Neurocomputing 128 (2014) 433–446

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Short, medium and long term forecasting of time series using the L-Co-R algorithm

E. Parras-Gutierrez [a], V.M. Rivas [a,*], M. Garcia-Arenas [b], M.J. del Jesus [a]

[a] Department of Computer Sciences, Campus Las Lagunillas s/n, 23071 Jaen, Spain
[b] Department of Computers, Architecture and Technology, C/ Periodista Daniel Saucedo s/n, 18071 Granada, Spain

ARTICLE INFO

Article history:
Received 21 February 2013
Received in revised form
12 June 2013
Accepted 26 August 2013
Communicated by P. Zhang
Available online 31 October 2013

Keywords:
Neural networks
Coevolutionary algorithms
Time series forecasting
Significant lags
Variable term horizon

ABSTRACT

This paper describes the coevolutionary algorithm L-Co-R (Lags COevolving with Radial Basis Function Neural Networks – RBFNs), and analyzes its performance in the forecasting of time series in the short, medium and long terms. The method allows the coevolution, in a single process, of the RBFNs as the time series models, as well as the set of lags to be used for predictions, integrating two genetic algorithms with real and binary codification, respectively. The individuals of one population are radial basis neural networks (used as models), while sets of candidate lags are individuals of the second population. In order to test the behavior of the algorithm in a new context of a variable horizon, 5 different measures have been analyzed, for more than 30 different databases, comparing this algorithm against six existing algorithms and for seven different prediction horizons. Statistical analysis of the results shows that L-Co-R outperforms other methods, regardless of the horizon, and is capable of predicting short, medium or long horizons using real known values.

1. Introduction

A time series can be defined as a chronological sequence of observed data from any periodical task or behavior, or activity in fields like Engineering, Biology, Economy, or Social Sciences, among many others [1]. Therefore, the task of predicting values of the series based on past and present values in order to achieve the information of the underlying model can be understood under the concept of time series forecasting.

Dealing with time series forecasting implies considering three important aspects, where the first one is the choice of the time periods (or lags) that must be used in order to forecast values. This way, the selection of these lags to be used as input variables for the model turns into a problem that can be dealt with data mining techniques. The second aspect to take into account is the trend, i.e., whether the time series tends to grow or decrease considering a long period of time. Finally, the prediction period must be considered. Usually, there exists a tendency to forecast using short horizons due to the difficulty of utilizing longer periods and, therefore, the results of the former tend to be more reliable.

There exist a wide number of techniques that have been developed to model and forecast time series. These techniques can be coarsely grouped into descriptive traditional technologies, linear and nonlinear modern models, and technologies coming from the soft computing area. Among all of these technologies, AutoRegressive Integrated Moving Average (ARIMA), by Box and Jenkins [1], is probably the most well known and widely used method. The method combines autoregressive and moving average terms into an equation in order to build a linear model to forecast new values. The autoregressive part of the equation relates the future value to the past and present ones, while the moving average component relates the future value to the errors of previous forecasting. Nevertheless, the models provided by the ARIMA method are simplistic linear models, unable to find complex subtle patterns in the time series data.

There also exist diverse techniques in the soft computing area developed to tackle time series forecasting, such as Artificial Neural Networks (ANNs), evolutionary algorithms, fuzzy logic or expert systems. The learning and generalization capabilities of ANNs have shown, by means of many successful applications, that they are a suitable alternative tool for both forecasting researchers and practitioners.

In the work presented here, the coevolutionary algorithm L-Co-R (Lags COevolving with RBFNs) [2] has been utilized which makes use of Radial Basis Function Networks (RBFNs) and Evolutionary Algorithms (EAs). RBFNs were described by Broomhead and Lowe [3] as feedforward neural networks, composed of a single hidden layer, whose neurons use radial basis functions as activation functions. The objective is to obtain neural networks capable of

* Corresponding author. Tel.: +34 953212344; fax: +34 953212472.
E-mail addresses: eparrasg@vrivas.es (E. Parras-Gutierrez), vrivas@ujaen.es (V.M. Rivas), mgarenas@atc.ugr.es (M. Garcia-Arenas), mjjesus@ujaen.es (M.J. del Jesus).

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.08.023

modeling time series on one hand, and finding out the specific lags of the time series for predicting future values on the other hand. This double goal is carried out by a coevolutionary process that divides the main problem into two subproblems which depend on each other. This way, one population evolves sets of time series lags to forecast future values, and the second population evolves a set of RBFNs to obtain an appropriate design for the aforementioned. The last one determines the architecture of the net and parameters like number of layers, connection between neurons, weights and radii for neurons, among others.

Thus, it employs a collaboration process in which the individuals of one population can cooperate with individuals from the other population to obtain better solutions.

Apart from minimizing the error obtained when predicting time series with short horizons as was used in [2], L-Co-R is also developed to be used with variable horizons of prediction. Unlike most of the methods, L-Co-R does not forecast long horizons using one-step estimated values, L-Co-R is capable of predicting any horizon, short, medium or long, using real known values (and therefore, values without error propagation), that the algorithm itself establishes as the most important to use in the prediction.

In order to determine the effectiveness of the L-Co-R in the short, medium and long term, we have set seven diverse horizons to predict with, and thirty-four different time series. The results obtained have been compared with the results of other six algorithms found in the literature, and finally we have analyzed the results obtained with five quality measures.

The rest of the paper is organized as follows: Section 2 introduces some preliminary topics related to this research; Section 3 describes the L-Co-R method; Section 4 presents the experimentation carried out and the results obtained, and finally Section 5 presents some conclusions.

2. State of the art

2.1. Time series forecasting

In recent years time series forecasting has been a major field of research in the area of statistics [4] as well as operational research [5]. In the latest years numerous methods have emerged with the objective of modeling and/or forecasting time series by means of linear and nonlinear models. Linear methods have been widely used to model time series, and among them the exponential smoothing methods [6,7] stand out, simple exponential smoothing, Holt's linear methods, some variations of Holt–Winter's methods, and state space models [8]. However, the ARIMA methods [1], which are also linear methods, established a border line between traditional and modern methods. ARIMA methods integrate autoregressive and moving average models in a three-stage iterative cycle consisting of: identification of the time series, estimation of the parameters of the model, and verification of the model.

Nevertheless, these linear time series forecasting methods were insufficient in many real applications, leading to the development of nonlinear time series forecasting. Nonlinear models include regime-switching models, which comprise the wide variety of existing threshold autoregressive models [9] such as self-exciting models [10], smooth transition models [11], and continuous-time models [12]. Nevertheless, as pointed out by Clements [13], the main problems with the current nonlinear methods are the following: they usually develop very complex models; they do not perform in a robust way; and, they are difficult to use. De Gooijer and Hyndman [4] also conclude that future research on nonlinear models should include, among other considerations, the search for easy to use software.

On the other hand, time series forecasting has been faced with soft computing approaches, such as the ones reported by Samanta [14] and Zhu et al. [15], who developed methods based on cooperative particle swarm optimization. Studies like [16,17] proposed fuzzy time series models for forecasting, and Yu and Huarng [18] applied ANNs for training and forecasting in their fuzzy time series model. Models such as support vector regression [19] and fuzzy expert system [20] were proposed for electricity demand forecasting, among others.

ANNs have also been successfully applied to time series and recognized as an important tool for forecasting. The work of Tang [21] concluded that neural networks not only could provide better long-term forecasting but also did a better job than ARIMA models with a short series of input data. Furthermore, contrary to the traditional linear and nonlinear time series models, ANNs are nonlinear data-driven approaches with more flexibility and effectiveness in modeling for forecasting [22]. Jain and Kumar determined in their study [23] that the ANN models were able to produce more accurate forecasts than traditional models because they do not presuppose any functional form of the model to be developed and they do not depend on assumptions of linearity.

There exist numerous studies of different application areas where ANNs are used to forecast time series. The work of Arizmendi [24] obtained accurate predictions of the airborne pollen concentrations using ANNs. Zhang and Hu [22] employed ANNs, and Rivas et al. [25] RBFNs, for forecasting British pound and US dollar exchange rates. Bezerianos et al. [26] employed RBFNs for the assessment and prediction of heart rate variability.

