International Journal of Hybrid Intelligent Systems 7 (2010) 75–87
DOI 10.3233/HIS-2010-0106
IOS Press

CO^{2}RBFN for short-term forecasting of the extra virgin olive oil price in the Spanish market

M.D. Pérez-Godoy[a,*], P. Pérez[a], A.J. Rivera[a], M.J. del Jesus[a], C.J. Carmona[a], M.P. Frías[b] and M. Parras[c]
[a]*Department of Computer Science, University of Jaén, Campus Las Lagunillas, 23071 Jaén, Spain*
[b]*Department of Statistics and Operation Research, University of Jaén, Campus Las Lagunillas, 23071 Jaén, Spain*
[c]*Department of Marketing, University of Jaén, Campus Las Lagunillas, 23071 Jaén, Spain*

Abstract. This paper presents the adaptation of CO^{2}RBFN, an evolutionary cooperative-competitive hybrid algorithm for the design of Radial Basis Function Networks, for short-term forecasting of the price of extra virgin olive oil. In the proposed cooperative-competitive environment, each individual represents a Radial Basis Function, and the entire population is responsible for the final solution. In order to calculate the application probability of the evolutive operators over a certain Radial Basis Function, a Fuzzy Rule Based System has been used.
The olive oil time series have been analyzed using CO^{2}RBFN. The results obtained have been compared with Auto-Regressive Integrated Moving Average (ARIMA) models and other data mining methods such as a fuzzy system developed with a GA-P algorithm, a multilayer perceptron trained with a conjugate gradient algorithm, and a radial basis function network trained using an LMS algorithm. The experimentation shows the high efficiency achieved by these methods, especially the data mining methods, which have slightly outperformed the ARIMA methodology.

1. Introduction

Olive oil is an oily juice obtained from the olive, a traditional tree crop of the Mediterranean region. Quality olive oil is obtained from fresh fruit, avoiding any treatment which, during extraction or storage, alters the chemical properties of its components. Both the International Oil Council and the EU regulation define Virgin Olive Oil as a product made from 100% olives (no additives, colorants, flavorings, or any other foreign matter allowed). Extra-virgin olive oil comes from cold pressing of the olives, contains no more than 0.8% acidity, and is judged to have a superior taste.

Olive oil has become an important business sector in a continuously expanding market. Spain is the first olive oil producing country in the world with an average annual production of 700,000–800,000 tons. Ja én is the most productive province of Spain, with an average annual harvest of 400,000–500,000 tons. Spain is also the first exporting country in the world, with annual average exports in the last 10 years of more than 300,000 tons. The agents involved in this sector are interested in the use of forecasting methods for the olive price. This is especially important in the official Market for the negotiation of futures contracts for olive oil (http://www.mfao.es): a society whose objective is to negotiate an appropriate price for the olive oil at the moment it is to be sold at a fixed time in the future. In this context, an accurate prediction of this price in the future could increase the global benefits.

In this context, the data provided for the design of the prediction system are the weekly extra-virgin olive oil prices obtained from *Poolred*,[1] an initiative of the

*Corresponding author. E-mail: lperez@ujaen.es.

[1]http://www.oliva.net/poolred/.

Foundation for the Promotion and Development of the Olive and Olive Oil located in Jaén, Spain. The data are a set of regular time-ordered observations of a quantitative characteristic of an individual phenomenon taken at successive periods or points of time, called time series. The problems in which the data are not independent but also have a temporal relationship are called time series forecasting problems.

Time series forecasting is an active research area and a typical paradigm for evaluating it are statistic models [16], such as ARIMA, and data mining methods. At the beginning of the 1970s the analysis of time series by means of Auto-Regressive Integrated Moving Average (ARIMA) methods [5], was a small revolution. These well-known techniques generate statistical forecasting models under the assumption of linearity among variables. Data mining is a research area concerned with extracting non-trivial information contained in a database, and has also been applied to time series forecasting. Among data mining techniques, mainly neural networks [9,19,37,44] and fuzzy rule based systems [3, 23,25,29,48] have been applied to this kind of problem.

In this paper, a hybrid cooperative-competitive evolutionary method for the design of RBFNs, CO^{2}RBFN is adapted to solving time series forecasting problems. The results obtained using CO^{2}RBFN are also compared with ARIMA methodology and other hybrid intelligent systems methods such as a Fuzzy System developed with a GA-P algorithm (FuzzyGAP), a MultiLayer Perceptron Network trained using a Conjugate Gradient learning algorithm (MLPConjGrad), and a classical design method for Radial Basis Function Network learning (RBFNLMS).

This paper is organized as follows: Section 2 discusses generalities about time series forecasting, describes the classical ARIMA method and reviews the RBFN design for forecasting problems. In Section 3 the extension of CO^{2}RBFN to time series forecasting is presented. The study and results obtained for the forecast methods are detailed in Section 4. In Section 5, conclusions and future works are outlined.

2. Preliminaries

2.1. Time series forecasting

A time series is a set of regular time-ordered observations of a quantitative characteristic of an individual or collective phenomenon taken at successive and in most cases equidistant, periods / points of time. The goal of any basic forecasting method is to predict an outcome from a set of past values.

The reasons for the study of time series are very diverse:

- the need to predict the behavior of a variable in the future,
- the need to control a given process,
- enhancing profits for a company, anticipating price increases or decreases in the market,
- the simulation of phenomena that can not be implemented physically,
- the generation of new physical or biological theories, and so on.

In general, the ultimate goal is always to increase our knowledge of a phenomenon or aspect of our environment data from past and present. Therefore, the main aim is to extract the regularities that are observed in the past behavior of the variable, i.e. obtain the mechanism that generates it, in order to have a better understanding of it in the future.

The type of analysis we use, as the well as models built on the basis of the study, depends largely on the type of question we wish to answer. When the comments correspond to a single variable, the analysis of time series usually aims at building a model to explain the structure (description) and anticipate the evolution (prediction) of the variable interest. Therefore, in the descriptive mode we can use the data to find patterns of behavior of the data, facts and rules of partnerships between occurrences, as well as discovering anomalies in behavior and generally characterizing the data. In the predictive mode, the data are analyzed in order to discover a model of future behavior from them for estimating, in advance, possible values and trends. For the olive oil market and specifically for the Futures Market, estimation of the olive oil price in the near or medium future is very useful for determining the price today for sale in the mentioned period of times. This is the reason we use mainly a predictive analysis in this paper.

2.2. ARIMA methodology

ARIMA models, also called Box-Jenkins models [5], predict variable's present values from its past values.

The development of an ARIMA methodology consists of the search for an ARIMA(p, d, q) model, which is able to generate the time series object of the study. Here p is the value for the auto-regressive parameter, d is the order of differentiation and q is the moving average parameter.

