our proposed algorithm is fully adaptive and enables us to avoid the time consuming and risky process of optimizing the parameters of our networks through a sensitivity analysis in the in-sample period. However, we apply some of the most promising architectures of the previous mentioned paper such as the Psi Sigma (Fulcher *et. al.* (2006), Dunis *et. al.* (2010 and 2011) and Sermpinis *et. al.* (2012)), the RNN (Kiani and Kastens (2008) and Bekiros and Georgoutsos (2008)) and the MLP (Panda and Narasimhan (2007), Adeodato *et. al.* (2011) and Dhamija and Bhalla (2011)) as benchmarks to our proposed model.

3. THE EUR/USD AND EUR/GBP EXCHANGE RATES AND RELATED FINANCIAL DATA

The European Central Bank (ECB) publishes a daily fixing for selected EUR exchange rates: these reference mid-rates are based on a daily concentration procedure between central banks within and outside the European System of Central Banks, which normally takes place at 2.15 p.m. ECB time. The reference exchange rates are published both by electronic market

information providers and on the ECB's website shortly after the concentration procedure has been completed. Although only a reference rate, many financial institutions are ready to trade at the EUR fixing and it is therefore possible to leave orders with a bank for business to be transacted at this level.

The ECB daily fixings of the EUR exchange rates are therefore tradable levels which makes using them a more realistic alternative to, say, London closing prices and these are the series that we investigate in this paper. We examine the ECB daily fixings of the EUR/USD, the EUR/GBP and the EUR/JPY since their first trading day on 4 January 1999 until 29 April 2011. The data period is partitioned as table 1.

Name of period		Beginning	End
Total dataset	3158	4 January 1999	29 April 2011
Training dataset	2645	4 January 1999	30 April 2009
Out-of-sample dataset [Validation set]	513	4 May 2009	29 April 2011

Table 1: The total dataset

The figure 1 below shows the total dataset for the three exchange rates under study.

Fig. 1: EUR Frankfurt daily fixing prices (total dataset)

The three observed time series are non-normal (the Jarque-Bera statistics confirms this at the 99% confidence level) containing slight skewness and high kurtosis. They are also nonstationary and hence we decided to transform them into a stationary daily series of rates of return[1] using the formula:

$$R_t = \ln\left(\frac{P_t}{P_{t-1}}\right) \qquad [1]$$

Where R_t is the rate of return and P_t is the price level at time t.

The summary statistics of the EUR/USD, EUR/GBP and EUR/JPY returns series reveal positive skewness and high kurtosis. The Jarque-Bera statistic confirms again that the two return series are non-normal at the 99% confidence level. These two return series will be forecasted from our models.

In the absence of any formal theory behind the selection of the inputs of a neural network, we conduct neural networks experiments and a sensitivity analysis on a pool of potential inputs in the training dataset in order to help our decision. Based on these experiments and the sensitivity analysis we select as inputs the sets of variables that provide the higher trading performance for each network in the in-sample period. This set of inputs (which is different for each network and series under study) is consisted by a set of autoregressive terms of the three exchange rates under study. We also explored autoregressive terms of other exchange rates as inputs (e.g. the ECB fixing of the EUR/CHF), commodities prices (e.g. Gold Bullion and Brent Oil) and stock market prices (e.g. the S&P 500 index). However, they did not seem to add any value during our sensitivity analysis.[2]

In order to train our neural networks we further divide our dataset as in table 2:

Name of period	Trading days	Beginning	End
Total dataset	3158	4 January 1999	29 April 2011
Training data set	2134	4 January 1999	30 April 2007
Test data set	511	2 May 2007	30 April 2009
Out-of-sample data set [Validation set]	513	4 May 2009	29 April 2011

Table 2: The neural networks datasets

4. FORECASTING MODELS

4.1 Statistical/Technical Models

In this paper, we benchmark our ARBF-PSO model with 3 different NNs, a Nearest Neighbors algorithm (k-NN) algorithm and 3 traditional strategies, namely an autoregressive moving average model (ARMA), a moving average convergence/divergence technical model (MACD) and a naïve strategy. The performance of our models is evaluated in terms of statistical accuracy and trading performance via a simulated trading strategy.

4.1.1 Naïve strategy

[1] Confirmation of its stationary property is obtained at the 1% significance level by both the Augmented Dickey Fuller (ADF) and Phillips-Perron (PP) test statistics.

[2] The 12 different sets of inputs for our 4 different NN architectures and 3 series under study are not presented here for the sake of space and are available upon request.

The naïve strategy simply takes the most recent period change as the best prediction of the future change. The model is defined by[3]:

$$\hat{Y}_{t+1} = Y_t \qquad [2]$$

where Y_t is the actual rate of return at period t

$\hat{Y}_{t+1}$ is the forecast rate of return for the next period

4.1.2 Moving Average

A moving average model is defined as:

$$M_t = \frac{(Y_t + Y_{t-1} + Y_{t-2} + \ldots + Y_{t-n+1})}{n} \qquad [3]$$

where M_t is the moving average at time t

n is the number of terms in the moving average

Y_t is the actual rate of return at period t

The MACD strategy used is quite simple. Two moving average series are created with different moving average lengths. The decision rule for taking positions in the market is straightforward. Positions are taken if the moving averages intersect. If the short-term moving average intersects the long-term moving average from below a 'long' position is taken. Conversely, if the long-term moving average is intersected from above a 'short' position is taken[4].

The forecaster must use judgement when determining the number of periods on which to base the short-term and long term moving averages. The combinations that performed best over the in-sample sub-period were retained for out-of-sample evaluation. The models selected were a combination of (1,8) for the EUR/USD, (3,6) for the EUR/GBP and (2,5) for the EUR/JPY exchange rate.

4.1.3 ARMA model

Autoregressive moving average models (ARMA) assume that the value of a time series depends on its previous values (the autoregressive component) and on previous residual values (the moving average component)[5].

The ARMA model takes the form:

$$Y_t = \phi_0 + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \varepsilon_t - w_1 \varepsilon_{t-1} - w_2 \varepsilon_{t-2} - \ldots - w_q \varepsilon_{t-q} \qquad [4]$$

where Y_t is the dependent variable at time t

[3] We also applied a simple random walk model to our series ($\hat{Y}_{t+1} = \lambda + \varepsilon_t$ where $\varepsilon_t \sim N(0,1)$ and λ is the in-sample mean). The performance of the random walk model was similar or slightly worse from our naïve strategy as it is described in equation [2] in the in-sample period and is available upon request.

[4] For example, a 'long' EUR/USD position means buying Euros at the current price, while a 'short' position means selling Euros at the current price.

[5] For a full discussion on the procedure, refer to Box *et al.*, (1994).

Y_{t-1}, Y_{t-2}, and Y_{t-p}	are the lagged dependent variable
ϕ_0, ϕ_1, ϕ_2, and ϕ_p	are regression coefficients
ε_t	is the residual term
ε_{t-1}, ε_{t-2}, and ε_{t-p}	are previous values of the residual
w_1, w_2, and w_q	are weights.

Using as a guide the correlogram and the information criteria in the training and the test sub periods we have chosen a restricted ARMA (5,5) for the EUR/USD and a restricted ARMA (6,6) for the EUR/GBP and a restricted ARMA (4,4) for the EUR/JPY exchange rate. In all models all their coefficients are significant at the 95% confidence interval as the null hypothesis that all coefficients (except the constant) are not significantly different from zero is rejected at the 95% confidence interval.

The selected ARMA models for the EUR/USD, EUR/GBP and EUR/JPY exchange rates are presented in equations [5], [6] and [7] respectively:

$$Y_t = 0.00071 + 0.49349Y_{t-1} - 0.53617Y_{t-2} + 0.31984Y_{t-3} - 0.40105Y_{t-5} - 0.48310\varepsilon_{t-1} + 0.49545\varepsilon_{t-2} - 0.285219\varepsilon_{t-3} + 0.43699\varepsilon_{t-5} \quad [5]$$

$$Y_t = 0.00011 - 0.42524Y_{t-1} - 0.43358Y_{t-2} + 0.53005Y_{t-6} + 0.46933\varepsilon_{t-1} + 0.44556\varepsilon_{t-2} - 0.49460\varepsilon_{t-6} \quad [6]$$

$$Y_t = 0.00036 - 0.87062Y_{t-1} + 1.02907Y_{t-3} + 0.57035Y_{t-4} + 0.89816\varepsilon_{t-1} - 1.056206\varepsilon_{t-3} - 0.60691\varepsilon_{t-4} \quad [7]$$

The model selected was retained for out-of-sample estimation.