Specifically to ANNs, the use of Radial Basis Functions (RBFs) as activation functions for them and their application to time series forecasting were first considered by Broomhead and Lowe in 1988 [3]. Afterwards, new studies reported by Carse and Fogarty [27], and Whitehead and Choate [28] focused on the prediction of time series.

In later studies, Harpham and Dawson [29] studied the effect of different basis functions on an RBFN for time series prediction. Moreover, Du [30] used time series with an encoding scheme for training RBFNs with genetic algorithms (GAs). Both the architecture (numbers and selections of nodes and inputs) and the parameters (centers and widths) of the RBFNs were represented in one chromosome and evolved simultaneously by GAs so that the selection of nodes and inputs could be achieved automatically.

Previous studies found in the literature can also be classified according to the prediction horizon (short-term, medium-term, and long-term). Generally, forecasting tends toward short-term prediction such as one-step-ahead prediction, since longer period prediction (medium-term or long-term) is more difficult, and sometimes may not be reliable because of the error propagation [31]. Thus, neural network models have been traditionally applied in short-term forecasting [32,33]. For instance, the study reported by Perez-Godoy [34] et al. applied a hybrid evolutionary cooperative–competitive algorithm for the application of RBFNs to the short-term and even medium-term forecasting of the extra-virgin olive oil price.

2.2. Lags selection in time series forecasting

As mentioned previously, another problem that emerges working with time series is the correct choice of the lags considered for representing the series. The relationship involving time series historical data defines a d-dimensional space where d is the minimum dimension capable of representing such a relationship. Takens' theorem [35] establishes that if d is sufficiently large it is possible to build a state space using the correct time lags and if this space is correctly rebuilt also guarantees that the dynamics of this space are topologically identical to the dynamics of the real systems state space.

Lag selection could also be seen as a special case of feature selection, as lags turn into the input variables to be given to the models. In the field of input selection, there exist studies like the one reported by Guillen et al. in [36] who applied co-evolution to feature selection, and RBFNs as the model to be built. Their method, in any case, was applied to function approximation and classification, but not to lag selection and time series forecasting. Co-evolution was also used by Stoean et al. [37] to deal with input variable selection in order to develop artificial neural networks, once more for classification problems. The same situation occurs in [38] in which a co-evolutionary paradigm is used in order to build a pool of solutions consisting of RBFNs and vectors of features. Individuals of both kinds are jointly used to solve classification problems.

In order to tackle the lags selection problem, an evolutionary method that performs a search for the minimum number of dimensions, Time-delay Added Evolutionary Forecasting, is presented in [39]. The methodology is inspired by Takens' theorem and consists of an iterative hybrid model composed of an ANN combined with a GA. In [40] the evolutionary selection of lags is divided into two stages: first, the optimal dimension of the reconstructed phase space is determined by the false nearest neighbors algorithm and then a near-optimal set of time lags is found with a GA for a fuzzy inference system.

There are some methods that carry out an automatic search for the relevant lags. The quantum-inspired evolutionary hybrid intelligent (QIEHI) algorithm [41], for instance, is an evolutionary hybrid intelligent method which is composed of an ANN and a modified evolutionary algorithm to search for the minimum dimension to determine the characteristic phase for time series. Another hybrid methodology composed of a modular morphological ANN with an evolutionary algorithm that searches for the best time lags is described in [42].

In [43] a study on the selection not only of the lags but also of the exogenous features with classical feature selection algorithms as a pre-processing stage is performed.

Lag selection is performed as a postprocessing stage in [44] with a sensitivity computation of the output to each time lag.

As can be observed, the approaches in the literature consider lags selection as a pre- or post-processing or as a part of the learning process but, instead of together, in hybrid processes with two or three stages. On the contrary, our goal is to address the selection of the lags which represent the series (with any type of correlation) jointly with the design process. For this reason, we consider coevolutionary algorithms a good mechanism for solving these problems.

2.3. Cooperative coevolution algorithms and time series forecasting

Cooperative coevolution, introduced by Potter and De Jong [45,46], consists of identifying the natural decomposition of a problem into subcomponents.

A population of individuals per subproblem is created and evolved by means of collaboration with individuals from other populations. There are many possible methods for choosing representatives with which to collaborate: random collaboration [47], best collaboration [45] (the most widely used in the methods of the literature), complete collaboration, and mixed collaboration [48]. Another important point is the collaboration credit assignment method, i.e., the way an individual is set a fitness when multiple collaborators are selected. There are three common methods: maximum, average, and minimum, although it has been proved to be significantly better using the maximum method than using minimum or average [47].

Cooperative coevolution has been employed for tasks like function optimization [49], multi-objective evolutionary optimization [50], instance selection [51], and feature selection [52], among others. Cooperative coevolution has also been used in order to train ANNs, such as the cooperative coevolutive approach for designing neural network ensembles [53] and RBFNs [54].

It is possible to find coevolution applied to forecasting tasks as in [55] where coevolution with the immune network, evolving the structure and parameters of the neural network, is applied to predicting the short-term load of a city in eastern China. The work reported by Qian-Li et al. [56] proposes a coevolutionary recurrent neural network for the multi-step-prediction of chaotic time series, estimating the proper parameters of phase space reconstruction and optimizing the structure of recurrent neural networks by coevolutionary strategy.

2.4. Quality measures for time series forecasting

Finally, in order to determine the accuracy of the forecast method applied to time series data, many measures have been proposed. Most textbooks recommended the use of the Mean Absolute Percentage Error (MAPE) [57] and this was the primary measure in the M-competition [58]. Other studies recommended other measures such as the Geometric Mean Relative Absolute Error (GMRAE), Median Relative Absolute Error (MdRAE), and Median Absolute Percentage Error (MdAPE) [59,60]. Later, the MdRAE, sMAPE (Symmetric Mean Absolute Percentage Error), and sMdAPE (Symmetric Median Absolute Percentage Error) were proposed [61]. Nevertheless, Hyndman and Koehler in their work [62] determined that all measures mentioned before were not generally applicable since they can be infinite or undefined and can produce misleading results. For this reason, they proposed a new measure suitable for all situations: the Mean Absolute Scaled Error (MASE), which was less sensitive to outliers, less variable on small samples, and more easily interpreted.

Among all of the different error measures that can be found in [62,4], those used in this work are MAE, MAPE, MdAPE, sMdAPE, and MAPE. Their equations are shown in Table 1 and are calculated according to the following definitions: Y_t is the observation at time $t = 1, \ldots, n$; F_t is the forecast of Y_t; e_t is the forecast error (i.e., $e_t = Y_t - F_t$); $p_t = 100e_t/Y_t$ is the percentage error, and finally q_t is determined as

$$q_t = \frac{e_t}{\frac{1}{n-1}\sum_{i=2}^{n} |Y_i - Y_{i-1}|}$$

Additionally, contingency tables have been computed for L-Co-R in order to show its behavior in directional error measures. The contingency table stores the number of times that increments and decrements in the original time series are correctly predicted by the model, as well as the time that they are incorrectly predicted. Afterwards, the χ^2 test is computed to compare the results against a pure random walk procedure.

Table 1
Used forecast accuracy measures.

Error measures		
MAE	Mean Absolute Error	$mean(\lvert e_t \rvert)$
MAPE	Mean Absolute Percentage Error	$mean(\lvert p_t \rvert)$
MdAPE	Median Absolute Percentage Error	$median(\lvert p_t \rvert)$
sMdAPE	Symmetric Median Absolute Percentage Error	$median(200\lvert Y_t - F_t \rvert (Y_t + F_t))$
MASE	Mean Absolute Scaled Error	$mean(\lvert q_t \rvert)$

3. L-Co-R: Lags COevolving with Rbfns

L-Co-R, Lags COevolving with Rbfns [2], is an algorithm which designs RBFNs for time series forecasting. It obtains an appropriate number of RBFs, a radius and a center for every RBF, the weights for the whole network, a suitable set of time lags, and in addition, it is able to remove the trend of the time series [63]. Our proposal solves the trend problem with an automatic data pre- and post-processing, and the learning of the rest by means of an EA. Since the main goal of the algorithm implies building at the same time both RBFNs and sets of significant lags that will be used to predict future values, L-Co-R is based on a coevolutionary approach. Thus, the main problem can be decomposed into two subproblems which depend on each other.