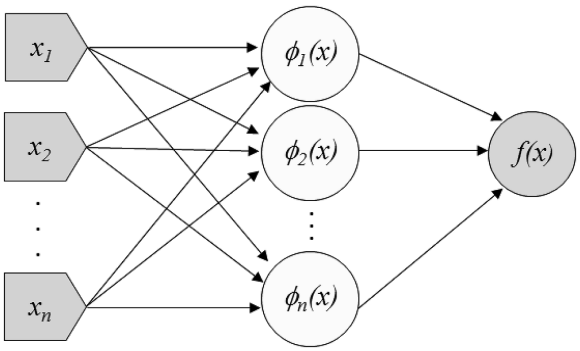

Fig. 1. RBFN Topology for time series forecasting.

ARIMA modeling involves the follow stages: (1) Identification of the model or the initial *p*, *d*, and *q* parameters; (2) Estimation of the *p* and *q* parameters; (3) Diagnosis of the residuals in order to investigate model adequacy.

The purpose of the identification phase is to select the most suitable model among candidates for modeling the time series. To do so, first it is necessary to achieve a stationary time series, that is one in which neither the average nor the variance or autocorrelations depend on time. At this moment, *d* is determined as the necessary order of differentiation for stabilising the series. In order to obtain the best model to fit the series, the sample partial autocorrelation function is compared with a catalogue of graphic patterns which are typical of the different models proposed. Then *p*, and *q* are calculated.

Once the model has been chosen the Maximum Likelihood Estimation (MLE) [10] is the most popular general approach to parameter estimation. Afterwards, an analysis of residuals (difference between the value provided by the model and that actually observed) is carried out in order to verify whether adjustment of the model to our data is adequate. Residuals must be random and normally distributed. If not, the process must be repeated and another model found.

2.3. RBFNs for forecasting problems

Radial Basis Function Networks (RBFNs) are an important artificial neural network paradigm [6] with interesting characteristics such as a simple topological structure or universal approximation ability [34]. They have been successfully used in time series prediction [26,27,31,42,44,45,49].

From a structural point of view, an RBFN is a feedforward neural network with three layers: an input layer with n nodes, a hidden layer with m neurons or RBFs, and an output layer with, in time series forescasting, one node (see Fig. 1).

The m neurons of the hidden layer are activated by a radially-symmetric basis function, $\phi_i : R^n \rightarrow R$, which can be defined in several ways, the Gaussian function being the most widely used: $\phi_i(\vec{x}) = \phi_i(e^{-(\|\vec{x}-\vec{c_i}\|/d_i)^2})$, where $\vec{c_i} \in R^n$ is the centre of basis function ϕ_i, $d_i \in R$ is the width (radius), and $\|\|$ is typically the Euclidean norm on R^n. This expression is the one used in this paper as the Radial Basis Function (RBF). The output nodes implement the following function:

$$f(\vec{x}) = \sum_{i=1}^{m} w_i \phi_i(\vec{x}) \quad (1)$$

The output of one basis function will be high when the input vector and the centre of this basis function are closer, always taking into account the value of the radius. The weights w_i show the contribution of an RBF to the respective output node, and therefore the output nodes implement the weighted sum of RBF outputs Eq. (1).

The objective of any RBFN design process is to determine centres, widths and the linear output weights connecting the RBFs to the output neuron layer. The most traditional learning procedure has two stages: first, unsupervised learning of centres and widths is used, and finally output weights are established by means of supervised learning. Clustering techniques [35] are normally used to adjust the centres. Regarding the widths, they may all be given the same value, may reflect the width of the previously calculated clusters (i.e., RBFs), or may be established as the average distance between RBFs, among other possibilities. In order to obtain the weights in the second stage, algorithms such as Least Mean Square (LMS) [46] or Singular Value Decomposition (SVD) [13] can be used. As for example in [14] this methodology has been applied to forecasting the groundwater head data. The self-organizing map (SOM) has been employed to complete the clustering task and place the centres. Weights are calculated by means of a gradient descent technique [28,44,49] are other methods for RBFN design based on clustering techniques and/or simple training weights techniques applied to forecasting tasks.

As well as this typical methodology, different strategies for RBFN design can be found based on deciding which RBFs to aggregate [38] or eliminate [47], and may be considered as forward or backward selection methods. In [26] a learning algorithm with growing and pruning techniques to RBFN design is described.

A Welsch M-estimator and median scale estimator are employed to get rid of the influence from the noise. The method is applied to noisy time series prediction. In [33] an aggregate method is used in forecasting inflation. One disadvantage of the aforementioned methods is that they could become trapped in local optima.

Another important paradigm for the RBFN design which overcomes this limitation is Evolutionary Computation (EC). EC [4,12,20], uses natural evolution and stochastic searching to design optimization algorithms. More precisely, EC maintains a population of individuals, which evolves according to the operators as mutation, recombination or selection and each individual in the population receives a measure of its fitness in the environment.

A review of EC applied to RBFN design can be found in [7,17]. Examples of RBFN design algorithms applied to time series forecasting can be found in [8,15,31, 41,43]. In most of the proposals within this evolutionary paradigm an individual represents a whole RBFN (Pittsburgh codification), and different operators are applied to the entire population to improve individual fitness. Nevertheless EC presents some difficulties for certain learning problems, especially in the evaluation of independent subcomponents (RBFs) [39].

An alternative to the classical Pittsburgh codification are the cooperative-competitive evolutionary or cooperative-coevolutive strategies [39,42,45], which provide a framework within which an individual of the population represents only a part of the solution, evolving in parallel, competing to survive but at the same time cooperating in order to find a common solution (the complete RBFN in our case). This approach has the advantage of being computationally less complex, since an individual does not represent the whole solution but only a part of it. With this approach, two main problems must be addressed: credit assignment, or the fitness allocated to each individual according to its contribution to the final solution, and the mechanism used in order to maintain diversity among individuals of the population.

Finally, Particle Swarm Optimization (PSO) can also be used in RBFN design. PSO [24] is a population-based stochastic optimization technique inspired by the social behavior of bird-flocking or fish-schooling. PSO shares many similarities, such as Genetic Algorithms, with EC techniques. The system is initialized with a population of random solutions and optimises them by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation.

In [27] a method for forecasting the stress and displacement nonlinear time series is proposed based on constructing RBFNs using the PSO algorithm. After determining the number of units in an RBF layer using k-means, all parameters such as central position, shape parameter and weights of RBFN are estimated dynamically with particle swarm optimization.

The authors developed a hybrid cooperative competitive evolutionary proposal for RBFN design, CO^{2}RBFN, applied to the classification problem [36]. In this paper, a new version of CO^{2}RBFN is presented in order to deal with the time series forecasting problem. This new version also improves the efficiency and the balance between exploration-exploitation in the evolutionary design process.

3. CO^{2}RBFN for time series forecasting

CO^{2}RBFN, is an evolutionary cooperative-competitive hybrid algorithm for the design of Radial Basis Function Networks (RBFNs). In this paper the algorithm is adapted to address the problem of short-term forecasting.

In CO^{2}RBFN, each individual of the population represents, with a real representation, a basis function (RBF) and the entire population is responsible for the final solution. The individuals cooperate towards a definitive solution, but they must also compete for survival.