4.1.4 Nearest Neighbours Model

Nearest Neighbours is a nonlinear and non-parametric forecasting method. It is based on the idea that pieces of time series in the past have patterns which might have resemblance to pieces in the future. Similar patterns of behaviour are located in terms of nearest neighbours using a distance called the Euclidean distance and these patterns are used to predict behaviour in the immediate future. It only uses local information to forecast and makes no attempt to fit a model to the whole time series at once. It is similar to technical analysis as it tries to find out patterns based on certain parameters. The user defines parameters such as the number of neighbours *K*, the length of the nearest neighbour's pattern *m* (also called the 'embedding dimension') and the weighting of final prices in a neighbour *α*. When *α* is greater than 1, a greater emphasis is given to similarity between the more recent observations. The nearest neighbours algorithm weights the nearest to furthest Neighbour sets according to the Euclidean distance from the actual time

series. These three parameters are very important for neighbours and very easy to apply computationally.

Guégan and Huck (2004) suggest that a good approximation for choosing the parameters *K* and *m* is dependent on the size of the information set. They choose m from the interval:

$$m = [R(\ln(T)), R(\ln(T)+2)] \quad [8]$$

where R is the rounding function rounding to the immediate lower figure and T the size of the dataset. They also suggested that K should be approximately twice the value of m. Thus for our dataset m lies between 8 and 10 and K lies between 16 and 20. Based on the above guidelines and Dunis and Nathani (2007) who applied Nearest Neighbours in financial series, we experiment in the in-sample dataset and we select the set of parameters that provide the highest trading performance in the in-sample period. These sets of parameters for each exchange rate under study are presented in table 3 below.

	m	K	a
EUR/USD	8	18	1.1
EUR/GBP	8	19	1
EUR/JPY	9	18	1.2

Table 3: Nearest Neighbours Parameters

4.2 Neural Networks

4.2.1. Benchmark Neural Networks Architectures

Neural networks exist in several forms in the literature. The most popular architecture is the Multi-Layer Perceptron (MLP). A standard MLP has at least three layers. The first layer is called the input layer (the number of its nodes corresponds to the number of explanatory variables). The last layer is called the output layer (the number of its nodes corresponds to the number of response variables). An intermediary layer of nodes, the hidden layer, separates the input from the output layer. Its number of nodes defines the amount of complexity the model is capable of fitting. In addition, the input and hidden layer contain an extra node called the bias node. This node has a fixed value of one and has the same function as the intercept in traditional regression models. Normally, each node of one layer has connections to all the other nodes of the next layer.

The training of the network (which is the adjustment of its weights in the way that the network maps the input value of the training data to the corresponding output value) starts with randomly

chosen weights and proceeds by applying a learning algorithm called backpropagation of errors[6] (Shapiro (2000)).The iteration length is optimised by maximising a fitness function in the test dataset.

In addition to the classical MLP network, we also apply a recurrent neural network. While a complete explanation of RNN models is beyond the scope of this paper, we present below a brief explanation of the RNN architecture. For an exact specification of recurrent networks, see Elman (1990). A simple recurrent network has an activation feedback which embodies short-term memory. In other words, the RNN architecture can provide more accurate outputs because the inputs are (potentially) taken from all previous values. The advantages of using recurrent networks over feedfoward networks (such as the MLPs) for modelling non-linear time series, has been well documented in the past (see amongst other Elman (1990) and Tenti (1996)). However as mentioned by Tenti (1996), "the main disadvantage of RNNs is that they require substantially more connections, and more memory in simulation than standard backpropagation networks" (p.569), thus resulting in a substantial increase in computational time. However, having said this, RNNs can yield better results in comparison with simple MLPs due to the additional memory inputs.

The third NN benchmark model considered in our study is the Psi Sigma. Psi Sigma networks can be considered as a class of feedfoward fully connected higher order neural networks. First introduced by Shin and Ghosh (1991), the Psi Sigma network utilizes product cells as the output units to indirectly incorporate the capabilities of higher-order networks while using a fewer number of weights and processing units. Their creation was motivated by the need to create a network combining the fast learning property of single layer networks with the powerful mapping capability of higher order neural networks while avoiding the combinatorial increase in the required number of weights. The order of the network in the context of Psi Sigma is represented by the number of hidden nodes. In a Psi Sigma network the weights from the hidden to the output layer are fixed to 1 and only the weights from the input to the hidden layer are adjusted, something that greatly reduces the training time. Moreover, the activation function of the nodes in the hidden layer is the summing function while the activation function of the output layer is a sigmoid. More details on the Psi Sigma architecture can be found in Shin and Ghosh (1991), Dunis *et. al.* (2010 and 2011) and Sermpinis *et. al.* (2012).

4.2.2. Proposed Fitness Function and NN optimization

[6] Backpropagation networks are the most common multi-layer networks and are the most commonly used type in financial time series forecasting (Kaastra and Boyd (1996)).

In our networks which are specially designed for financial purposes we apply a novel multi-objective fitness function. This fitness function focuses on achieving three goals at the same time. First of all, the annualized return in the training period should be maximized and secondly the Mean Square Error (MSE) of the networks output should be minimized. Finally, we are interested in finding the simplest neural network that achieves these goals and thus the number of hidden neuron should be minimized. Based on the above the fitness function for all our networks takes the form of the equation below[7]:

$$\text{Fitness} = \text{Annualized\_Return} - \text{MSE} - 10^{-2} * \text{number\_of\_hidden\_neurons} \quad [9]$$

To the best of our knowledge no similar approach has been applied yet in the relevant literature and equation [9] is an original contribution of this paper. The existing financial forecasting approaches guide their predictors using as fitness functions statistical measures such as the MSE (e.g. Fulcher *et. al.* (2006)), the RMSE (e.g. Panda and Narasimhan (2007)) or the correct directional movement prediction rate (e.g. Dunis *et. al.* (2010 and 2011)). However, statistical accuracy does not always imply financial profitability. The proposed fitness function aims to bring a balance between trading profitability (first factor of equation [9]) and statistical accuracy (second factor). Thus, the proposed fitness function clearly outperforms existing fitness functions as our main goal is to extract a profitable trading strategy and not only a prediction that presents low error measures.

After our networks are optimized, the predictive value of each model is evaluated by applying it to the validation dataset (out-of-sample dataset). Since the starting point for each network is a set of random weights, forecasts can differ between networks. In order to eliminate any variance between our NN forecasts, we use the average of a committee of 10 NNs which present the highest profit in the test sub-period. In our study we apply our NNs to the task of forecasting the daily return of the EUR/USD, EUR/GBP and EUR/JPY exchange rates. The characteristics of the NNs used in this paper are presented in Appendix A.1.

4.2.3 ARBF-PSO method

4.2.3.1 Radial Basis Function Neural Networks (RBFNN)

A radial basis function neural network (RBFNN) is a feedforward neural network where hidden units do not implement an activation function, but a radial basis function. An RBFNN approximates a desired function by superposition of nonorthogonal, radially symmetric functions. They have been proposed by Broomhead and Lowe (1988) as an approach to improve

[7] The number of hidden neurons is multiplied with 10^{-2} because the simplicity of the derived neural network is of secondary importance compared to the other two objectives (maximize the annualized return and minimizing the

accuracy of artificial neural networks while decreasing training time complexity. Their architecture is depicted in Figure 2.

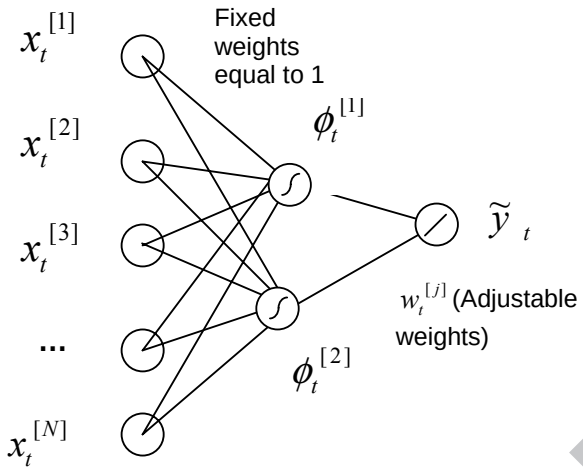

Fig. 2: A RBF Neural Network with N inputs and 2 hidden nodes

x_t $(n = 1,2,\hbar\ ,N+1)$ are the model inputs (including the input bias node)

$\tilde{y}_t$ is the RBFNN output

$w_t^{[j]}$ (j=1,2) are the adjustable weights

ϕ is the Gaussian function: $$\phi^{[i]}(x_t) = e^{-\frac{\|x_t - C_i\|^2}{2\sigma_i^2}} \quad [10]$$

where C_i is a vector indicating the centre of the Gaussian Function and σ_i is a value indicating its width. C_i, σ_i and the weights w_i are parameters which should be optimized through a learning phase in order to train the RBFNN.

$\oslash$ is the linear output function: $$F(\phi) = \sum_i \phi^{[i]} \quad [11]$$

The error function to be minimised is:

$$E(C, \sigma, w_t) = \frac{1}{T}\sum_{t=1}^{T}(y_t - \tilde{y}_t(w_t, C, \sigma))^2 \quad [12]$$

with y_t being the target value and T the number of iterations.