L-Co-R simultaneously evolves two populations of different individual species, in which any member of each population can cooperate with the best individual from the other one in every generation. Fig. 1 shows an example of individuals from population of RBFNs (subfigure a) and population of lags (subfigure b). Therefore, the new algorithm is composed of the following two populations:

- Population of RBFNs: a set of RBFNs evolves to design an appropriate architecture of the net. The population uses a real codification in which every individual represents a set of neurons (RBFs) that composes the network. The number of neurons is variable since it can increase or decrease during the evolutionary process. Every neuron (a in Fig. 1) is defined by a center (b) and a radius (c). The center (b) is a vector with the same dimension as the inputs. The exact dimension of the input space is given by an individual of the population of lags (the one chosen to evaluate the net). The radius (c) is calculated as half of the average distance between the centers of the network.
- Population of lags: sets of lags evolve in order to forecast future values of the time series. This population utilizes a binary codification scheme where each gene indicates whether the specific lag in the time series will be used to predict the values or not. The length of the chromosome is set at the beginning corresponding to the specific parameter, so that it cannot vary its size during the execution of the algorithm.

In both populations every individual is itself a possible solution to the subproblem.

The main goal of L-Co-R is to forecast any given time series for any horizon, reducing any hand made preprocessing step, and building suitable RBFNs designed with appropriate sets of lags, leading to an optimized quality measure.

3.1. General scheme

The algorithm follows the general scheme shown in Fig. 2.

The method performs a preliminary stage of preprocessing which removes the trend of the time series. Then, the L-Co-R algorithm creates the two initial populations and evaluates every individual of each population. Once the initial populations have been created, the coevolutionary process starts.

Firstly, the population of lags selects the individuals which are going to form part of the subpopulation. Genetic operators are applied with a cross generational elitist selection, heterogeneous recombination, and cataclysmic mutation (CHC) scheme [64] and the individuals are evaluated by choosing the collaborators from the population of RBFNs, assigning the result as fitness to the individual that was being evaluated. The worst individuals are deleted from the population and the best individuals remain for the next generation.

Secondly, the population of RBFNs begins to evolve when the population of lags has been evolved during a pre-specified number of generations. Then, the individuals of the subpopulation are selected, the operators are applied, and a collaborator from the population of lags is designated in order to establish the fitness of every individual.

Finally, at the end of the coevolutionary process, two models formed by a neural network and a set of lags are obtained. The first

```
Trend preprocessing
t = 0;
initialize P_lags(t);
initialize P_RBFNs(t);
evaluate individuals in P_lags(t);
evaluate individuals in P_RBFNs(t);
while termination condition not satisfied do
begin
  t = t+1;
  /* Evolve population of lags */
  for i=0 to max_gen_lags do
  begin
    set threshold;
    select P_lags'(t) from P_lags(t);
    apply genetic operators in P_lags'(t);
    /* Evaluate P_lags'(t) */
      choose collaborators from P_RBFNs(t);
      evaluate individuals in P_lags'(t);
    replace individuals from P_lags(t) with P_lags'(t);
    if threshold < 0
    begin
      diverge P_lags(t);
    end
  end
  /* Evolve population of RBFNs */
  for i=0 to max_gen_RBFNs do
  begin
    select P_RBFNs'(t) from P_RBFNs(t);
    apply genetic operators in P_RBFNs'(t);
    /* Evaluate P_RBFNs'(t) */
      choose collaborators from P_lags(t);
      evaluate individuals in P_RBFNs'(t);
    replace individuals from P_RBFNs(t) with P_RBFNs'(t);
  end
end
train models and select the best one
forecast test values with the final model
Trend postprocessing
```

Fig. 2. General scheme of method L-Co-R.

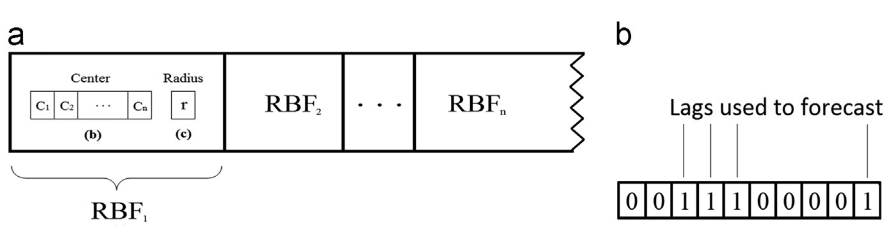

Fig. 1. Examples of codification of the populations. (a) RBFN individual and (b) lag individual.

model is composed of the best net and its best collaborator, and the second one is formed by the best set of lags and its best collaborator. Next, they are trained again and the one with the best fitness will be the final model. Then, the forecast values for the data test are obtained, and at this point, the postprocessing phase takes place so that the final test error can be computed.

L-Co-R has been implemented following a sequential scheme, so the two populations take turns evolving. During each generation only one of the two populations is active. Contrary to other algorithms, which at the end of the generation the population that was evolving communicates its best individual to the population that was waiting, in L-Co-R, the collaborator is given only when a member of population needs it. Each population selects a collaborator from the other population to assess its fitness.

L-Co-R is implemented to use a best collaboration scheme [45] and optimistic approach [47] for credit assignment. More precisely, for every individual in the first population the algorithm chooses the best collaborator of the other population. Exceptionally, at the beginning of the evolutionary process, since the population has not been evaluated, individuals are evaluated by a random collaborator.

Once every individual has selected its collaborator (the best one), the population asks for the collaborator to the other population. Thus, the communication is not produced at the end of a generation, but when a population asks for the specific collaborator it needs. On the other hand, the other population has been keeping the best representative in every generation. So, the individual who is going to be evaluated is coupled with the collaborator and the result obtained is set as its fitness. The fitness function is calculated using the following equation:

$$F = \frac{1}{\sqrt{\frac{1}{n}\sum_{t=0}^{n}(Y_t - F_t)^2}} \tag{1}$$

3.2. Evolutionary process

Both populations are randomly generated for the first generation. The population of RBFNs considers that every individual will have a number of neurons chosen at random which may not exceed a maximum number previously fixed only for this first generation. Subsequently, the number of neurons may be growing or shrinking as the algorithm evolves. The vector of weights is initialized to zero, the center is determined by choosing patterns from the training set at random, and the radius is estimated by calculating the half of the average distance from centers.

The population of lags takes into account that at least one gene of the chromosome must be set to one, since at least one input has to be given to the net to obtain the forecast value. The set of lags is evolved by means of the CHC [64] algorithm.

The populations incorporate evolutionary operators specifically designed to work with the individuals of every population. Thus, the operators have been designed to cover the search space in an effective way, maximizing the success probability.

The operators used by L-Co-R for every population are the following:

- Population of RBFNs:
  - Selection: this population implements tournament selection.
  - X_fix crossover operator: it replaces a sequence of neurons in the hidden layer of a network with an equal size sequence of neurons in the hidden layer of other network. This operator enables the sharing of information between the networks without affecting the hidden layer size.
  - Mutation: there are four operators to mutate the individuals. The choice of one of these mutation operators is carried out randomly, giving the deleter operator a double possibility of being selected.
    - C_random: the application of this operator can modify the point where each RBF of hidden neurons of the net is centered. The number of neurons affected is determined by an internal application factor. The operator performs an exploration of the solution space replacing the center of the neuron with a new random center. Each of the components of the new center is chosen following a uniform probability distribution in the range [min, max] determined from input patterns.
    - R_random: in the same way, this operator modifies the radius value of hidden neurons. The operator assigns a random value to the radius following an internal probability.
    - Adder: it adds new neurons to the hidden layer. The values for the center and radius vectors of a new neuron are randomly set, within the range for each dimension of input space.
    - Deleter: this operator does the opposite of the adder operator, it deletes neurons from the hidden layer. The exact number of neurons varies from one net to another, since the operator is applied to each neuron with a probability.
      The deleter operator has a twofold objective. The first one is to reduce the complexity of the network without losing their ability to approximate the training data set. The second one is to prevent overtraining networks, since a high capacity of generalization is desirable.
  - Replacement: the new individuals and the parent ones are joined in a unique population. Then, the worst individuals are eliminated keeping the best ones until the population reaches the original population size. Therefore, the best individuals remain in the next generation.
- Population of lags:
  - Selection: as the CHC algorithm establishes, the individuals are crossed verifying the incest prevention. After the cross, the individuals and their parents compete to survive with an elitism approach.
  - Crossover: the HUX crossover operator is used by this population for breeding. It guarantees that the two offsprings are always at the maximum Hamming distance from their parents.
  - Replacement: the population follows the same process of replacement as described previously. The new individuals and the parent ones are joined in an unique population. Then, the worst individuals are eliminated keeping the best ones until the population reaches the original population size. Therefore, the best individuals remain in the next generation.
  - Diverge: when the population is stagnated a restart is produced. The best individual is kept and the rest of the population is generated again in a random way.