In this cooperative-competitive environment, in which the solution depends on the behaviour of many components, the fitness of each individual is known as credit assignment. In order to measure the credit assignment of an individual, three factors have been proposed to evaluate the role of each RBF in the network. These factors are: the RBF contribution to the network output, the error in the basis function radius, and the degree of overlapping among RBFs.

There are four evolutionary operators that can be applied to an RBF: an operator that eliminates the RBF, two operators that mutate the RBF, and finally an operator that maintains the RBF parameters in order to explore and exploit the search space and to preserve the best RBF, respectively.

The application of the operators is determined by a Fuzzy Rule-Based System (FRBS). The inputs of this system are the three parameters used for credit assignment and the outputs are the operators' application probability. To design the set of rules we must take into account the fact that an RBF is worse if its contribution is low, its error is high and its overlapping is also high, otherwise it is better. In this way the probability of

1. Initialize RBFN
2. Train RBFN
3. Evaluate RBFs
4. Apply operators to RBFs
5. Substitute the eliminated RBFs
6. Select the best RBFs
7. If the stop condition is not verified go to step 2

Fig. 2. Main steps of CO²RBFN.

eliminating an RBF is high when this RBF is worse and so on.

The main steps of CO²RBFN, explained in the following subsections, are shown in the pseudocode in Fig. 2.

3.1. RBFN initialization

To define the initial network, with a number of RBFs established by the size of the population, a simple process is used: a specified number, m, of neurons (i.e. the size of population) is randomly allocated among the different patterns of the training set. To do so, each RBF centre, $\vec{c}_i$, is randomly established to a pattern of the training set. The RBF widths, d_i, will be set to half the average distance between the centres. Finally, the RBF weights, w_i, are set to zero.

3.2. RBFN training

During this stage, RBF weights are trained. The Least Mean Square (LMS) algorithm [46] has been used to calculate the RBF weights. This technique exploits the local information that can be obtained from the behaviour of the RBFs. The equation shows the update of the weights.

$$\overline{w}_{k+1} = \overline{w}_k + \alpha \frac{e_k \overline{x}_k}{|\overline{x}_k^2|} \tag{2}$$

where k is the number of iteration, $\overline{w}_{k+1}$ is the next value of the weight vector, $\overline{w}_k$ is the present value of the weight vector, $\overline{x}_k$ is the value of the actual input pattern vector. The present linear error, e_k, is defined as the difference between the desired output and the output network before adaptation. The α value is the *speed of learning*, it measures the size of the adjustment to be made. The choice of α controls stability and speed of convergence.

3.3. RBF evaluation

A credit assignment mechanism is required in order to evaluate the role of each basis function in the cooperative-competitive environment.

For an RBF ϕ_i, three parameters, a_i, e_i, o_i are defined:

– The contribution, a_i, of the RBF ϕ_i, $i = 1 \dots m$, is determined by considering the weight, w_i, and the number of patterns of the training set inside its width, npi_i. An RBF with a low weight and few patterns inside its width will have a low contribution:

$$a_i = \begin{cases} |w_i| & if \quad npi_i > q \\ |w_i| * (npi_i/q) & otherwise \end{cases} \tag{3}$$

where q is the average of the npi_i values minus the standard deviation of the npi_i values.

– The error measure, e_i, for each RBF ϕ_i, is obtained by calculating the Mean Absolute Percentage Error (MAPE):

$$e_i = abs\left(\frac{\sum_{\forall p_i}\left|\frac{f(pi_i)-y(pi_i)}{f(pi_i)}\right|}{npi_i}\right) \tag{4}$$

where $f(pi_i)$ is the output of the model for the point pi_i, inside the width of RBF ϕ_i, $y(pi_i)$ is the real output at the same point, and npi_i is the number of points inside the width of RBF ϕ_i.

– The overlapping of the RBF ϕ_i and the other RBFs is quantified by using the parameter o_i. This parameter is calculated by taking into account the fitness sharing [12] methodology, whose aim is to maintain diversity in the population. This factor is expressed as:

$$o_i = \sum_{j=1}^{m} o_{ij} \tag{5}$$

$$o_{ij} = \begin{cases} (1 - \|\phi_i - \phi_j\|/d_i) \; if \|\phi_i - \phi_j\| < d_i \\ 0 \qquad\qquad otherwise \end{cases}$$

where o_{ij} measures the overlapping of the RBF ϕ_i y ϕ_j $j = 1 \dots m$.

3.4. Applying operators to RBFs

In this algorithm four operators have been defined in order to be applied to the RBFs:

– Operator Remove: eliminates an RBF.

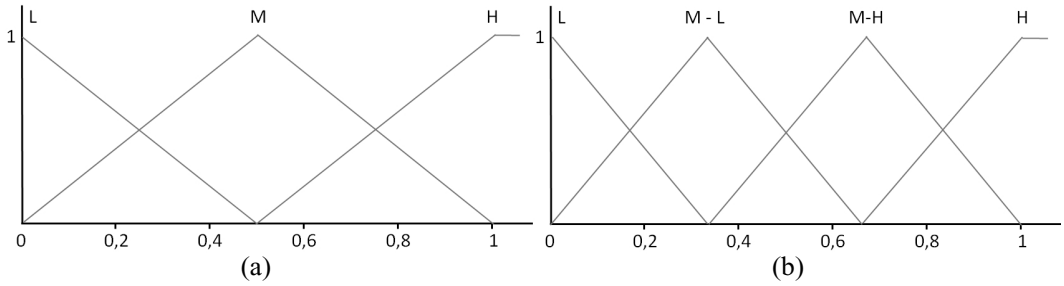

Fig. 3. a) input variables membership functions for the FRBS. b) output variables membership function.

Table 1
Fuzzy rule base representing expert knowledge in the design of RBFNs

	Antecedents			Consequents			
	v_a	v_e	v_o	p_{remove}	p_{rm}	p_{bm}	p_{null}
R1	L			M-H	M-H	L	L
R2	M			M-L	M-H	M-L	M-L
R3	H			L	M-H	M-H	M-H
R4		L		L	M-H	M-H	M-H
R5		M		M-L	M-H	M-L	M-L
R6		H		M-H	M-H	L	L
R7			L	L	M-H	M-H	M-H
R8			M	M-L	M-H	M-L	M-L
R9			H	M-H	M-H	L	L

– Operator Random Mutation: modifies the centre and width of an RBF. The width is altered with a probability inversely proportional to the number of features of the regression problem (n), at a percentage below 50% of the old width. The coordinates of the centre increase or decrease at a percentage below 50% of the width. The number of coordinates to be mutated is randomly obtained and is a number below 25% of the total number of coordinates.
– Operator Biased Mutation: modifies the width and all coordinates of the centre using local information of the RBF environment. The technique used follows the recommendations [11] that are similar to those used by the algorithm LMS algorithm. The error for the patterns within the radius of the RBF, ϕ_i, are calculated. For each coordinate of the center and the radius a value Δc_{ij} and Δd_i respectively are calculated. The new coordinates and the new radius are obtained by changing (increasing or decreasing) its old values to a random number (between 5% and 50% of its old width), depending on the sign of the value calculated.

$$\Delta d_i = \sum_k e(\overrightarrow{p_k}) \cdot w_i \qquad (6)$$

where $e(\overrightarrow{p_k})$ is the error for the pattern $\overrightarrow{p_k}$.

$$\Delta c_{ij} = sign(c_{ij} - p_{kj}) \cdot e(\overrightarrow{p_k}) \cdot w_i \qquad (7)$$

– Operator Null: in this case all the parameters of the RBF are maintained.