4.2.3.2 ARBF-PSO method

The hybrid methodology proposed in the present paper is an extension of the hybrid algorithm proposed by Ding *et. al.* (2005). In this algorithm the Particle Swarm Optimization (PSO) methodology was used to locate the parameters C_i of the RBFNN while in parallel locating the optimal number for the hidden layers of the network. This methodology is extended in our proposed algorithm in order to increase accuracy, make it appropriate for predicting financial time series and avoid the time consuming step of optimizing the parameters of PSO.

MSE).

The PSO algorithm, proposed by Kennedy and Eberhart (1995), is a population based heuristic search algorithm based on the simulation of the social behavior of birds within a flock. In PSO, individuals which are referred to as particles are placed initially randomly within the hyper dimensional search space. Changes to the position of particles within the search space are based on the social-psychological tendency of individuals to emulate the success of other individuals. The consequence of modeling this social behavior is that the search process is such that particles stochastically return towards previously successful regions in the search space.

The performance of an RBFNN highly depends on its structure and on the effective calculation of the RBF function's centers C_i and widths σ and the network's weights. If the centers of the RBF are properly estimated then their widths and the networks weights can be computed accurately with existing heuristic and analytical methodologies which described below in this paper. In this approach the PSO searches only for optimal values of the parameters C_i and for the optimal number of hidden neurons (the RBFNN structure) which should be used. Each particle i is initialized randomly to have m_i hidden neurons (within a predefined interval starting from the number of inputs until 100 which is the maximum hidden layer size that we applied) and is represented as shown in equation [13]:

$$C^i = \begin{bmatrix} c^i_{11} & c^i_{12} & \dots & c^i_{1d} \\ c^i_{21} & c^i_{22} & \dots & c^i_{2d} \\ . & . & \dots & . \\ . & . & \dots & . \\ . & . & \dots & . \\ c^i_{m^i 1} & c^i_{m^i 2} & \dots & c^i_{m^i d} \\ N & N & N & N \\ . & . & \dots & . \\ . & . & \dots & . \\ . & . & \dots & . \\ N & N & N & N \end{bmatrix} \qquad [13]$$

where N is a large number to point that it does not represent an RBF centre. Using this specific representation scheme, we allow the production of RBFNNs with fewer hidden nodes than the maximum allowed hidden layers size. Thus, the produced optimization algorithm becomes even more flexible and enables the algorithm to locate the optimal network structure for each problem. By optimizing equation [13] the identification of the RBFNN structure and the effective calculation of the RBF function centers can be accomplished in parallel.

In our PSO variation, initially we create a random population of particles, with candidate solutions represented as showed in equation [13], each one having an initially random velocity

matrix to move within the search space. It is this velocity matrix that drives the optimization process, and reflects both the experiential knowledge of the particle and socially exchanged information from the particles neighborhood. The form of the velocity matrix for every particle is described in the equation below:

$$V^i = \begin{bmatrix} v_{11}^i & v_{12}^i & \dots & v_{1d}^i \\ v_{21}^i & v_{22}^i & \dots & v_{2d}^i \\ . & . & \dots & . \\ . & . & \dots & . \\ . & . & \dots & . \\ v_{m^i 1}^i & v_{m^i 2}^i & \dots & v_{m^i d}^i \\ N & N & \dots & N \\ . & . & \dots & . \\ . & . & \dots & . \\ . & . & \dots & . \\ N & N & \dots & N \end{bmatrix} \quad [14]$$

From the centers of its particle described in equation [13] using the Moody-Darken (1989) approach we compute the RBF widths using the following equation:

$$\sigma_j^i = \left\| c_j^i - c_k^i \right\| \quad [15]$$

where c_k^i is the nearest neighbor of the centers c_j^i. For the estimation of the nearest neighbors we apply the Euclidean distance which is computed for every pair of centers.

At this point of the algorithm the centers and the widths of the RBFNN have been computed. The computation of its optimal weights w^i is accomplished by solving the equation:

$$w^i = (H_i^T \cdot H_i)^{-1} \cdot H_i^T \cdot Y \quad [16]$$

where

$$H_i = \begin{bmatrix} \phi_1^i(x_1) & \phi_2^i(x_1) & \dots & \phi_{m^i}^i(x_1) \\ \phi_1^i(x_2) & \phi_2^i(x_2) & \dots & \phi_{m^i}^i(x_2) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \phi_1^i(x_{n_1}) & \phi_2^i(x_{n_1}) & \dots & \phi_{m^i}^i(x_{n_1}) \end{bmatrix}$$

where n_1 is the number of training samples.

The computation of $(H_i^T \cdot H_i)^{-1}$ is computationally hard when the rows of H_i are highly dependent. In order to solve this problem we filtered the in-sample dataset and when the mean absolute distance of two training samples is less than 10^{-3} from the mean values of their input values then randomly we do not use one of them in our final training set. By this way, the algorithm becomes faster while maintaining its high accuracy. This analytical approach for the estimation of the RBFNN weights is superior in comparison with the application of meta-heuristic methods (PSO, Genetic Algorithms, Swarm Fish algorithm) that have been already

presented in the literature because it eradicates the risk of getting trapped into local optima and the final solution is assured to be optimal for a subset of the training set.

Next, our novel multi-objective fitness function [9] was used for evaluating the performance of its particle.

Iteratively, the position of each particle is changed by adding in it its velocity vector and the velocity matrix for each particle is changed using the equation below:

$$V^{i+1} = w * V^i + c1 * r1 * (C^i_{pbest} - C^i) + c2 * r2 * (C^i_{gbest} - C^i) \quad [17]$$

where w is a positive-valued parameter showing the ability of each particle to maintain its own velocity, C^i_{pbest} is the best solution found by this specific particle so far, C^i_{gbest} is the best solution found by every particle so far, c1 and c2 are used to balance the impact of the best solution found so far for a specific particle and the best solution found by every particle so far in the velocity of a particle. Finally, r1, r2 are random values in the range of [0,1] sampled from a uniform distribution.

Ideally, PSO should explore the search space thoroughly in the first iterations and so the values for the variables w and c1 should be kept high. For the final iterations the swarm should converge to an optimal solution and the area around the best solution should be explored thoroughly. Thus, c2 should be valued with a relatively high value and w, c1 with low values. In order to achieve the described behavior for our PSO implementation and to avoid getting trapped in local optima when being in an early stage of the algorithm's execution we developed a PSO implementation using adaptive values for the parameters w, c1 and c2. Equations [18], [19] and [20] mathematically describe how the values for these parameters are changed through PSO's iterations helping us to endow the desired behavior in our methodology.

$$w(t) = (0.4/n^2) * (t-n)^2 + 0.4 \quad [18]$$

$$c1(t) = -2 * t/n + 2.5 \quad [19]$$

$$c2(t) = 2 * t/n + 0.5 \quad [20]$$

where t is the present iteration and n is the total number of iterations.

In order to enhance the optimization of the RBFNN structure we applied two more operators in our hybrid method in addition to the classical ones of the PSO. The first operator is used to add a hidden neuron in every particle with a probability equal to 0.1. The second operator reduces the hidden neurons by one with a probability equal to 0.1. The probabilities which were applied for increasing and decreasing the hidden neurons, were set as equal to reassure that the

algorithm is not further biased towards larger or smaller architectures. Furthermore, a high probability equal to 0.8 was assigned to the fact of not changing the network's architecture in order to enforce the algorithm to explore thoroughly the potential of existing architectures in the population before investigating different ones.

For the initial population of particles we use a small value of 30 particles and the number of iterations used was 100 combined with a convergence criterion. Using this termination criterion the algorithm stops when the population of the particles is deemed as converged. The population of the particles is deemed as converged when the average fitness across the current population is less than 5% away from the best fitness of the current population. Specifically, when the average fitness across the current population is less than 5% away from the best fitness of the population, the diversity of the population is very low and evolving it for more generations is unlikely to produce different and better individuals than the existing ones or the ones already examined by the algorithm in previous generations.

The adaptive methodology described above is capable of protecting our results from the data-snooping effect. Data snooping occurs when a given set of data is used more than once for purposes of inference or model selection. When such data reuse occurs, there is always the possibility that any satisfactory results obtained may simply be due to chance rather than to any merit inherent in the method yielding the results (White (2000)). The problem can be particularly serious when using neural network models which are basically atheoretical. However in our case, the simultaneous optimization of the RBF's structure and parameters in a single optimization procedure should prevent the data snooping effect. In order to verify that our proposed model is free from the data snooping bias in our study, we apply the White's (2000) Reality Check test. The null hypothesis of the test is that the model selected in a specification search has no predictive superiority over a given benchmark model. Following the relevant literature we use as benchmarks a martingale model (Yang *et. al.* (2010)) and a strategy of absence from the market or else of 0% return (Qi and Wu (2006)). The "nominal *p*-value" is calculated by applying the Reality Check methodology only to the best ARBF-PSO network in terms of MSE while the "White's *p*-value" is computed by applying the Reality Check methodology to all the 10 ARBF-PSO networks used in this study. In table 4 below we present the White's (2000) Reality Check p-values[8] for our proposed model.