4. Experimental study

This section describes the experiments carried out to test the behavior of L-Co-R predicting time series as the forecasting horizon grows. For this reason, a new context has been created in which L-Co-R has been applied to long prediction periods. The effectiveness of the algorithm has been compared with other methods, and a statistical study is included.

The experimentation has been realized for seven different horizons using thirty-four public databases of examples which have different characteristics with respect to the number of data,

Table 2
Characteristics of time series used.

Time series	Data	Units	Period	Description
Accidents	240	Units	Monthly (Jan. 1979–Dec. 1998)	Number of accidents during a working day
AccDeath	216	Units	Monthly (Jan. 1990–Dec. 2007)	Number of road accident casualties
AccVictims	216	Units	Monthly (Jan. 1990–Dec. 2007)	Number of road accident casualties
Airline	144	Thousands	Monthly (Jan. 1949–Dec. 1960)	Airplane passengers of international flies
WmFrancfort	156	Index	Monthly (Jan. 1988–Dec. 2000)	Monthly values about market of Frankfort
WmLondon	156	Index	Monthly (Jan. 1988–Dec. 2000)	Monthly values about market of London
WmMadrid	156	Index	Monthly (Jan. 1988–Dec. 2000)	Monthly values about market of Madrid
WmMilan	156	Index	Monthly (Jan. 1988–Dec. 2000)	Monthly values about market of Milan
WmNewYork	156	Index	Monthly (Jan. 1988–Dec. 2000)	Monthly values about market of New York
WmParis	156	Index	Monthly (Jan. 1988–Dec. 2000)	Monthly values about market of Paris
WmTokyo	156	Index	Monthly (Jan. 1988–Dec. 2000)	Monthly values about market of Tokyo
Colgtems	276	Market quota	Weekly (Jan. 1958–Apr. 1963)	Market quota of Colgate toothpaste
Colgtepr	276	Price	Weekly (Jan. 1958–Apr. 1963)	Price of Colgate toothpaste
Crestms	276	Market quota	Weekly (Jan. 1958–Apr. 1963)	Market quota of Crest toothpaste
Crestpr	276	Price	Weekly (Jan. 1958–Apr. 1963)	Price of Crest toothpaste
Deceases	228	Units	Monthly (Jan. 1980–Dec. 1998)	Number of monthly deceases
Spectators	233	Thousands	Monthly (Jan. 1990–May 2009)	Number of thousand spectators who were in the cinema
SpaMovSpec	233	Thousands	Monthly (Jan. 1990–May 2009)	Spectators who watched a Spanish movie in the cinema
ForMovSpec	233	Thousands	Monthly (Jan. 1990–May 2009)	Spectators who watched a foreign movie in the cinema
Exchange	208	Price	Weekly (Dec. 1979–Dec. 1983)	Exchange rates between British Pound and US Dollar
Gasoline	618	Thousands	Weekly (Jul. 1993–May 2005)	Finished motor gasoline production (thousand barrels)
MortCanc	43	Units	Monthly (Jan. 2006–Jul. 2009)	Number of canceled mortgages
MortMade	79	Units	Monthly (Jan. 2003–Jul. 2009)	Number of made mortgages
Books	132	Thousands	Monthly (Jan. 1998–Dec. 2008)	Editorial production of books
Motorcycles	234	Units	Monthly (Jan. 1990–Jun. 2009)	Manufacture of motorcycles
Unemployed	164	Units	Monthly (Jan. 1996–Aug. 2009)	Number of Spanish unemployed people
FreeHouPrize	58	Euros	Quarterly (Q1 1995–Q2 2009)	Price per m^2 of private housing
Prisoners	235	Units	Monthly (Jan. 1990–Jul. 2009)	Number of prisoners
Takings	233	Euros	Monthly (Jan. 1990–mayo 2009)	Average spending per spectator
TurIn	234	Thousands	Monthly (Jan. 1990–Jun. 2009)	Internal air traffic
TurOut	234	Thousands	Monthly (Jan. 1990–Jun. 2009)	External air traffic
TUrban	164	Thousands	Monthly (Jan. 1996–Aug. 2009)	Number of passengers transported by urban transport
Cars	236	Units	Monthly (Jan. 1990–Aug. 2009)	Vehicle manufacture (cars)
HouseFin	211	Units	Monthly (Jan. 1992–Jul. 2009)	Number of finished houses

period of time and topic they represent. Most data bases have been extracted from the Spanish National Statistics Institute.[1] A brief description of every one is given in Table 2.

The time series can be accessed at https://sites.google.com/site/presetemp/datos. For the experimentation, we considered the first 75% of the observations to form the training data and the other 25% to test, for the thirty-four data sets.

On the other hand, the proposed method is compared with another six different methods found in the literature:

- EvRBF (Evolutionary Radial Basis Function Neural Networks) proposed by Rivas et al. [25].
- Fuzzy-WM (Fuzzy Rule Learning) by Wang and Mendel [65].
- NNEP (Neural Network Evolutionary Programming) proposed by Martinez-Estudillo et al. [66].
- PolCuadraticLMS (LMS Quadratic Regression) by Rustagi [67]
- RBFN by Broomhead and Lowe [3].
- ARIMA proposed by Box and Jenkins and better known as Box–Jenkins models [1].

Nevertheless, as described below, we have shown two different configurations for the NNEP and RBFN, shown in the tables of results with two additional columns. These methods, except ARIMA, are extracted from Keel [68]. Table 3 shows the specific parameter values employed by every method utilized.

The NNEP2 and RBFN2 columns derive from a specific adaptation of NNEP and RBFN methods, respectively. They are the result of a study of the complexity of the nets found by L-Co-R.

[1] National Statistics Institute (http://www.ine.es/).

Once the study was finished, the average number of neurons was used as a parameter for the algorithms. Then, NNEP and RBFN were equaled to L-Co-R having the same initial complexity of the networks, resulting in NNEP2 and RBFN2. Finally, NNEP2 and RBFN2 were executed with the nearest integer average number to the number of neurons obtained by L-Co-R with every data set.

In order to work with the eight methods mentioned before, the Estimated Partial Autocorrelation Function (EPAF) was used. It indicates which intervals of time from data sets are considered more important to be taken into account when patterns of data are going to be formed. One of the main advantages of L-Co-R is that it is not necessary to apply any a priori preprocessing in this sense, since the algorithm is able to automatically find the most suitable lags during the evolution of the algorithm by itself. So, a previous study of the significant lags was performed to test every method used.

As shown in Fig. 2, L-Co-R applies a preprocessing step, in which the trend is removed, and a postprocessing one to compute the real, trended outputs. In order to compare the remaining algorithms in the same conditions, the rest of algorithms were given the data once preprocessed without trend and the post-processing phase was also carried out.

According to the stochastic nature of L-Co-R, any of the 5 quality measures considered have been estimated as executing 30 times in any experiment, using the same training and test sets per database in any execution.

Due to space limitations, this section shows the results for the following horizons: 2, 4, 10, 20, and 50. For the same reason, only two (MAPE and MASE) out of the five considered error measures considered are commented on this section. These two measures are the ones in which L-Co-R yields the best and worst results, respectively. Nevertheless, the whole set of results can be accessed at http://simidat.ujaen.es/neurocomputing2013, including the

Table 3
Parameters used by the different methods.