These mutation operators allow us to obtain an appropriate balance between exploitation and exploration, which is a desirable feature in every evolutionary algorithm. Biased mutations use local information from the RBF environment in order to achieve an optimal adaptation. On the other hand, random mutations carry out alterations that lead to the exploration of the environment and thus avoid local optimums.

The operators are applied to the whole population of RBFs. The probability for choosing an operator is determined by means of a Mandani-type fuzzy rule based system [30] which represents expert knowledge about the operator application in order to obtain a simple and accurate RBFN.

The inputs of this system are parameters a_i, e_i and o_i used for defining the credit assignment of the RBF ϕ_i. These inputs are considered as linguistic variables va_i, ve_i and vo_i. The outputs, p_{remove}, p_{rm}, p_{bm} and p_{null}, represent the probability of applying Remove, Random Mutation, Biased Mutation and Null operators, respectively. The number of linguistic labels has been empirically determined and the fuzzy sets have been defined according to their meaning. Figure 3 shows the membership functions for the input and output variables respectively. As defuzzification method the centre of area/gravity technique is used. Table 1 shows the rule base used to relate the described antecedents and consequents. In the table each row represents one rule. For example, the interpretation of the first rule is: If the contribution of an RBF is Low Then the probability of applying the operator Remove is Medium-High, the probability of applying the operator Random Mutation is Medium-High, the probability of applying the operator Biased Mutation is Low and the probability of applying the operator Null is Low.

The rule base represents expert knowledge, as mentioned, in the design of RBFNs. It was developed taking into account the fact that an RBF is worse if its contribution (a_i) is low, its error (e_i) is high and its overlapping (o_i) is also high. On the other hand, an RBF is better when its contribution is high, its error is low and its overlapping is also low. A worse RBF indicates that this neuron has problems performing a good role in its environment and therefore, important changes such as random mutations or even removing the RBF must be promoted. In these cases the probability of applying the biased mutation operator and the null operator is low. However, a better neuron implies that the RBF is working well in its environment. In these situations exploitation is promoted increasing the probability of applying the biased mutation operator. The probability of maintaining the neuron with the same parameters, applying the null operator, is also augmented. In these cases the probability of removing the RBF will be low. The probability of applying random mutation is usually high in order to promote a parsimonious evolution. It can be highlighted that this rule base represents general knowledge related with the design of RBFNs. This generic rule base has been successfully applied in classification problems [36].

3.5. Introduction of new RBFs

In this step of the algorithm, the eliminated RBFs are substituted by new RBFs. The new RBF is located in the center of the area with maximum error or in a randomly chosen pattern with a probability of 0.5 respectively.

In the first instance, the areas are defined as a set of neighboring patterns with a width equal to the average of the width of the RBF. The width of the new RBF will be set to the average of the RBFs in the population plus half of the minimum distance to the nearest RBF. Its weights are set to zero.

If it is chosen randomly, the RBF is located in the first pattern found outside any RBF width. The width of the new RBF is set to the average of the RBFs in the population and its weights are set to zero.

3.6. Replacement strategy

After applying the mutation operators, new RBFs appear. The algorithm uses the replacement scheme to decide which new RBFs will be included in the new population. To do so, for each mutated RBF a net, with the child RBF but without the parent RBF, is built. Then, every new net is evaluated (by means its training classification error) in order to determine the RBF (child or parent) with the best behaviour and to include it in the population.

4. Experimentation and results

As mentioned, the data used in this work have been obtained from *Poolred*, http://www.oliva.net/poolred/, an initiative of the Foundation for the Promotion and Development of the Olive and Olive Oil located in Jaén, Spain. The time series dataset contains the weekly extra-virgin olive oil price per kilogram.

The task addressed in this work is that of performing a short-term forecast (next week) of the extra-virgin olive oil price. In this study, the data used are from the 32^{nd} week (August) of the year 2000 to the 52^{nd} week (December) of the year 2005 in Spain. The cases in the data set were divided into two subsets: one for training and the other for testing. The data from the 32^{nd} week of 2000 to the 32^{th} week of 2005 were used for training. The performance of the different methods was tested by estimating the data from the 33^{nd} week to the 52^{nd} week of 2005. The size of datasets have been chosen taking into account the use of ARIMA models. Figure 4 shows the time series data and training and test datasets.

The patterns are composed of $(n-4, n-3, n-2, n-1, n, n+1)$, where the price to forecast is $n+1$ and must be determined from the past prices $n-4$ to n. This decision is justified from the ARIMA analysis, see below, where the Figs 6, 7 and the ARIMA models extracted shown that the prediction of the next value $n+1$ may be related to the five past values.

To estimate prediction capacity, two error measures have been considered: the Mean Squared Error (MSE $= \sum_i^n (f_i - y_i)^2/n$) and the Mean Absolute Percentage Error (MAPE $= \sum_i^n (\mid (f_i - y_i)/f_i \mid)/n$), where f_i is the predicted output of the model and y_i is the desired output.

4.1. Other data mining methods

CO²RBFN has been compared with ARIMA models and other intelligent data mining methods, specifically:

– FuzzyGAP method [40]. A GA-P method [22] uses an evolutionary computation method, a hybrid between genetic algorithms and genetic programming, and optimized to perform symbolic regressions. Each element comprises a chain of parame-