Benchmark Models		EUR/USD	EUR/GBP	EUR/JPY
Martingale model	Nominal *p*-value	0.000	0.000	0.000
	White's *p*-value	0.001	0.003	0.001
Absence from the market	Nominal *p*-value	0.000	0.000	0.000
	White's *p*-value	0.003	0.003	0.002

Table 4: White Reality Check p-values for the ARBF-PSO method

We note from the table above that the null hypothesis can be rejected at the 1% level in all cases and our ARBF-PSO forecasts are free from the data snooping bias.

In summary the novelty of our algorithm lies in the following points. First of all, the training samples in our methodology are initially filtered before we insert them to our network, and when the mean absolute distance of two training samples is less than 10^{-3} from the mean values of their input values then randomly we do not use one of them in our final training set. By this way we decrease the computational complexity of the method while maintaining its accuracy. Furthermore, in comparison with other optimization techniques which have already been applied for training RBFNNs for forecasting financial time series, our algorithm optimizes both the networks structure and its parameters. Specifically, the proposed algorithm optimizes the number of hidden nodes and in parallel the optimal centers of the RBF functions. The widths of the RBF functions and the RFBNNs weights are estimated in an analytical manner without the risk of getting trapped into local optima, decreasing the algorithms complexity. Moreover, all PSO parameters in our approach are adaptive using equations [18], [19] and [20] making our method appropriate for usage by non experts while at the same time avoiding the risky and time consuming trial and error approach of optimizing the parameters of a NN. It is also worth nothing that the simultaneous optimization of the RBF's structure and parameters in a single optimization procedure prevents the data snooping effect which is present in existing methodologies for neural networks structure and parameters optimization and leads to overestimating their performance. Finally, in order to adapt our methodology to our specific task of financial time series forecasting we use the multi-objective fitness function [9] to select more profitable predictors for our ARBF-PSO and the other three NNs algorithms retained.

A simple execution of the proposed methodology for modeling and trading a financial index, using 10 years historical data, does not require more than one hour using a modern personal

[8] Following the approach of Qi and Wu (2006) who also applied White's (2000) Reality Check test in FX data, we performed our test with 500 bootstrap replications and a bootstrap smoothing parameter equal to 0.75.

computer. The adaptive nature of the proposed algorithm makes a simple execution of the algorithm sufficient because the algorithm's parameters are automatically tuned. This is less than the time needed for a MLP, a RNN and a PSN network but admittedly more than the time needed for the optimization of an ARMA model (a few minutes).

5. EMPIRICAL RESULTS

5.1 STATISTICAL PERFORMANCE

5.1.1 Statistical Accuracy Measures

The in-sample statistical performance of our models is presented in table 5 below for the EUR/USD, the EUR/GBP and the EUR/JPY exchange rates. For the four error statistics retained (Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and Theil-U) the lower the output, the better the forecasting accuracy of the model concerned. The Pesaran-Timmermann (PT) test (1992) examines whether the directional movements of the real and forecast values are in step with one another, or to put it another way, it checks how well rises and falls in the forecast value follow actual rises and falls. The null hypothesis is that the model under study has no power on forecasting the relevant exchange rate.

		NAIVE	ARMA	MACD	k-NN	MLP	RNN	PSI	ARBF-PSO
EUR/USD	MAE	**0.0106**	**0.0054**	**0.0053**	**0.0054**	**0.0050**	**0.0049**	**0.0051**	**0.0047**
	MAPE	**789.15%**	**184.66%**	**179.37%**	**165.53%**	**131.22%**	**115.32%**	**116.78%**	**112.32%**
	RMSE	**0.0249**	**0.0073**	**0.0171**	**0.0074**	**0.0062**	**0.0060**	**0.0061**	**0.0054**
	Theil-U	**0.7943**	**0.7209**	**0.7832**	**0.6875**	**0.5821**	**0.5424**	**0.5343**	**0.4824**
	PT-statistic	**0.03**	**2.87***	**0.82**	**3.64***	**4.43***	**8.76***	**10.12***	**14.23***
EUR/GBP	MAE	**0.0076**	**0.0042**	**0.0044**	**0.0039**	**0.0034**	**0.0031**	**0.0030**	**0.0028**
	MAPE	**229.16%**	**142.92%**	**147.26%**	**121.11%**	**102.92%**	**98.92%**	**99.22%**	**93.92%**
	RMSE	**0.0237**	**0.0027**	**0.0035**	**0.0020**	**0.0017**	**0.0019**	**0.0017**	**0.0014**
	Theil-U	**0.8235**	**0.3615**	**0.3999**	**0.2682**	**0.2515**	**0.2512**	**0.2342**	**0.2042**
	PT-statistic	**0.76**	**3.12***	**0.91**	**3.33***	**5.88***	**9.92***	**11.23***	**14.67***
EUR/JPY	MAE	**0.0123**	**0.0087**	**0.0075**	**0.0064**	**0.0054**	**0.0050**	**0.0052**	**0.048**
	MAPE	**842.64%**	**197.53%**	**284.43%**	**168.32%**	**146.39%**	**125.76%**	**132.43%**	**123.32%**
	RMSE	**0.0267**	**0.0092**	**0.0134**	**0.0083**	**0.0063**	**0.0059**	**0.0061**	**0.0056**
	Theil-U	**0.7994**	**0.6823**	**0.7226**	**0.6237**	**0.5772**	**0.5375**	**0.5471**	**0.4983**
	PT-statistic	**0.02**	**2.68***	**1.14**	**2.97***	**4.32***	**7.46***	**9.32***	**13.11***

Table 5: In-sample statistical performance

Note: * denote rejection of the null hypothesis of no forecasting power at the 1% confidence interval

We note that ARBF-PSO presents slightly better in-sample statistical performance than its benchmarks. The RNN and PSI forecasts are similar in terms of statistical accuracy with the MLP presenting the third more accurate forecasts. The realizations of the statistical accuracy measures are considerably higher for our more traditional benchmarks, something that was expected based on their simplistic nature. On the other hand, the PT-statistics indicate that only our Naïve and MACD strategies are poor forecasts for the three exchange rates under study. In table 6 below we present the statistical performance of our models for the out-of-sample period.

		NAIVE	ARMA	MACD	k-NN	MLP	RNN	PSI	ARBF-PSO
EUR/USD	MAE	**0.0074**	**0.0093**	**0.0056**	**0.0057**	**0.0058**	**0.0055**	**0.0053**	**0.0052**
	MAPE	**484.96%**	**196.07%**	**164.98%**	**177.43%**	**186.07%**	**166.72%**	**163.41%**	**145.72%**
	RMSE	**0.0093**	**0.0067**	**0.0070**	**0.0069**	**0.0065**	**0.0061**	**0.0058**	**0.0057**
	Theil-U	**0.7012**	**0.7433**	**0.7768**	**0.7631**	**0.7276**	**0.7106**	**0.6506**	**0.6306**
	PT-statistic	**0.23**	**0.02**	**0.52**	**2.42***	**3.78***	**6.36***	**7.91***	**8.96***
EUR/GBP	MAE	**0.0056**	**0.0054**	**0.0057**	**0.0051**	**0.0050**	**0.0047**	**0.0045**	**0.0041**
	MAPE	**487.51%**	**225.92%**	**240.27%**	**192.74%**	**189.32%**	**157.92%**	**159.98%**	**144.92%**
	RMSE	**0.0099**	**0.0063**	**0.0074**	**0.0072**	**0.0055**	**0.0050**	**0.0048**	**0.0048**
	Theil-U	**0.9257**	**0.7228**	**0.7357**	**0.7032**	**0.5103**	**0.4367**	**0.4243**	**0.4040**
	PT-statistic	**0.44**	**2.14**	**0.03**	**2.86***	**2.99***	**6.12***	**7.04***	**9.22***
EUR/JPY	MAE	**0.0155**	**0.0103**	**0.0098**	**0.0088**	**0.0068**	**0.0050**	**0.0052**	**0.051**
	MAPE	**573.12%**	**223.84%**	**253.17%**	**201.34%**	**178.98%**	**151.48%**	**157.90%**	**149.41%**
	RMSE	**0.0298**	**0.0125**	**0.0189**	**0.0105**	**0.0077**	**0.0065**	**0.0073**	**0.0063**
	Theil-U	**0.8012**	**0.7717**	**0.7632**	**0.7213**	**0.6831**	**0.6451**	**0.6930**	**0.6003**
	PT-statistic	**0.01**	**0.12**	**0.04**	**2.01**	**3.58***	**5.02***	**4.74***	**7.32***

Table 6: Out-of-sample statistical performance

Note: * denote rejection of the null hypothesis of no forecasting power at the 1% confidence interval

From the results above, we note that ARBF-PSO retains its forecasting superiority in the out-of-sample sub-period for the statistical measures applied. Once more, the RNN and the PSI present the second best performance with MLP having the third best forecasts in terms of statistical accuracy. The PT statistics indicate that all our non linear models are good forecasts of the exchange rates under study except the k-NN model for the EUR/JPY series.