Method	Parameter	Value	Method	Parameter	Value
L-Co-R	PopSizeLag	50	EvRBF	Population size	100
	MaxGenerationLag	5		Generations	10
	MaxLongCrom	10%		Validation rate	0.25
	PopSizeRbfn	50		Neurons rate	0.1
	MaxGenerationsRbfn	10		Tournament size	30
	ValidationRate	0.25		Replacement rate	0.75
	NeuronsRate	0.05		Crossover rate	0.9
	TournamentSize	3		Mutator rate	0.1
	ReplacementRate	0.5	Fuzzy-WM	Number of labels	5
	XOverRate	0.8		KB Output File Format with	
	MutatorRate	0.2		Weight values to 1?	0
	MaxGenerations	20	NNEP2	Number of neurons in hidden layer	Depends on the data set
NNEP	Number of neurons in hidden layer	4			
	Transfer function in each neuron	Product_Unit		Transfer function in each neuron	Product_Unit
	Number of generations	1000		Number of generations	1000
PolCuadraticLMS	–	–	RBFN2	Number of hidden neurons	Depends on the data set
RBFN	Number of hidden neurons	50			

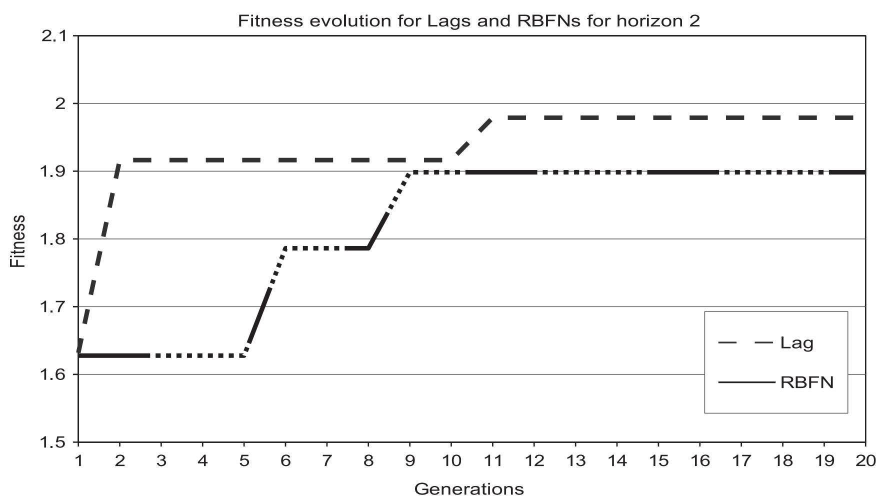

Fig. 3. The evolution of the fitness from an execution for horizon 2.

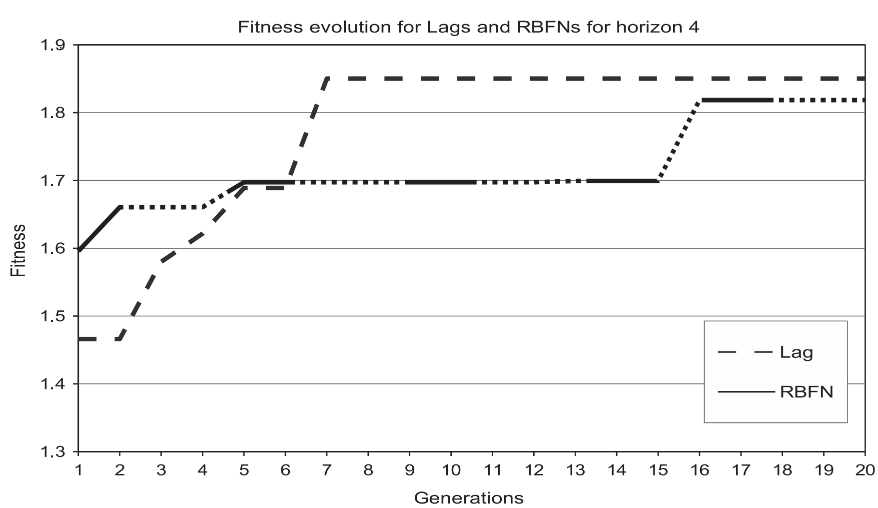

Fig. 4. The evolution of the fitness from an execution for horizon 4.

seven horizons, the 5 error measures, and a comparison between lags selected by the Estimated Partial Autocorrelation Function and the ones selected by L-Co-R.

The co-evolutionary approach followed by L-Co-R is able to evolve both neural nets and lag populations. Figs. 3–7 show how fitness increases for these two populations throughout the

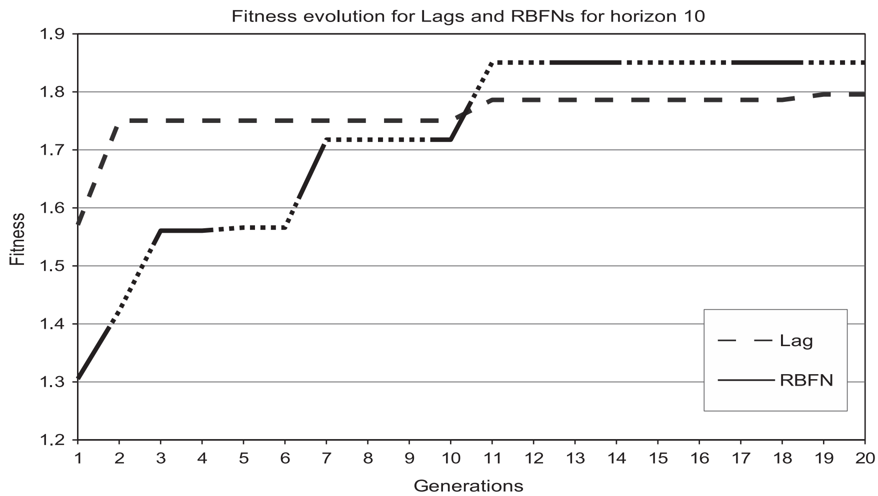

Fig. 5. The evolution of the fitness from an execution for horizon 10.

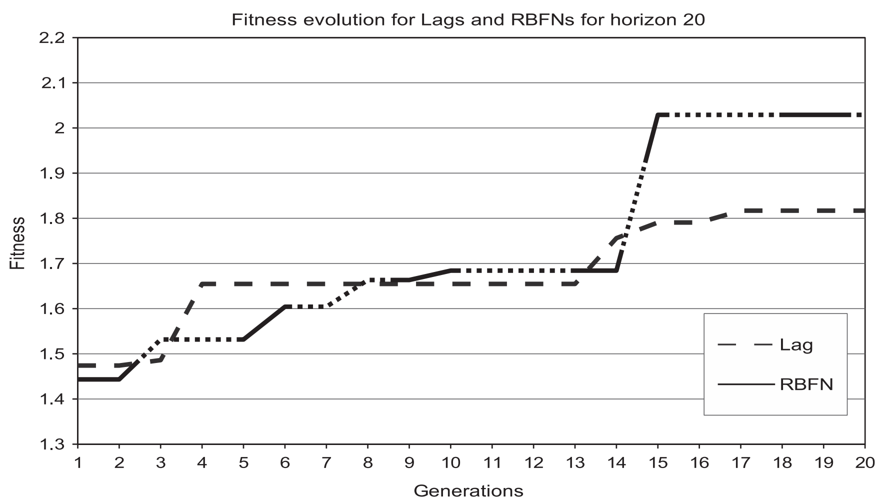

Fig. 6. The evolution of the fitness from an execution for horizon 20.