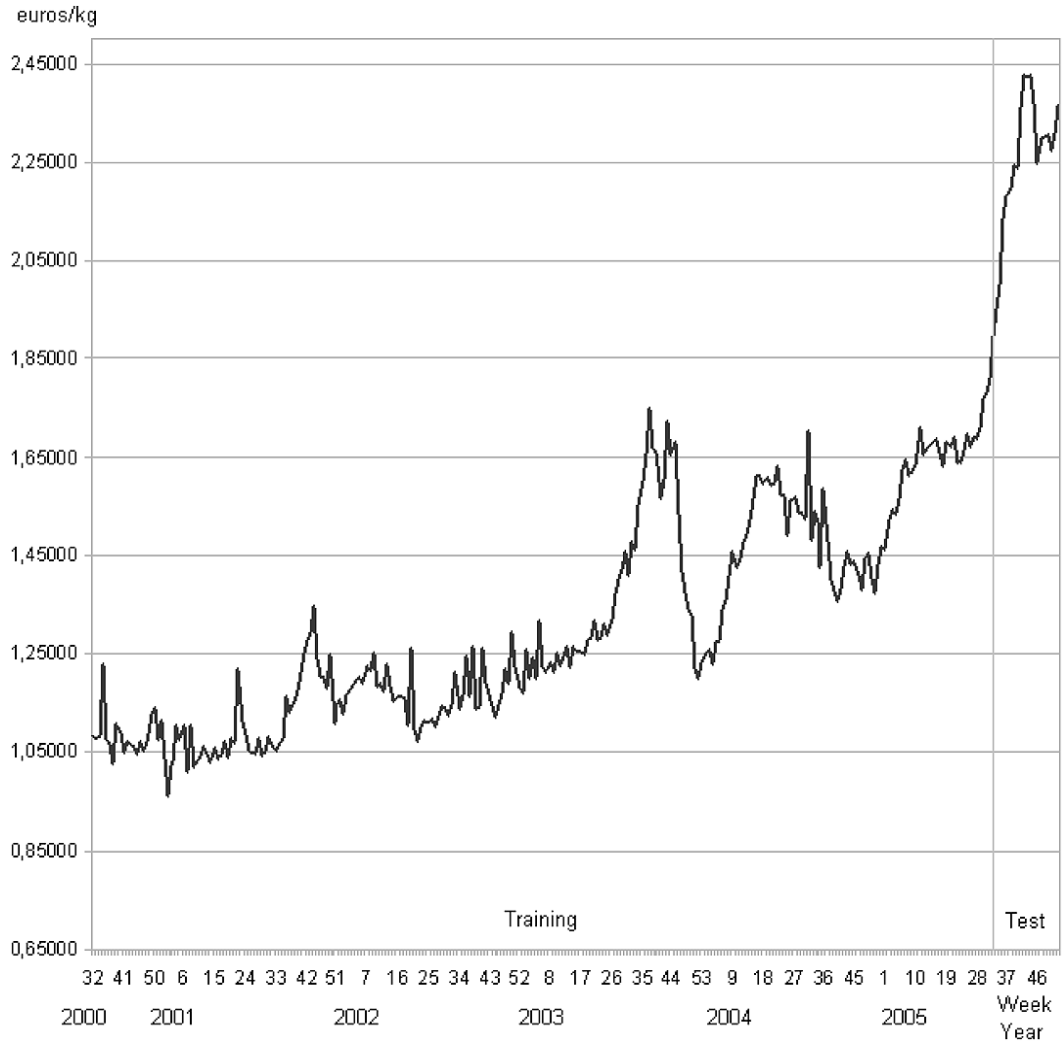

Fig. 4. Weekly extra-virgin olive oil prices in Spain.

ters and a tree which describes a function, depending on these parameters. The two operators by means of which new members of the population are generated are crossover and mutation. In the GA-P algorithm both operations are performed independently over the tree and the parameter chain. In the FuzzyGAP algorithm proposed by [40] the fuzzy sets of the fuzzy model are codified on the terminal nodes of the tree and fuzzy arithmetic operators are used to evaluate the tree. The goal is to find a function g where the difference $y - g(\vec{x})$ is small for every value of $\vec{x}$. In this way, an $\alpha - cut$ is defined as an interval model with a β confidence degree, where the functions g^+ y g^- are defined so that $g^+(\vec{x})$ is the maximum value of the confidence interval and $g^-(\vec{x})$ the minimum. The output of a fuzzy model is defined by the union of the outputs produced by every one of its $\alpha - cuts$ when the same input is applied to all of them.

– MLPConjGrad [32]. A Multilayer Perceptron (MLP) [18] network consists of a set of source nodes forming the input layer, one or more hidden layers of computation nodes, and an output layer of nodes. The input signal propagates through the network layer-by-layer. For the MLP, the most common activation functions, used in the computation nodes, are the logistic sigmoidal $\varphi(x) = 1/1 + e^{-x}$ or the hyperbolic tangent $\varphi(x) = \frac{1-e^{-x}}{1+e^{-1}}$. In regression, the net output is given by $f(\vec{x}) = \sum_i w_i \varphi_i$ where w_i are the weights to be learned. In [21] it is demonstrated that the MLP has a universal approximation abil-

Table 2
Parameters used for CO²RBFN

Parameter	Value
Generations of the main loop	200
Number of RBF's	4

Table 3
Parameters used for FuzzyGAP

Parameter	Value
Numlabels	3
Numrules	8
Popsize	30
Numisland	2
Numitera	10000
Toursize	4
Probmuta	0,01

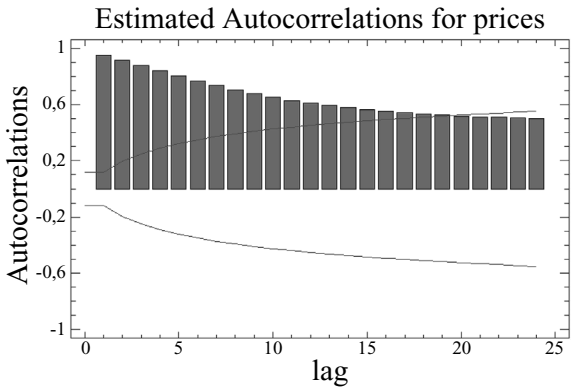

Fig. 5. Sample ACF and 95% confidence limits for the average weekly extra-virgin olive oil price series.

ity. The MLP was also successful in time series prediction [37,44].

The typical learning method for the MLP is the backward-propagation algorithm, which uses the gradient descent algorithm to converge the weights using gradient information. This method presents some drawbacks: It is very slow and there is no automatic way to select configuration parameters.

MLPConjGrad [32] uses the conjugate-gradient algorithm to adjust weight values using the gradient during the backward propagation of errors throughout the network. Compared to gradient descent, the conjugate gradient algorithm takes a more direct path to the optimal set of weight values. Usually, the conjugate gradient is significantly faster and more robust than the gradient descent. The Conjugate gradient also does not require the user to specify learning rate and momentum parameters.

– RBFNLMS. Builds an RBFN with a pre-specified number of RBFs. By means of the K-Means clustering algorithm it chooses an equal number of points from the training set to be the centres of the neurons. Finally, it establishes a single radius for all the neurons as half the average distance between the set of centres. Once the centres and radii of the network have been fixed, the set of weights is analytically computed using the LMS algorithm, Eq. (2) [46].

The parameters used for CO²RBFN are shown in Table 2. The implementation of the rest data mining methods has obtained from KEEL [2]. The main parameters used are set to the values indicated by the authors and they are shown from Table 3 to Table 5, respectively. All the data mining methods (including CO²RBFN) are non deterministic methods and so, have been executed 10 times.