5.1.2 Diebold-Mariano Test

In order to test if our best model in terms of statistical measures produces forecasts that are statistically significant and superior to its counterparts, we apply the Diebold-Mariano (1995) statistic for predictive accuracy for both MSE and MAE loss functions (for more details on the test see Diebold and Mariano (1995)). The results of the Diebold-Mariano statistic, comparing the ARBF-PSO network with its benchmarks for the three exchange rates under study are summarized in the table 7 below.

		NAIVE	ARMA	MACD	k-NN	MLP	RNN	PSI
EUR/USD	MSE	**-8.456**	**-4.827**	**-5.542**	**-5.049**	**-4.214**	**-3.194**	**-3.224**
	MAE	**-9.143**	**-5.431**	**-6.281**	**-5.905**	**-5.104**	**-4.871**	**-4.523**
EUR/GBP	MSE	**-8.812**	**-5.991**	**-7.840**	**-4.576**	**-4.402**	**-3.002**	**-3.344**
	MAE	**-9.924**	**-6.214**	**-8.515**	**-5.385**	**-4.919**	**-3.461**	**-3.951**
EUR/JPY	MSE	**-10.572**	**-6.718**	**-7.294**	**-5.751**	**-5.018**	**-3.851**	**-3.919**
	MAE	**-10.917**	**-6.210**	**-7.821**	**-6.003**	**-5.891**	**-4.190**	**-4.387**

Table 7: Diebold-Mariano Statistics

From the above table we note that the null hypothesis of equal predictive accuracy is rejected for all comparisons and for both loss functions at a 1% confidence interval, since all the test statistics are above the critical value of 2.33. Moreover, the statistical superiority of the ARBF-

PSO forecasts is confirmed as for both loss functions the realizations of the Diebold-Mariano (1995) statistic are negative[9].

5.2 TRADING PERFOMANCE

5.2.1 Trading Strategy and Transaction Costs

In the previous section we evaluate our forecasts through a series of statistical accuracy measures and tests. However, statistical accuracy is not always synonymous of financial profitability. In financial applications, the practitioner's utmost interest is in producing models that can be translated to profitable trades. It is therefore crucial to further examine our proposed model and evaluate its utility through a trading strategy. The trading strategy applied is to go or stay 'long' when the forecast return is above zero and go or stay 'short' when the forecast return is below zero. The 'long' and 'short' EUR/USD, EUR/GBP or EUR/JPY position is defined as buying and selling Euros at the current price respectively. When the forecast return is 0 we keep our position.

Transaction costs for a tradable amount, say USD 5-10 million, are about 1 pip per trade (one way) between market makers. But since we consider the EUR/USD, EUR/GBP and EUR/JPY time series as a series of middle rates, the transaction costs is one spread per round trip. For our dataset a cost of 1 pip is equivalent to an average cost of 0.007%, 0.012% and 0.008% per position for the EUR/USD, the EUR/GBP and the EUR/JPY respectively. Table 8 below presents the trading performance of our models in the in-sample sub-period.

		NAIVE	ARMA	MACD	k-NN	MLP	RNN	PSI	ARBF-PSO
EUR/USD	*Information Ratio (excluding costs)*	0.01	0.89	0.29	1.73	1.91	2.46	3.15	3.74
	Annualised Volatility (excluding costs)	10.56%	10.60%	10.59%	10.51%	10.58%	10.52%	10.49%	10.41%
	Annualised Return (excluding costs)	0.11%	9.42%	3.07%	18.14%	20.23%	25.87%	33.02%	38.96%
	Maximum Drawdown (excluding costs)	-33.77%	-13.76%	-29.67%	-31.74%	-29.40%	-21.38%	-26.76%	-25.57%
	Positions Taken (annualised)	129	128	39	95	103	101	93	96
	Transaction Costs (annualised)	0.90%	0.90%	0.27%	0.67%	0.72%	0.71%	0.65%	0.67%
	Annualised Return (including costs)	**-0.79%**	**8.52%**	**2.80%**	**17.47%**	**19.51%**	**25.16%**	**32.37%**	**38.29%**
	Information Ratio (including costs)	**-0.08**	**0.80**	**0.26**	**1.66**	**1.84**	**2.39**	**3.09**	**3.68**
EUR/GBP	*Information Ratio (excluding costs)*	0.53	1.46	0.52	2.05	2.60	3.15	3.35	4.71
	Annualised Volatility (excluding costs)	7.94%	7.93%	7.96%	7.96%	7.94%	7.81%	7.79%	7.62%
	Annualised Return (excluding costs)	4.19%	11.57%	4.16%	16.32%	20.61%	24.60%	26.12%	36.35%
	Maximum Drawdown (excluding costs)	-25.43%	-18.64%	-21.63%	-20.54%	-19.52%	-14.74%	-18.48%	-13.35%
	Positions Taken (annualised)	128	140	81	103	145	69	64	61
	Transaction Costs (annualised)	1.57%	1.68%	0.97%	1.24%	1.74%	0.83%	0.77%	0.73%
	Annualised Return (including costs)	**2.62%**	**9.89%**	**3.19%**	**15.08%**	**18.87%**	**23.77%**	**25.35%**	**35.62%**
	Information Ratio (including costs)	**0.33**	**1.17**	**0.40**	**1.89**	**2.37**	**3.04**	**3.25**	**4.67**
EUR/JPY	*Information Ratio (excluding costs)*	0.62	0.89	0.42	1.37	1.48	2.15	2.13	2.45
	Annualised Volatility (excluding costs)	12.60%	12.61%	12.63%	12.60%	12.58%	12.52%	12.51%	12.57%
	Annualised Return (excluding costs)	7.75%	11.24%	5.28%	17.22%	18.61%	26.92%	26.61%	30.79%

[9] We apply the Diebold-Mariano test to couples of forecasts (ARBF-PSO vs. another forecasting model). A negative realization of the Diebold-Mariano test statistic indicates that the first forecast (ARBF-PSO) is more accurate than the second forecast. The lower the negative value, the more accurate are the ARBF-PSO forecasts.

	Maximum Drawdown (excluding costs)	-39.96%	-20.43%	-28.71%	-22.37%	-18.34%	-36.09%	-24.41%	-20.41%
	Positions Taken (annualised)	120	166	155	142	71	65	88	95
	Transaction Costs (annualised)	0.96%	1.33%	1.24%	1.14%	0.57%	0.52%	0.70%	0.76%
	Annualised Return (including costs)	**6.79%**	**9.91%**	**4.04%**	**16.08%**	**18.04%**	**26.40%**	**25.91%**	**30.03%**
	Information Ratio (including costs)	**0.54**	**0.79**	**0.32**	**1.28**	**1.43**	**2.11**	**2.07**	**2.39**

Table 8: In-sample trading performance

Clearly our ARBF-PSO network demonstrates a superior trading performance in terms of annualised return and information ratio for all exchange rates in the in-sample period. The PSI and the RNN produce the second and the third most profitable trades. In comparison with the statistical performance of our models for the same period, we note that although PSI and RNN present very close statistical forecasts, this is not always the case in trading terms. In general all our models are producing positive annualised returns except the naïve strategy for the EUR/USD something that was expected as all our models (except the naïve) were optimized during this period. In table 9 we present our results for the out-of-sample period.