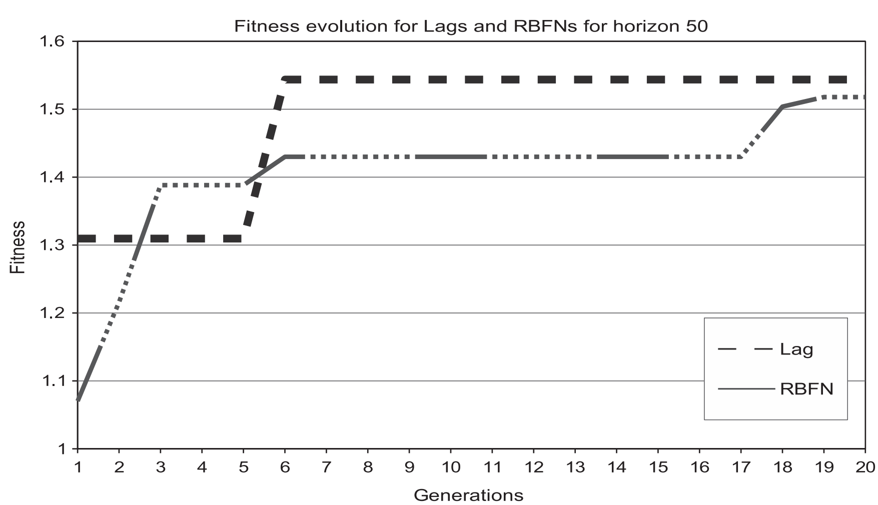

Fig. 7. The evolution of the fitness from an execution for horizon 50.

execution of the algorithm. Randomly chosen executions have been drawn, one for each of the considered horizons.

Figs. 8 and 9 graphically show the results of the methods for MAPE and MASE, respectively. Every figure represents the number of data bases (expressed as a percentage) in which every algorithm obtains the best result, distinguishing each horizon individually.[2] As can be observed, L-Co-R achieves the best percentage with both measures and with respect to all horizons.

[2] The results for the other horizons and quality measures can be found in http://simidat.ujaen.es/neurocomputing2013, Figs. 1–5.

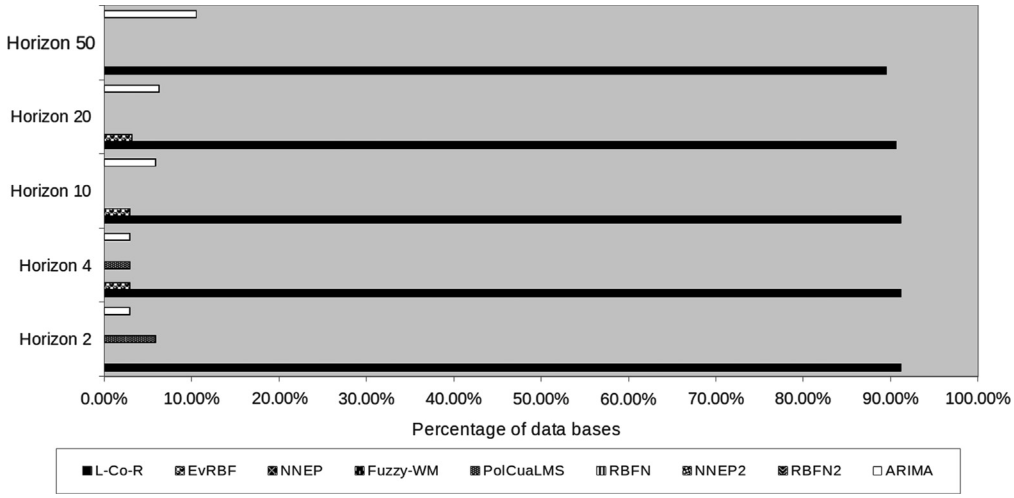

Fig. 8. Percentage of time series datasets in which every algorithm obtains better results than the other methods with respect to MAPE, for horizons 2, 4, 10, 20, and 50. The algorithms with value 0 are not represented in the graphic.

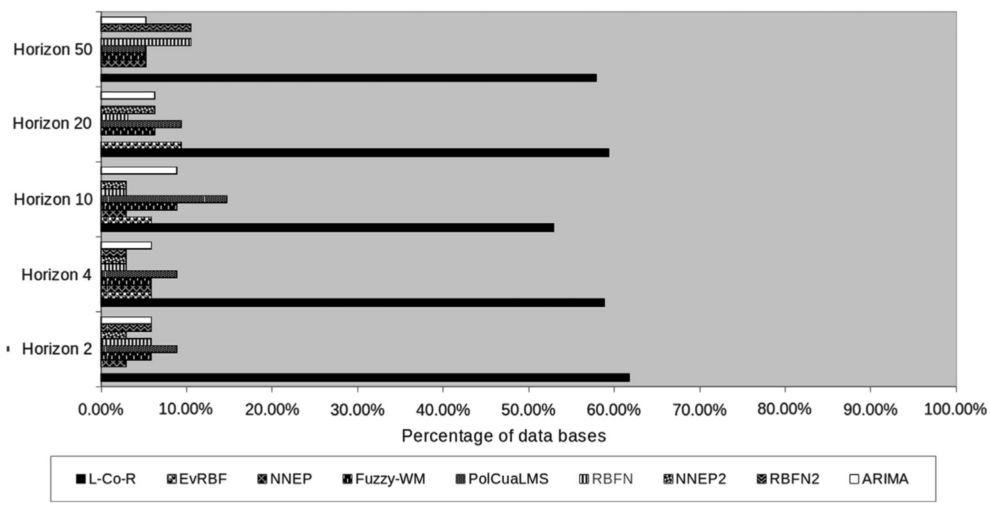

Fig. 9. Graphic of the number of time series (expressed in %) in which every algorithm obtains better results than the other methods with respect to MASE, for horizons 2, 4, 10, 20, and 50.

Table 4
Percentage of databases per horizon in which L-Co-R behaves better than pure random walk. This table summarizes the process of building the contingency tables, computing the *p-values* returned by the χ^2 test, and counting those databases for which the *p-value* turns out to be smaller than 0.05.

Horizon	2	4	10	20	50
Percentage (%)	47.60	70.59	61.76	52.94	57.89

More precisely, L-Co-R gets the best result in more than 90% of time series predicting with horizon 2, 4, 10, and 20, and more than 89% with horizon 50, respecting the quality measure MAPE. With regard to MASE, L-Co-R obtains better results than the other methods in more than 52% of data bases, using any horizon considered. In both graphics, methods yielding a percentage equal to 0 have been removed.

In order to use a directional prediction error for L-Co-R, contingency tables and their respective χ^2 test have been computed for all databases and the horizons taken into account. Table 4 shows the percentage of databases in which the *p-value* returned by the χ^2 test is smaller than 0.05, i.e., L-Co-R obtains significantly better than a pure random walk procedure.

4.1. Statistical study and conclusions of the experimentation

A statistical study was performed in order to check if the differences between methods are significant for each horizon and quality measure considered.[3] The procedure carried out was the following:

1. First, we tested if is possible to use parametric statistical techniques over the sample of results: to do this we checked the three necessary conditions: independency, the normality and homoscedasticity [69,70]. With respect to the normality condition, we applied the Shapiro–Wilk test as used in the study reported by García et al. [71]. This test confirmed that the condition was not fulfilled, and that therefore a non-parametric test has to be used.
2. Friedman and Iman-Davenport tests were applied in order to study whether significant differences exist between all methods. For all horizons, the statistics of Friedman and Iman-Davenport were clearly greater than their associated critical values, so it can be concluded that there are significant differences among the observed

[3] The results concerning the 5 quality measures are available at http://simidat.ujaen.es/neurocomputing2013, Tables 2–36.

Table 5
Average rankings of the algorithms (Friedman) for horizon 2. The algorithm with the lowest value is taken as the control one.

MAE		MAPE		MASE		MdAPE		sMdAPE	
Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking
L-Co-R	**1.647**	**L-Co-R**	**1.441**	**L-Co-R**	**2.412**	**L-Co-R**	**1.265**	**L-Co-R**	**1.918**
ARIMA	3.411	ARIMA	3.029	RBFN	4.235	ARIMA	3.450	ARIMA	3.353
RBFN2	4.264	RBFN	4.412	RBFN2	4.294	RBFN2	4.912	EvRBF	5.147
RBFN	4.382	RBFN2	4.412	PolCua.	4.559	RBFN	5.088	RBFN2	5.412
Fuzzy-WM	5.118	PolCua.	5.412	ARIMA	4.823	NNEP	5.235	RBFN	5.418
PolCua.	5.147	Fuzzy-WM	5.618	Fuzzy-WM	4.882	NNEP2	6.000	NNEP	5.441
NNEP	6.294	NNEP	6.382	NNEP	5.941	Fuzzy-WM	6.147	NNEP2	5.765
NNEP2	6.794	NNEP2	6.676	NNEP2	6.382	PolCua.	6.294	Fuzzy-WM	6.206
EvRBF	7.941	EvRBF	7.618	EvRBF	7.471	EvRBF	6.559	PolCua.	6.353

Table 6
Average rankings of the algorithms (Friedman) for horizon 4.