Table 4
Parameters used for MLPConjGrad

Parameter	Value
topologymlp	10

Table 5
Parameters used for RBFNLMS

Parameter	Value
Percent	0,1
Neurons	50

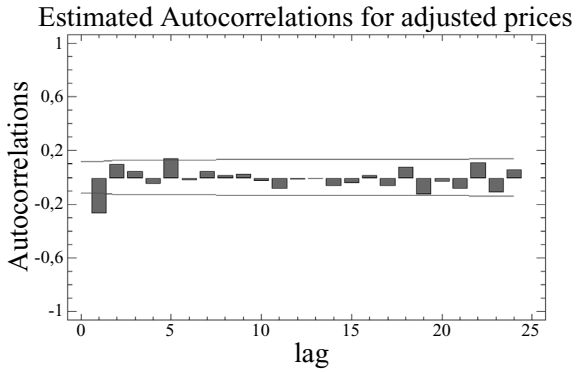

Fig. 6. Sample ACF and 95% confidence limits for the average weekly extra-virgin olive oil price series after one differentiation.

4.2. Results

The preliminary analysis of the weekly extra-virgin olive oil prices portrayed in Fig. 4 shows a nonstationary series, because prices tend to increase over time. This inherent nonstationarity is also confirmed by the graph in Fig. 5, where the sample ACF (Autocorrelation

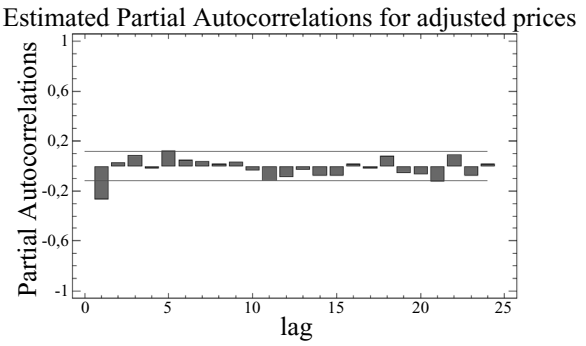

Fig. 7. Sample PACF and 95% confidence limits for the average weekly extra-virgin olive oil price series after one differentiation.

Function) attenuates slowly for the given extra-virgin olive oil price series.

When the series differentiates one time the nonstationarity is removed, as shown by the sample ACF in Fig. 6. The large value at lag one indicates the need for an MA (moving average) parameter in the model. At lag 5, the sample ACF just crosses the 95% confidence limit, and this behaviour may be due to chance or could indicate the need for another parameter in the model. The sample PACF (Partial Autocorrelation Function) in Fig. 7 for the differentiated extra-virgin olive oil price series may be interpreted as attenuating quickly at the first few lags due to the need for MA components.

Based upon this identification information, the weekly extra-virgin olive oil price series could be modeled by an ARIMA(0,1,1) or ARIMA(0,1,5). If the weekly extra-virgin olive oil price series were modeled by an ARIMA(1,1,1), that is, with an AR component, the p-value for the AR(1) term is 0,964692, greater than or equal to 0,05, the reason why it is not statistically significant. Therefore, we would have to consider eliminating the constant term of the model.

The Sample ACF and Sample PACF for the average weekly extra-virgin olive oil price series after one differentiation are quite different to the theoretical ACF and PACF of a model with seasonal patterns.

Afterwards, parameters of the model are calculated using the Maximum Likelihood Estimation (MLE) [10]. Then the Akaike Information Criterion (AIC) [1] is applied, which is the simplest model which provides a good fit to the data. For the model ARIMA (0,1,5) AIC obtains a value of 1947,508 and for the ARIMA (0,1,1) model the AIC value is 1948,930. From these AIC values the ARIMA (0,1,5) model is shown to be better than the ARIMA (0,1,1). The results of the residual tests have satisfied independence, normality and constant variance assumptions.

Table 6
MAPE error in methods and datasets

Method	Test MAPE	Training MAPE
ARIMA (0,1,1)	3,256%	2,879%
ARIMA (0,1,5)	2,659%	2,873%
FuzzyGAP	2,692%	2,868%
MLPConjGrad	2,803%	2,468%
RBFNLMS	2,943%	2,826%
CO^{2}RBFN	2,257%	2,807%

Table 7
MSE error in methods and datasets

Method	Test MSE	Training MSE
ARIMA (0,1,1)	0,00857	0,00254
ARIMA (0,1,5)	0,00541	0,00247
FuzzyGAP	0,00550	0,00248
MLPConjGrad	0,00596	0,00176
RBFNLMS	0,00637	0,00242
CO^{2}RBFN	0,00413	0,00246

Tables 6 and 7 show MAPE and MSE errors obtained by ARIMA models. Figure 8 shows the graphical behavior of the ARIMA models in test data set.

As has been previously demonstrated, the addressed time series data have a positive trend. Therefore it is convenient to differentiate these data once repeating the process for data mining methods, in order to achieve a stationary series where training datasets values are in the same range as test datasets values.

In Tables 6 and 7, the mean values obtained by them are shown. The graphical behavior in the test dataset is shown in Fig. 8.

4.3. Analysis of the results

A first sight of the time series graphic, Fig. 4, shows a non-trivial data series with a final behaviour of the data different from the initial phase. Despite this fact, the ARIMA method and data mining methods have achieved good results both in training datasets and test datasets.

The different methods can be sorted, taking into account their errors test results, as follows: CO^{2}RBFN, ARIMA (0,1,5), Fuzzy GAP, MLPConjGrad, RBFNLMS, and ARIMA(0,1,1). CO^{2}RBFN has obtained the best results. It can be observed that the FuzzyGAP method has a good behaviour similar to the ARIMA (0,1,5) model but significatively inferior to CO^{2}RBFN. On the contrary MLPConjGrad suffers from overlearning of training data, which decreases its prediction ability.

The results of ARIMA(0,1,5) are very important because they imply a good application of ARIMA methodology.