		NAIVE	**ARMA**	**MACD**	**k-NN**	**MLP**	**RNN**	**PSI**	**ARBF-PSO**
EUR/USD	*Information Ratio (excluding costs)*	0.07	-1.19	0.26	0.84	1.26	1.65	2.01	2.49
	Annualised Volatility (excluding costs)	10.55%	10.52%	10.55%	10.55%	10.52%	10.49%	10.47%	10.42%
	Annualised Return (excluding costs)	0.73%	-12.55%	2.72%	8.88%	13.24%	17.27%	21.06%	25.93%
	Maximum Drawdown (excluding costs)	-19.87%	-32.64%	-13.27%	-14.61%	-12.83%	-17.16%	-9.14%	-8.10%
	Positions Taken (annualised)	127	108	40	132	126	124	123	119
	Transaction Costs (annualised)	0.89%	0.76%	0.28%	0.92%	0.88%	0.87%	0.86%	0.83%
	Annualised Return (including costs)	**-0.16%**	**-13.31%**	**2.44%**	**7.96%**	**12.36%**	**16.40%**	**20.20%**	**25.10%**
	Information Ratio (including costs)	**-0.02**	**-1.26**	**0.23**	**0.75**	**1.17**	**1.56**	**1.93**	**2.41**
EUR/GBP	*Information Ratio (excluding costs)*	0.51	1.00	0.09	0.91	1.32	1.97	2.05	3.11
	Annualised Volatility (excluding costs)	9.37%	9.37%	9.36%	9.38%	9.34%	9.33%	9.32%	9.26%
	Annualised Return (excluding costs)	4.80%	9.38%	0.85%	8.57%	12.31%	18.41%	19.14%	28.83%
	Maximum Drawdown (excluding costs)	-10.19%	-16.36%	-12.87%	-16.14%	-14.49%	-11.88%	-9.67%	-7.49%
	Positions Taken (annualised)	132	124	83	114	122	143	151	148
	Transaction Costs (annualised)	1.58%	1.49%	0.96%	1.37%	1.46%	1.71%	1.81%	1.77%
	Annualised Return (including costs)	**3.22%**	**7.89%**	**-0.11%**	**7.20%**	**10.85%**	**16.70%**	**17.33%**	**27.06%**
	Information Ratio (including costs)	**0.34**	**0.84**	**-0.01**	**0.77**	**1.16**	**1.79**	**1.86**	**2.92**
EUR/JPY	*Information Ratio (excluding costs)*	-0.81	0.10	-0.09	0.46	0.79	1.46	1.34	1.80
	Annualised Volatility (excluding costs)	13.27%	13.32%	13.32%	13.29%	13.29%	13.28%	13.25%	13.31%
	Annualised Return (excluding costs)	-10.71%	1.30%	-1.19%	6.11%	10.53%	19.36%	17.71%	23.96%
	Maximum Drawdown (excluding costs)	-44.30%	-17.43%	-31.61%	-19.81%	-19.26%	-15.14%	-13.76%	-8.51%
	Positions Taken (annualised)	134	171	43	147	146	140	152	136
	Transaction Costs (annualised)	1.09%	1.37%	-0.34%	1.18%	1.17%	1.12%	1.22%	1.09%
	Annualised Return (including costs)	**-11.80%**	**-0.07%**	**-1.53%**	**4.93%**	**9.36%**	**18.24%**	**16.49%**	**22.87%**
	Information Ratio (including costs)	**-1.10**	**-0.01**	**-0.12**	**0.37**	**0.70**	**1.37**	**1.24**	**1.72**

Table 9: Out-of-sample trading performance

We observe from the bold values of the above table that the ARBF-PSO confirms its trading superiority in the out-of-sample period. This is consistent with the superior statistical performance of our proposed methodology compared to its other NN and statistical benchmarks. A superiority that can be attributed to the proposed network algorithm that seems to excel in recognising the pattern of the two series under study compared to more traditional models. It is also worth noting that ARBF-PSO forecasts present a remarkably low maximum drawdown in

the out-of-sample period for all the three exchange rates under study. Thus the maximum potential losses of an investor are almost 3 or 4 times lower than the annualised profit in the out-of-sample period. Concerning our benchmarks models, we note that PSI and RNN present the second and the third best performance in terms of trading efficiency while our k-NN model has the fourth best performance with positive returns after transaction costs for all three exchange rates. The results of the more traditional models are mixed with the MACD for the EUR/USD and the naïve and ARMA for the EUR/GBP presenting slightly positive annualised returns after transaction costs.

Concerning our proposed fitness function of equation [9], the results from the statistical and trading evaluation of our NNs forecasts seem promising. Firstly we note that all our NNs present significant profits after transaction costs for all series under study in the out-of-sample period. Moreover, we did not note any large inconsistencies in our NNs statistical and trading performance for both exchange rates and sub-periods. Large inconsistencies could indicate that the training of our NNs is biased to either statistical accuracy or trading efficiency, something that could possibly lead to profitable in-sample forecasts but disastrous out-of-sample results. In the next section, we introduce a trading strategy to further increase the trading performance of our models.

5.3 LEVERAGE TO EXPLOIT HIGH INFORMATION RATIOS

In order to further improve the trading performance of our models we introduce a leverage based on GJR (1,1) one day ahead volatility forecasts[10].The intuition of the strategy is to avoid trading when volatility is very high while at the same time exploiting days when the volatility is relatively low. As mentioned by Bertolini (2010), there are few papers on market-timing techniques for foreign exchange, with the notable exception of Dunis and Miao (2005). The opposition between market-timing techniques and time-varying leverage is only apparent as time-varying leverage can also be easily achieved by scaling position sizes inversely to recent risk measures behaviour. One of the primary restrictions of GARCH models is their symmetric response to positive and negative shocks. However, it has been argued that a negative shock to

[10] We also explored a GARCH (1,1) model in forecasting volatility. Its statistical accuracy in the test sub-period in terms of the MAE, MAPE, RMSE and the Theil-U statistics is worse compared with GJR. Moreover, when we measure the utility of GARCH in terms of trading efficiency for our models within the context of our strategy in the test sub-period, our results in terms of annualised returns are slightly better with GJR for our models for all three exchange rates. However, we note that the use of the GJR model is quite puzzling for the series under study as an exchange rate is a *relative* price (contrary to, say, a stock market price). So for example a 'negative shock to the EUR/USD' is in reality a negative shock to the EUR and at the same time a positive shock for the USD, and there is therefore no reason *a priori* why its impact should be larger than "a positive shock of the same magnitude" to the EUR/USD (i.e. a positive shock to the EUR and at the same time a negative shock for the USD), unless one makes

financial time series is likely to cause volatility to rise by more than a positive shock of the same magnitude (Bekaert and Wu (2000) and Brooks (2003)). A popular asymmetric formulation of GARCH which has an additional term to account for possible asymmetries is the GJR model, where the conditional variance is given by:

$$\sigma_t^2 = w + au_{t-1}^2 + \beta\sigma_{t-1}^2 + \gamma u_{t-1}^2 I_{t-1} \qquad [21]$$

where σ_t^2 and u_t is the conditional variance and the error respectively at time t and $I_{t-1} = 1$ if u_{t-1} ħ 0 or 0 otherwise.

So we firstly forecast the one day ahead exchange rate volatility with a GJR (1,1) model in the test and validation sub-periods. Then, following Dunis and Miao (2005) we split these two periods into six sub-periods, ranging from periods with extremely low volatility to periods experiencing extremely high volatility. Periods with different volatility levels are classified in the following way: first the average (μ) difference between the actual volatility in day t and the forecast for day t+1 and its 'volatility' (measured in terms of standard deviation (ζ) are calculated; those periods where the difference is between μ plus one ζ are classified as 'Lower High Vol. Periods'. Similarly, 'Medium High Vol.' (between $\mu + \zeta$ and $\mu + 2\ \zeta$) and 'Extremely High Vol.' (above $\mu + 2\ \zeta$) periods can be defined. Periods with low volatility are also defined following the same 1 ζ and 2 ζ approach, but with a minus sign. For each sub-period a leverage is assigned starting with 0 for periods of extremely high volatility to a leverage of 2.5 for periods of extremely low volatility. Table 10 below presents the sub-periods and their relevant leverage factor.

	Extremely Low Vol.	Medium Low Vol.	Lower Low Vol.	Lower High Vol.	Medium High Vol.	Extremely High Vol.
Leverage	2.5	2	1.5	1	0.5	0

Table 10: Sub-periods and leverage factor

The parameters of our strategy (μ and ζ) are updated every three months by rolling forward the estimation period. So for example, for the first three months of our validation period, μ and ζ are computed based on the twenty four months of the test sub-period. For the following three months, the two parameters are computed based on the last fifteen months of our test sub-period and the first three of the validation sub-period. The cost of leverage (interest payments for the additional capital) is calculated at 1.75% p.a. (that is 0.0069% per trading day[11]). This leverage

the assumption that financial markets have a negative bias against the EUR (something that might explain the superior perfomance of GJR than GARCH in our study).

[11] The interest costs are calculated by considering a 1.75% interest rate p.a. (the Euribor rate at the time of calculation) divided by 252 trading days. In reality, leverage costs also apply during non-trading days so that we should calculate the interest costs using 360 days per year. But for the sake of simplicity, we use the approximation

is applied to the models that have achieved an in-sample information ratio of at least 2 and, as such, would have been candidates for leveraging out-of-sample. Our final results are presented in table 11.