MAE		MAPE		MASE		MdAPE		sMdAPE	
Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking
L-Co-R	**1.529**	**L-Co-R**	**1.294**	**L-Co-R**	**2.441**	**L-Co-R**	**1.088**	**L-Co-R**	**1.853**
ARIMA	3.500	ARIMA	3.147	PolCua.	4.265	ARIMA	3.235	ARIMA	3.206
RBFN	4.059	RBFN	4.147	RBFN	4.294	RBFN	4.912	EvRBF	5.029
RBFN2	4.471	RBFN2	4.618	ARIMA	4.824	RBFN2	5.353	NNEP	5.353
PolCua.	5.029	PolCua.	5.471	RBFN2	4.882	NNEP	5.647	RBFN	5.412
Fuzzy-WM	5.059	Fuzzy-WM	5.529	Fuzzy-WM	4.912	Fuzzy-WM	5.941	NNEP2	5.618
NNEP	6.647	NNEP	6.675	NNEP	6.059	NNEP2	6.176	RBFN2	5.765
NNEP2	6.971	NNEP2	6.912	NNEP2	6.118	PolCua.	6.265	Fuzzy-WM	6.176
EvRBF	7.735	EvRBF	7.118	EvRBF	7.206	EvRBF	6.382	PolCua.	6.588

Table 7
Average rankings of the algorithms (Friedman) for horizon 10.

MAE		MAPE		MASE		MdAPE		sMdAPE	
Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking
L-Co-R	**1.559**	**L-Co-R**	**1.206**	**L-Co-R**	**2.353**	**L-Co-R**	**1.029**	**L-Co-R**	**1.706**
ARIMA	3.382	ARIMA	2.918	Fuzzy-WM	4.559	ARIMA	3.265	ARIMA	3.206
RBFN2	4.382	RBFN	4.706	PolCua.	4.588	RBFN	5.235	EvRBF	4.941
RBFN	4.559	RBFN2	4.794	RBFN	4.647	RBFN2	5.235	NNEP	5.294
Fuzzy-WM	5.176	Fuzzy-WM	5.265	RBFN2	4.765	NNEP	5.382	RBFN2	5.529
PolCua.	5.265	PolCua.	5.559	ARIMA	4.765	NNEP2	5.971	NNEP2	5.618
NNEP	6.471	NNEP	6.471	NNEP2	5.824	PolCua.	6.088	RBFN	5.618
NNEP2	6.500	NNEP2	6.765	NNEP	6.265	Fuzzy-WM	6.353	PolCua.	6.412
EvRBF	7.706	EvRBF	7.324	EvRBF	7.235	EvRBF	6.441	Fuzzy-WM	6.441

results with a level of significance $\alpha \le 0.05$, in all cases. According to these results, a post hoc statistical analysis is needed. A ranking of the methods obtained from the Friedman test will determine the algorithm which achieves the best classification (that which has the lowest result compared with the other methods for all measures), so it will be taken as the control algorithm.

3. In order to find whether the control algorithm presents statistical differences with regard to the remaining methods in the comparison, we apply the Holm procedure [72], as is recommended in [71].

Tables 5–9 show the ranking of the methods obtained by the Friedman method. The best method for every error measure is stressed in bold at the top. As can be seen in these tables, the L-Co-R method achieves the best ranking with a result that is lower than the rest for all measures, and for every horizon, so it is taken as the control algorithm.

The results of the Holm procedure, to see whether the control algorithm presents statistical differences from the other algorithms, are shown in Tables 10–14 with regard to horizons 2, 4, 10, 20, and 50,[4] respectively. These tables present all the adjusted p-values for each comparison which involves the control algorithm, for MAPE and MASE,[5] respectively. The *p-value* is indicated in each comparison considering a level of significance $\alpha = 0.05$, and z (corresponding to column number 5 in the tables) as the statistic that compares the i-th and the j-th method. The z value is computed according to the following equation:

$$z = (R_i - R_j)/sqrt(M(M+1)/6N) \qquad (2)$$

where R_i is the value of the Friedman ranking for the algorithm i, R_j is the value of the Friedman ranking for the control algorithm. M is the total number of methods, and N is the number of datasets used for the comparative; thus, for this set of experiments, $M=9$ and $N=34$.

[4] Tables 44 and 47 from http://simidat.ujaen.es/neurocomputing2013 show the results for the rest horizons.

[5] The results of all quality measures can be seen at http://simidat.ujaen.es/neurocomputing2013, Tables 45, 46, 48–50.

Table 8
Average rankings of the algorithms (Friedman) for horizon 20.

MAE		MAPE		MASE		MdAPE		sMdAPE	
Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking
L-Co-R	**1.469**	**L-Co-R**	**1.250**	**L-Co-R**	**2.218**	**L-Co-R**	**1.063**	**L-Co-R**	**1.594**
ARIMA	2.906	ARIMA	2.781	RBFN2	4.531	ARIMA	3.156	ARIMA	3.063
RBFN2	4.469	RBFN2	4.531	ARIMA	4.563	RBFN	5.156	EvRBF	4.750
RBFN	4.531	RBFN	4.781	PolCua.	4.718	RBFN2	5.531	NNEP	5.436
Fuzzy-WM	5.406	Fuzzy-WM	5.438	RBFN	4.750	NNEP	5.531	RBFN	5.500
PolCua.	5.656	PolCua.	5.844	Fuzzy-WM	4.969	NNEP2	5.750	NNEP2	5.844
NNEP	6.375	NNEP	6.406	NNEP2	5.906	Fuzzy-WM	5.844	RBFN2	6.031
NNEP2	6.563	NNEP2	6.719	NNEP	6.188	EvRBF	6.313	Fuzzy-WM	6.156
EvRBF	7.625	EvRBF	7.250	EvRBF	7.156	PolCua.	6.656	PolCua.	6.625

Table 9
Average rankings of the algorithms (Friedman) for horizon 50.

MAE		MAPE		MASE		MdAPE		sMdAPE	
Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking
L-Co-R	**1.421**	**L-Co-R**	**1.263**	**L-Co-R**	**2.368**	**L-Co-R**	**1.000**	**L-Co-R**	**2.000**
ARIMA	3.158	ARIMA	2.474	PolCua.	3.737	ARIMA	3.053	ARIMA	3.105
RBFN	4.053	RBFN2	4.316	RBFN	4.263	RBFN	4.789	EvRBF	4.105
RBFN2	4.105	RBFN	4.474	RBFN2	4.474	RBFN2	5.263	RBFN	5.263
PolCua.	4.842	PolCua.	4.895	ARIMA	4.579	NNEP	5.526	NNEP	5.474
Fuzzy-WM	5.579	Fuzzy-WM	6.105	Fuzzy-WM	5.579	PolCua.	6.105	RBFN2	5.737
NNEP	6.526	NNEP	6.895	NNEP	6.053	NNEP2	6.211	NNEP2	5.789
NNEP2	7.105	EvRBF	7.105	NNEP2	7.053	EvRBF	6.368	PolCua.	6.421
EvRBF	8.211	NNEP2	7.474	EvRBF	7.895	Fuzzy-WM	6.684	Fuzzy-WM	7.105

Table 10
Results of Holm's procedure in every comparison between the control algorithm and the other algorithms for horizon 2. The column labeled as z establishes the ranking between the algorithms, α is equal to 0.05, and the hypothesis that the control algorithm does not yield results significantly better than algorithm i is rejected when $p<\alpha/i$.