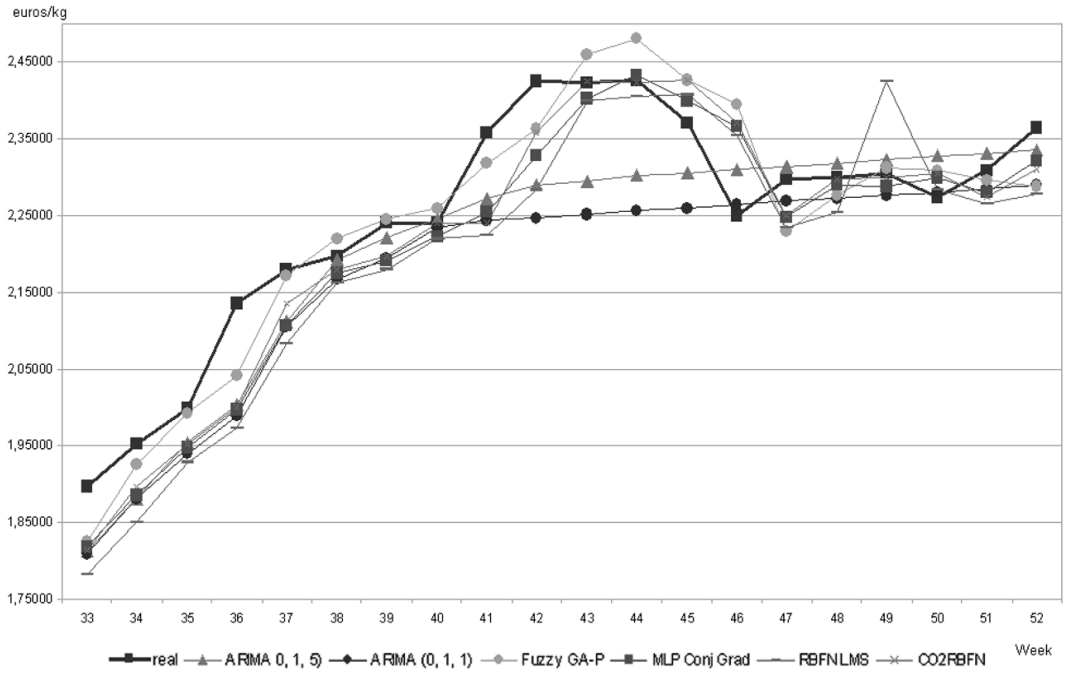

Fig. 8. Olive oil price and its predicted values.

From the behavior shown in Fig. 8 we can conclude that the proposed data mining method CO^{2}RBFN follows the shape of the original data but ARIMA models do not, mainly in the "hill" that the test dataset shows in the middle of the graphic. This may be due to the fact that ARIMA models only have an MA component but no AR (autoregressive) components, with implies a low efficiency for these types of shape, but tests with AR components in ARIMA models were unsatisfactory. This leads us to conclude that for this time series forecasting problem data mining methods are a good solution for accurate prediction in a short time. Among them CO^{2}RBFN is the proposal which has obtained the best results in test data, and so, shows the best prediction ability.

5. Concluding remarks

In this paper, we propose CO^{2}RBFN, a hybrid evolutionary cooperative-competitive algorithm for RBFN design applied to time series forecasting. An important key point of our proposal is the identification of the role (credit assignment) of each basis function in the whole network. In order to evaluate this value for a given RBF three factors are defined and used: the RBF contribution to the network's output, a_i; the error in the basis function radius, e_i; and the degree of overlapping among RBFs, o_i. In order to drive the cooperative-competitive process four operators are used: Remove, Random Mutation, Biased Mutation (based on clustering) and Null. The application of these operators is determined by a fuzzy rule-based system which represents expert knowledge of the RBFN design. The inputs of this system are the three parameters used for credit assignment.

The olive oil sector has an important role in the economy of Spain, especially in Jaén, and forecasting olive oil prices is a key aspect in the development of the sector.

The objective of our research has been to carry out a short term forecast of the weekly extra virgin olive oil price by means of our proposal and to compare its results with the ones obtained by the well-known classical statistical ARIMA method and a set of reliable data mining methods. The data mining methods applied for the comparison are: MLPConjGrad, a multilayer perceptron network which trains which a conjugate gradient algorithm; FuzzyGAP, a fuzzy system developed with a GA-P algorithm and RBFNLMS, a radial basis function network trained with the LMS algorithm.

Our study of the time series data has detected a positive trend of the data that implies a nonstationary series and the convenience of differentiating time series

data. Both the data mining and ARIMA methods have obtained good results, especially data mining methods like the FuzzyGAP or CO^{2}RBFN algorithms.

The proposed data mining method, CO^{2}RBFN, follows the shape of the original data but ARIMA models do not, mainly in the "hill" that the test dataset shows in the middle of the graphic. This may be due to the fact that ARIMA models only haven an MA component but no AR (autoregressive) components, which implies low efficiency for these types of shape, but tests with AR components in ARIMA models were unsatisfactory. This leads us to conclude that for this time series forecasting problem data mining methods are a good solution for accurate prediction in a short time. Among them CO^{2}RBFN is the proposal which has obtained the best results in test data, and so shows the best prediction ability.

As future work exogenous features like meteorology or econometric data can be taken into account in order to increase the performance of the forecast. Also, long term predictions of the olive oil price could be addressed. Another future line is applying CO^{2}RBFN to regression problems.

Acknowledgments

Supported by the Spanish Ministry of Science and Technology under the Projects TIN2008-06681-C06-02, the Andalusian Research Plan TIC-3928 and the Local Project UJA_08_16_30.

References

[1] H. Akaike, A new look at statistical model identification, *IEEE Transactions on Automatic Control* **19** (1974), 716–723.

[2] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. Del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas, J.C. Fernández and F. Herrera, KEEL: A Software Tool to Assess Evolutionary Algorithms for Data Mining Problems, *Soft Computing 13:3* **13**(3) (2009), 307–318.

[3] A. Azadeh, M. Saberi, S.F. Ghaderi, A. Gitiforouz and V. Ebrahimipour, Improved estimation of electricity demand function by integration of fuzzy system and data mining approach, *Energy Conversion and Management*, 2008.

[4] T. Bäck, U. Hammel and H. Schwefel, Evolutionary computation: comments on the history and current state, *IEEE Transaction Evolutive Compututation* **1**(1) (1997), 3–17.

[5] G. Box and G. Jenkins, *Time Series Analysis: Forecasting and Control*, revised edition. San Francisco: Holden Day, 1976.

[6] D. Broomhead and D. Lowe, Multivariable functional interpolation and adaptive networks, *Complex System* **2** (1998), 321–355.

[7] O. Buchtala, M. Klimek and B. Sick, Evolutionary optimization of radial basis function classifiers for data miningapplications, *IEEE Transactions on Systems, Man and Cybernetics Part B* **35**(5) (2005), 928–947.

[8] C. Chen, Y. Wu and B.L. Luk, Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks, *IEEE Transaction Neural Networks* **10**(5) (1999), 1239–1243.

[9] H.C. Co and R. Boosarawongse, Forecasting Thailand's rice export: Statistical techniques vs. artificial neural networks, *Computers and Industrial Engineering* **53**(4) (2007), 610–627.

[10] A.W.F. Edwards, *Likelihood*, Cambridge University Press, Cambridge, United Kingdom, 1972.

[11] J. Ghost, L. Deuser and S. Beck, A neural network based hybrid system for detection, characterization and classification of short-duration oceanic signals, *IEEE Jl Of Ocean Enginering* **17**(4) (1992), 351–363.

[12] D. Goldberg and J. Richardson, Genetic algorithms with sharing for multimodal function optimization, in: *Proc. Second International Conference on Genetic Algorithms*, Grefenstette, ed., Lawrence Erlbaum Associates, 1987, pp. 41–49.

[13] G. Golub and C. Van Loan, *Matrix computations*, (3rd ed.), J. Hopkins University Press, 1996.

[14] L. Gwo-Fong and C. Lu-Hsien, Time series forecasting by combining the radila basis function network and the self-organizing map, *Hydrological Processes* **19** (2005), 1925–1937.