		NAIVE	ARMA	MACD	k-NN	MLP	RNN	PSI	ARBF-PSO
EUR/USD	*Information Ratio (excluding costs)*	0.07	-1.19	0.26	0.84	1.26	1.48	1.86	2.03
	Annualised Volatility (excluding costs)	10.55%	10.52%	10.55%	10.55%	10.52%	17.13%	17.21%	17.21%
	Annualised Return (excluding costs)	0.73%	-12.55%	2.72%	8.88%	13.24%	25.31%	32.04%	35.02%
	Maximum Drawdown (excluding costs)	-19.87%	-32.64%	-13.27%	-14.61%	-12.83%	-12.77%	-6.69%	-6.01%
	Positions Taken (annualised)	127	108	40	132	126	114	113	109
	*Average Leverage Factor (ex post)**	n.a	n.a	n.a	n.a	n.a	1.44	1.50	1.33
	Transaction + Leverage Costs (annualised)	0.89%	0.76%	0.28%	0.92%	0.88%	1.71%	1.70%	1.65%
	Annualised Return (including costs)	**-0.16%**	**-13.31%**	**2.44%**	**7.96%**	**12.36%**	**23.60%**	**30.34%**	**33.37%**
	Information Ratio (including costs)	**-0.02**	**-1.26**	**0.23**	**0.75**	**1.17**	**1.38**	**1.76**	**1.94**
EUR/GBP	*Information Ratio (excluding costs)*	0.51	1.00	0.09	0.91	1.32	1.87	2.14	2.81
	Annualised Volatility (excluding costs)	9.37%	9.37%	9.36%	9.38%	9.34%	11.52%	11.55%	11.53%
	Annualised Return (excluding costs)	4.80%	9.38%	0.85%	8.57%	12.31%	21.56%	24.68%	32.42%
	Maximum Drawdown (excluding costs)	-10.19%	-16.36%	-12.87%	-16.14%	-14.49%	-9.01%	-7.72%	-5.12%
	Positions Taken (annualised)	132	124	83	114	122	100	108	105
	*Average Leverage Factor (ex post)**	n.a	n.a	n.a	n.a	n.a	1.18	1.31	1.13
	Transaction + Leverage Costs (annualised)	1.58%	1.49%	0.96%	1.37%	1.46%	1.85%	1.99%	1.91%
	Annualised Return (including costs)	**3.22%**	**7.89%**	**-0.11%**	**7.20%**	**10.85%**	**19.71%**	**22.69%**	**30.51%**
	Information Ratio (including costs)	**0.34**	**0.84**	**-0.01**	**0.77**	**1.16**	**1.71**	**1.96**	**2.65**
EUR/JPY	*Information Ratio (excluding costs)*	-0.81	0.10	-0.09	0.46	0.79	1.31	1.18	1.55
	Annualised Volatility (excluding costs)	13.27%	13.32%	13.32%	13.29%	13.29%	18.12%	18.09%	18.10%
	Annualised Return (excluding costs)	-10.71%	1.30%	-1.19%	6.11%	10.53%	23.81%	21.38%	28.02%
	Maximum Drawdown (excluding costs)	-44.30%	-17.43%	-31.61%	-19.81%	-19.26%	-12.81%	-11.01%	-6.47%
	Positions Taken (annualised)	134	171	43	147	146	122	134	118
	*Average Leverage Factor (ex post)**	n.a	n.a	n.a	n.a	n.a	1.23	1.20	1.17
	Transaction + Leverage Costs (annualised)	1.09%	1.37%	-0.34%	1.18%	1.17%	1.43%	1.58%	1.36%
	Annualised Return (including costs)	**-11.80%**	**-0.07%**	**-1.53%**	**4.93%**	**9.36%**	**22.38%**	**19.80%**	**26.66%**
	Information Ratio (including costs)	**-1.10**	**-0.01**	**-0.12**	**0.37**	**0.70**	**1.24**	**1.09**	**1.47**

***** The average leverage factor *ex post* is computed as the ratio of the annualised returns after costs of tables 13 and 11 for those models which the leverage was applied

Table 11: Out-of-sample trading performance –final results[11]

From the table above we note that our trading strategy is successful for all models applied. On average there is an increase in annualised returns after transaction costs of 8.53%, 3.94% and 3.75% for the EUR/USD, the EUR/GBP and the EUR/JPY exchange rate respectively when applying the leveraged trading strategy. Most importantly, we note that the application of our strategy leads to a substantial reduction in maximum drawdowns, the essence of risk for an investor in financial markets. In terms of model individual performance, the superiority of the ARBF-PSO is confirmed in terms of annualised return, information ratio and maximum drawdown. Our other two NNs models (the RNN and the PSI) which were candidates for a leverage, similarly show a satisfactory performance in terms of their profitability with PSI

of 252 trading days to spread the leverage costs of non-trading days equally over the trading days. This approximation prevents us from keeping track of how many non-trading days we hold a position.

[11] Not taken into account the interest that could be earned during times where the capital is not traded (non-trading days) or not fully invested and could therefore be invested.

presenting better performance for the EUR/USD and EUR/GBP and the RNN for the EUR/JPY exchange rate.

6. CONCLUDING REMARKS

In this paper, we introduce a hybrid Neural Network architecture of Particle Swarm Optimization and Adaptive Radial Basis Function (ARBF-PSO), a time-varying leverage trading strategy based on Glosten, Jagannathan and Runkle (1993) volatility forecasts and a neural network fitness function for financial forecasting purposes. We apply the proposed architecture to the task of forecasting the one day ahead return of the EUR/USD, the EUR/GBP and the EUR/JPY ECB daily fixings and benchmark its results with three different NNs, a Nearest Neighbors algorithm (k-NN) and three statistical/technical forecasting models. In terms of results, the ARBF-PSO outperforms all its benchmarks in terms of statistical accuracy and trading efficiency for both in- and out-of-sample periods. In terms of our proposed time-varying leverage trading strategy, we note that in the models applied there was a substantial increase in the trading performance after transaction costs. We also note a reduction in maximum drawdowns after its application. Finally, concerning our proposed fitness function for NNs, we note that all our networks produce substantial profitability in both in- and out-of-sample periods. Moreover, we observe that the ranking of our models is almost the same in statistical and trading terms. This allows us to argue that our NNs were trained in a way that allowed them to increase not only their statistical accuracy but also their trading efficiency.

The newly introduced ARBF-PSO methodology expands in many directions the current prevailing computational intelligence and statistical models for financial forecasting and trading. It is fully adaptive, replacing some hard to tune model parameters such as the neural network's architecture with softer ones such as the initial size of the population of particles for the particle swarm optimization module. Furthermore, the simultaneous optimization of the RBF's structure and parameters in a single optimization procedure prevents the data snooping effect which is present in existing methodologies for neural networks structure and parameters optimization and leads to overestimating their performance. In the end, ARBF-PSO deploys a novel fitness function which was proved to be able to pilot candidate solutions to more profitable trading strategies. Our results should go some way towards convincing scientists and decision makers to experiment further beyond the bounds of the more traditional Operational Research models.

APPENDIX

A.1 Neural Networks Characteristics

In the table 12 below, we present the characteristics of the neural networks with the best trading performance in in-sample sub-period which we used in our committees. The choice of the characteristics was based after an extensive experimentation in the in-sample sub-period. For the ARBF-PSO algorithm the choice of the number of hidden nodes is incorporated within the network's algorithm (the maximum possible number of hidden nodes in our ARBF-PSO algorithm is set to 100) and the training is fully adaptive (see Section 4.2.3.2). Therefore no further experimentation was needed.

	Parameters	**MLP**	**RNN**	**PSI**	**ARBF-PSO**
EUR/USD	*Learning algorithm*	*Gradient descent*	*Gradient descent*	*Gradient descent*	*PSO*
	Learning rate	*0.003*	*0.002*	*0.5*	*NA**
	Momentum	*0.005*	*0.004*	*0.5*	*NA**
	Iteration steps	*30000*	*20000*	*20000*	*NA**
	Initialisation of weights	*N(0,1)*	*N(0,1)*	*N(0,1)*	*NA**
	Input nodes	*9*	*8*	*10*	*7*
	Hidden nodes	*8*	*7*	*6*	*14*
	Output node	*1*	*1*	*1*	*1*
EUR/GBP	*Learning algorithm*	*Gradient descent*	*Gradient descent*	*Gradient descent*	*PSO*
	Learning rate	*0.002*	*0.003*	*0.5*	*NA**
	Momentum	*0.004*	*0.005*	*0.5*	*NA**
	Iteration steps	*35000*	*30000*	*25000*	*NA**
	Initialisation of weights	*N(0,1)*	*N(0,1)*	*N(0,1)*	*NA**
	Input nodes	*8*	*8*	*9*	*8*
	Hidden nodes	*9*	*7*	*5*	*9*
	Output node	*1*	*1*	*1*	*1*
EUR/JPY	*Learning algorithm*	*Gradient descent*	*Gradient descent*	*Gradient descent*	*PSO*
	Learning rate	*0.003*	*0.003*	*0.5*	*NA**
	Momentum	*0.005*	*0.005*	*0.5*	*NA**
	Iteration steps	*30000*	*30000*	*30000*	*NA**
	Initialisation of weights	*N(0,1)*	*N(0,1)*	*N(0,1)*	*NA**
	Input nodes	*10*	*9*	*8*	*9*
	Hidden nodes	*13*	*11*	*6*	*10*
	Output node	*1*	*1*	*1*	*1*

Table 12: Network characteristics

* The learning rate, the momentum, the number of iterations steps and the initialization of weights are the characteristics that are describing the training process of our NNs benchmarks which are trained through gradient descent. Our proposed ARBF-PSO algorithm is fully adaptive and is trained through a PSO approach (see Section 4.2.3.2).

REFERENCES

Adeodato, P., Arnaud,A., Vasconcelos, G. Cunha, R. and Monteiro, D. (2011), 'MLP Ensembles Improve Long Term Prediction Accuracy Over Single Networks', *International Journal of Forecasting*, 27, 3 661-671.

Andreou, P., Charalampous, C. and Martzoukos, S. (2008), 'Pricing and Trading European Options by Combining Artificial Neural Networks and Parametric Models with Implied Parameters', *European Journal of Operational Research,* 185, 3, 1415-1433.

Bekaert, G., and G. Wu. (2000), 'Assymetry Volatility and Risk in Equity Markets', *The Review of Financial Studies*, 13, 1, 1–42.

Bekiros, S. D. and Georgoutsos, D. A. (2008), 'Direction-of-change Forecasting Using a Volatility-Based Recurrent Neural Network', *Journal of Forecasting*, 27, 407–417.

Bekiros, S. D. (2010), 'Fuzzy adaptive decision-making for boundedly rational traders in speculative stock markets', *European Journal of Operational Research*, 202, 1, 285-293.

Bertolini, L. (2010), *Trading Foreign Exchange Carry Portfolios*, PhD Thesis, Cass Business School, City University London.

Box, G., Jenkins, G. and Gregory, G. (1994), *Time Series Analysis: Forecasting and Control*, Prentice-Hall, New Jersey.

Brooks, C. (2003), *Introductory Econometrics for Finance*, Cambridge University Press, Cambridge.

Broomhead, S. and Lowe, D. (1988), 'Multivariate Functional Interpolation and Adaptive Networks', *Complex Systems*, 2, 321-355.

Dhamija, A. and Bhalla, V. (2011), 'Exchange rate forecasting: comparison of various architectures of neural networks', *Neural Computing & Applications*, 3, 20, 355-363.

Diebold, F. X. and Mariano, R. S. (1995), 'Comparing Predictive Accuracy', *Journal of Business and Economic Statistics*, 13, 253-263.

Ding, H., Xiao, Y. and Yue, J. (2005), 'Adaptive Training of Radial Basis Function Networks Using Particle Swarm Optimization Algorithm', *Lecture Notes in Computer Science*, 3610, 119-128.

Donaldson, R. G. and Kamstra, M. (1996), 'Forecast Combining with Neural Networks', *Journal of Forecasting*, 15, 49–61.

Dunis, C. and Miao, J. (2005), 'Optimal Trading Frequency for Active Asset Management: Evidence from Technical Trading Rules', *Journal of Asset Management*, 5, 5, 305-326.

Dunis, C. and Nathani, A. (2007), 'Quantitative Trading of Gold and Silver using Nonlinear Models', *Neural Network World*, 16, 2, 93-111.

Dunis, C., Laws, J. and Sermpinis, G. (2010), 'Modelling and Trading the EUR/USD Exchange Rate at the ECB fixing', *The European Journal of Finance*, 16, 6, 541-560.

Dunis, C., Laws, J., and Sermpinis, G. (2011), 'Higher order and recurrent neural architectures for trading the EUR/USD exchange rate', *Quantitative Finance*, 11, 4, 615-629.

Elman, J. L. (1990), 'Finding Structure in Time', *Cognitive Science*, 14, 179-211.

Fulcher, J., Zhang, M. and Xu, S. (2006), 'The Application of Higher-Order Neural Networks to Financial Time Series', in Kamruzzaman, J., Begg, R. and Sarker, R. [eds.] *Artificial Neural Networks in Finance and Manufacturing*, Hershey, PA: Idea Group, London.

Glosten, L., Jagannathan, R. and Runkle, D. (1993), 'On the Relation Between the Expected Value and the Volatility of the Nominal Excess Returns on Stocks', *Journal of Finance*, 48, 5, 1779-1801.

Guégan, D. and Huck, N. (2004), 'On the Use of Nearest Neighbours to Forecast in Finance', *Note de Recherche IDHE-MORA*, n 10-2004, ENS Cachan.

Huck, N. (2010). 'Pairs Ttrading and Outranking: The Multi-Step-Ahead Forecasting Case', *European Journal of Operational Research*, 207, 3, 1702-1716.

Kaastra, I. and Boyd, M. (1996), 'Designing a neural network for forecasting financial and economic time series', *Neurocomputing*, 10, 3, 215-236.

Kennedy J. and Eberhart R.C. (1995), 'Particle Swarm Optimization', *Proceedings of the IEEE International Conference on* Neural *Networks*, 4, 1942-1948.

Kiani, K. and Kastens, T. (2008), 'Testing Forecast Accuracy of Foreign Exchange Rates: Predictions from Feed Forward and Various Recurrent Neural Network Architectures', *Computational Economics*, 4, 32, 383-406.

Lisboa, P. and Vellido, A. (2000), 'Business Applications of Neural Networks', vii-xxii, in P. Lisboa, B. Edisbury and A. Vellido [eds.] *Business Applications of Neural Networks: The State-of-the-Art of Real-World Applications,* World Scientific, Singapore.

Matias, J. M. and Reboredo, J. C. (2012), 'Forecasting Performance of Nonlinear Models for Intraday Stock Returns', *Journal of Forecasting*, 31, 172–188.

Moody, J. and Darken, C. J. (1989), 'Fast Learning in Networks of Locally Tuned Processing Units', *Neural Computation*, 1, 2, 281-294.

Nekoukar, V. and Beheshti, H. (2010), 'A Local Linear Radial Basis Function Neural Network for Financial Time-Series Forecasting', *Applied Intelligence*, 33, 3, 352-356.

Panda, C. and Narasimhan, V. (2007), 'Forecasting exchange rate better with artificial neural network', *Journal of Policy Modeling*, 29, 2, 227-236.

Pesaran, M.H., and Timmermann, A. (1992), 'A Simple Nonparametric test of Predictive Performance', *Journal of Business and Economic Statistics*, 10, 4, 461-465.

Qi, M. and Wu, Y., (2006), 'Technical trading-rule profitability, data snooping, and reality check: Evidence from the foreign exchange market', *Journal of Money, Credit and Banking*, 38, 2135–2158.

Sermpinis, G., Laws, J., and Dunis, C. (2012), 'Modelling and Trading the Realised Volatility of the FTSE100 Futures with Higher Order Neural Networks', *European Journal of Finance*, In press.

Shapiro, A. F. (2000), 'A Hitchhiker's Guide to the Techniques of Adaptive Nonlinear Models', *Insurance, Mathematics and Economics*, 26, 2, 119-132.

Shen, W., Guo, X., Wu, C. and Wu, D. (2011), 'Forecasting Stock Indices Using Radial Basis Function Neural Networks Optimized by Artificial Fish Swarm Algorithm', *Knowledge-Based Systems*, 24, 3, 378-385.

Shin, Y. and Ghosh, J. (1991) 'The Psi-Sigma Network: An Efficient Higher-Order Neural Network for Pattern Classification and Function Approximation', *Proceedings IJCNN*, Seattle, 13-18.

Tenti, P. (1996), 'Forecasting Foreign Exchange Rates Using Recurrent Neural Networks', *Applied Artificial Intelligence*, 10, 6, 567-581.

Yang, J., Su, X., and Kolari, J. W. (2008), 'Do Euro Exchange Rates Follow a Martingale? Some Out-of-Sample Evidence', *Journal of Banking and Finance*, 32, 729 - 740.

Yang, J., Cabrera, J. and Wang, T. (2010), 'Nonlinearity, Data-Snooping, and Stock Index ETF Return Predictability', *European Journal of Operational Research*, 200, 2, 498-507.

Zhang, G., Patuwo, B. and Hu, M. (1998), 'Forecasting with artificial neural networks: The state of the art', *International Journal of Forecasting*, 14, 1, 35-62.

Zhang, M. (2009), '*Artificial Higher Order Neural Networks for Economics and Business*', IGI Global, Hershey.

Wang, J.J, Wang, J.Z., Zhe-George, Z. and Shu-Po, G. (2011), 'Stock Index Forecasting Based on a Hybrid Model', *Omega*, In press

White, H., (2000), 'A reality check for data snooping', *Econometrica*, 68, 1097–1126.

Research Highlights

We introduce a hybrid Neural Network architecture of Particle Swarm Optimization and Adaptive Radial Basis Function.
We introduce a time varying leverage trading strategy.
We introduce a Neural Network fitness function for financial forecasting purposes.