[15] H. Du and N. Zhang, Time series prediction using evolving radial basis function networks with new encoding scheme, *Neurocomputing* **71**(7–9) (2008), 1388–1400.

[16] P.H. Franses and D. van Dijk, Non-linear time series models in empirical finance, *Cambridge University Press*, 2000.

[17] C. Harpham, C.W. Dawson and M.R. Brown, A review of genetic algorithms applied to training radial basis function networks, *Neural Computing and Applications* **13** (2004), 193–201.

[18] S. Haykin, *Neural Networks: A Comprehensive Foundation*, (2nd Edition), Prentice Hall, 1998.

[19] B.F. Hobbs, U. Helman, S. Jitprapaikulsarn, S. Konda and D. Maratukulam, Artificial neural networks for short-term energy forecasting: Accuracy and economic value, *Neurocomputing* **23**(1–3) (1998), 71–84.

[20] J.H. Holland, Adaptation in Natural and Artificial Systems, *The University of Michigan Press*, 1975.

[21] K. Hornik, M. Stinchcombe and H. Wite, Multilayer feedforward networks are universal approximators, *Neural Network* **2** (1989), 359–366.

[22] L. Howard and D. D'Angelo, The GA-P: A Genetic Algorithm and Genetic Programming Hybrid, *IEEE Expert* (1995), 11–15.

[23] J.R. Jang, ANFIS: Adaptative-Network-based Fuzzy Inference System, *IEEE Trans Systems, Man and Cybernetics* **23**(3) (1993), 665–685.

[24] J. Kennedy and R. Eberhart, Particle swarm optimization, *Procedding of the IEEE Int. Conf. on Neural Networks, Piscataway, NJ* (1995) pp. 1942–1948.

[25] M. Khashei, S. Reza Hejazi and M. Bijari, A new hybrid artificial neural networks and fuzzy regression model for time series forecasting, *Fuzzy Sets and Systems* **159**(7) (2008), 769–786.

[26] C. Lee, Y. Chiang, C. Shih and C. Tsai, Noisy time series prediction using M-estimator based robust radial basis function neural networks with growing and pruning techniques, *Expert Systems with Applications* **36**(3 PART 1) (2009), 4717–4724.

[27] G.L. Liang, W.Y. Xu, Y.Z. He and Y.X. Zhao, Study and application of PSO-RBFNN model to nonlinear time series forecasting for geotechnical engineering, *Yantu Lixue/Rock and Soil Mechanics* **29**(4) (2008), 995–1000.

[28] Z. Lin, D. Zhang, L. Gao and Z. Kong, Using an adaptive self-tuning approach to forecast power loads, *Neurocomputing* **71**(4–6) (2008), 559–563.

[29] J. Liu, T.M. McKenna, A. Gribok, B.A. Beidleman, W.J. Tharion and J. Reifman, A fuzzy logic algorithm to assign confidence levels to heart and respiratory rate time series, *Physiological Measurement* **29**(1) (2008), 81–94.

[30] E. Mandani and S. Assilian, An experiment in linguistic synthesis with a fuzzy logic controller, *Int J Man Mach Stud* **7**(1) (1975), 1–13.

[31] K. Meng, Z.Y. Dong and K.P. Wong, Self-adaptive radial basis function neural network for short-term electricity price forecasting, *IET Generation, Transmission and Distribution* **3**(4) (1975), 325–335.

[32] F. Moller, A scaled conjugate gradient algorithm for fast supervised learning, *Neural Networks* **6** (1990), 525–533.

[33] S. Moshiri, N.E. Cameron and D. Scuse, Static, Dynamic, and Hybrid Neural Networks in Forecasting Inflation, *Computational Economics* **14**(3) (1999), 219–235.

[34] J. Park and I. Sandberg, Universal approximation using radial-basis function networks, *Neural Comput* **3** (1991), 246–257.

[35] W. Pedrycz, Conditional fuzzy clustering in the design of radial basis function neural networks, *IEEE Transactions on Neural Networks* **9**(4) (1998), 601–612.

[36] M.D. Pérez-Godoy, A.J. Rivera, M.J. Jesús and I. Rojas, Co-EvRBFN: an approach to solving the classification problem with a hybrid cooperative-coevolutive algorithm, *Proceedings of the International Workshop on Artificial Neural Networks*, (2007) pp. 324–332.

[37] R. Pino, J. Parreno, A. Gomez and P. Priore, Forecasting next-day price of electricity in the Spanish energy market using artificial neural networks, *Engineering Applications of Artificial Intelligence* **21**(1) (2008), 53–62.

[38] J. Plat, A Resource Allocating Network for Funcion Interpolation, *Neural Computation* **3** (1991), 213–225.

[39] M. Potter and K. De Jong, Cooperative Coevolution: an architecture for evolving coadapted subcomponents, *Evolutionary Compututation* **8**(1) (2000), 1–29.

[40] L. Sánchez and I. Couso, Fuzzy Random Variables-Based Modeling with GA-P Algorithms, in: *Information, Uncertainty and Fusion*, B. Bouchon, R.R. Yager and L. Zadeh, eds, 2000, pp. 245–256.

[41] V. Rivas, J.J. Merelo, P. Castillo, M.G. Arenas and J.G. Castellano, Evolving RBF neural networks for time-series forecasting with EvRBF, *Information Science* **165** (2004), 207–220.

[42] A.J. Rivera, I. Rojas, J. Ortega and M.J. del Jess, A new hybrid methodology for cooperative-coevolutionary optimization of radial basis function networks, *Soft Computing*, 2007, doi: 10.1007/s00500-006-0128-9.

[43] A.F. Sheta and K. De Jong, Time-series forecasting using GA-tuned radial basis functions, *Information Sciencie* **133** (2001), 221–228.

[44] M. Ture and I. Kurt, Comparison of four different time series methods to forecast hepatitis A virus infection, *Expert Systems with Applications* **31**(1) (2006), 41–46.

[45] B. Whitehead and T. Choate, Cooperative-competitive genetic evolution of Radial Basis Function centers and widths for time series prediction, *IEEE Trans. on Neural Networks* **7**(4) (1996), 869–880.

[46] B. Widrow and M.A. Lehr, 30 Years of adaptive neural networks: perceptron, madaline and backpropagation, *Proceedings of the IEEE* **78**(9) (1990), 1415–1442.

[47] L. Yingwei, N. Sundarajan and P. Saratchandran, A sequential learning scheme for function approximation using minimal radial basis function neural network, *Neural Computation* **9** (1997), 361–478.

[48] T. Yu and D. Wilkinson, A co-evolutionary fuzzy system for reservoir well logs interpretation, *Evolutionary computation in practice* (2008), 199–218.

[49] Z. Yun, Z. Quan, S. Caixin, L. Shaolan, L. Yuming and S. Yang, (2008). RBF neural network and ANFIS-based short-term load forecasting approach in real-time price environment, *IEEE Transactions on Power Systems* **23**(3) (2008), 853–858.