Measure	**Alg$_{\text{Control}}$**	i	**Algorithm**	z	p	α/i	**Hypothesis**
MAPE	L-Co-R	8	EvRBF	9.299	1.418E-20	0.006	**Rejected**
		7	NNEP2	7.882	3.223E-15	0.007	**Rejected**
		6	NNEP	7.439	1.013E-13	0.008	**Rejected**
		5	Fuzzy-WM	6.288	3.219E-10	0.01	**Rejected**
		4	PolCuaLMS	5.978	2.260E-09	0.013	**Rejected**
		3	RBFN2	4.472	7.736E-06	0.017	**Rejected**
		2	RBFN	4.472	7.736E-06	0.025	**Rejected**
		1	ARIMA	2.391	1.680E-02	0.05	**Rejected**
MASE	L-Co-R	8	EvRBF	7.616	2.611E-14	0.006	**Rejected**
		7	NNEP2	5.978	2.260E-09	0.007	**Rejected**
		6	NNEP	5.314	1.074E-07	0.008	**Rejected**
		5	Fuzzy-WM	3.720	1.996E-04	0.01	**Rejected**
		4	ARIMA	3.631	2.823E-04	0.013	**Rejected**
		3	PolCuaLMS	3.232	1.227E-03	0.017	**Rejected**
		2	RBFN2	2.834	4.597E-03	0.025	**Rejected**
		1	RBFN	2.745	6.044E-03	0.05	**Rejected**

Table 11
Results of Holm's procedure in every comparison between the control algorithm and the other algorithms for horizon 4. The column labeled as z establishes the ranking between the algorithms, α is equal to 0.05, and the hypothesis that the control algorithm does not yield results significantly better than algorithm i is rejected when $p<\alpha/i$.

Measure	**Alg$_{\text{Control}}$**	i	**Algorithm**	z	p	α/i	**Hypothesis**
MAPE	L-Co-R	8	EvRBF	8.768	1.825E-18	0.006	**Rejected**
		7	NNEP2	8.458	2.729E-17	0.007	**Rejected**
		6	NNEP	8.236	1.777E-16	0.008	**Rejected**
		5	Fuzzy-WM	6.376	1.813E-10	0.01	**Rejected**
		4	PolCuaLMS	6.288	3.219E-10	0.013	**Rejected**
		3	RBFN2	5.004	5.623E-07	0.017	**Rejected**
		2	RBFN	4.295	1.745E-05	0.025	**Rejected**
		1	ARIMA	2.790	5.276E-03	0.05	**Rejected**
MASE	L-Co-R	8	EvRBF	7.173	7.311E-13	0.006	**Rejected**
		7	NNEP2	5.535	3.111E-08	0.007	**Rejected**
		6	NNEP	5.447	5.136E-08	0.008	**Rejected**
		5	Fuzzy-WM	3.720	1.996E-04	0.01	**Rejected**
		4	RBFN2	3.675	2.376E-04	0.013	**Rejected**
		3	ARIMA	3.587	3.348E-04	0.017	**Rejected**
		2	RBFN	2.790	5.276E-03	0.025	**Rejected**
		1	PolCuaLMS	2.745	6.044E-03	0.05	**Rejected**

The result of statistic z is used with the tables of normal distribution to obtain the p-value (Tables 10–14). The Holm procedure is used to compare each p-value against α/i, so that if p-value is less than α/i it is possible to assert that there exist significant differences as the hypothesis of equal means can be rejected.

As shown in Tables 10–14,[6] there are significant differences between L-Co-R and the remaining methods for all measures used. Therefore, L-Co-R really shows a better behavior with respect to test error compared to other methods regarding horizons 2, 4, 10, and 20. Even with the methods NNEP2 and RBFN2, in which the complexity of the initial networks is the same as in L-Co-R, it yielded better results with significant differences.

Regarding horizon 50, both MAPE and MASE error measures argue that L-Co-R cannot be considered better, significantly, than the ARIMA method. MASE also indicates that the results yielded by L-Co-R are not significantly better than the PolCuadraticLMS ones.

Finally, taking all the statistical studies carried out into account, we can conclude that L-Co-R has a good behavior in time series

[6] See also Tables 44 and 47 in http://simidat.ujaen.es/neurocomputing2013 for horizons 1 and 8.

Table 12
Results of Holm's procedure in every comparison between the control algorithm and the other algorithms for horizon 10. The column labeled as z establishes the ranking between the algorithms, α is equal to 0.05, and the hypothesis that the control algorithm does not yield results significantly better than algorithm i is rejected when $p < \alpha/i$.

Measure	$Alg_{Control}$	i	Algorithm	z	p	α/i	Hypothesis
MAPE	L-Co-R	8	EvRBF	9.210	3.249E-20	0.006	**Rejected**
		7	NNEP2	8.369	5.808E-17	0.007	**Rejected**
		6	NNEP	7.926	2.259E-15	0.008	**Rejected**
		5	PolCuaLMS	6.554	5.619E-11	0.01	**Rejected**
		4	Fuzzy-WM	6.111	9.917E-10	0.013	**Rejected**
		3	RBFN2	5.402	6.581E-08	0.017	**Rejected**
		2	RBFN	5.269	1.369E-07	0.025	**Rejected**
		1	ARIMA	2.568	1.022E-02	0.05	**Rejected**
MASE	L-Co-R	8	EvRBF	7.351	1.973E-13	0.006	**Rejected**
		7	NNEP	5.889	3.877E-09	0.007	**Rejected**
		6	NNEP2	5.225	1.740E-07	0.008	**Rejected**
		5	ARIMA	3.631	2.823E-04	0.01	**Rejected**
		4	RBFN2	3.631	2.823E-04	0.013	**Rejected**
		3	RBFN	3.454	5.525E-04	0.017	**Rejected**
		2	PolCuaLMS	3.365	7.645E-04	0.025	**Rejected**
		1	Fuzzy-WM	3.321	8.968E-04	0.05	**Rejected**

Table 13
Results of Holm's procedure in every comparison between the control algorithm and the other algorithms for horizon 20. The column labeled as z establishes the ranking between the algorithms, α is equal to 0.05, and the hypothesis that the control algorithm does not yield results significantly better than algorithm i is rejected when $p < \alpha/i$.

Measure	$Alg_{Control}$	i	Algorithm	z	p	α/i	Hypothesis
MAPE	L-Co-R	8	EvRBF	8.764	1.892E-18	0.006	**Rejected**
		7	NNEP2	7.988	1.376E-15	0.007	**Rejected**
		6	NNEP	7.531	5.028E-14	0.008	**Rejected**
		5	PolCuaLMS	6.710	1.952E-11	0.01	**Rejected**
		4	Fuzzy-WM	6.116	9.581E-10	0.013	**Rejected**
		3	RBFN	5.158	2.500E-07	0.017	**Rejected**
		2	RBFN2	4.793	1.647E-06	0.025	**Rejected**
		1	ARIMA	2.237	2.532E-02	0.05	**Rejected**
MASE	L-Co-R	8	EvRBF	7.212	5.527E-13	0.006	**Rejected**
		7	NNEP	5.797	6.762E-09	0.007	**Rejected**
		6	NNEP2	5.386	7.207E-08	0.008	**Rejected**
		5	Fuzzy-WM	4.017	5.904E-05	0.01	**Rejected**
		4	RBFN	3.697	2.181E-04	0.013	**Rejected**
		3	PolCuaLMS	3.651	2.607E-04	0.017	**Rejected**
		2	ARIMA	3.423	6.187E-04	0.025	**Rejected**
		1	RBFN2	3.378	7.312E-04	0.05	**Rejected**

forecasting with short, medium and long-term horizons, being better in most cases than the rest of the algorithms considered. L-Co-R stands out for its accuracy over a large set of sample data, which has different characteristics and nature.

5. Conclusions and future research

In this paper the effectiveness of the L-Co-R method, a coevolutionary algorithm for time series forecasting, for long-time forecasting and with a changing horizon environment is tested. Two different populations coevolve to obtain future values predictions whatever the given period: short, medium or long term. On one hand, a population of RBFNs evolves sets of neural networks in order to obtain an appropriate network architecture. On the other hand, a population of time lags evolves sets of important lags, which will be utilized to make future predictions. In order to implement the coevolution, both individuals of lag

Table 14
Results of Holm's procedure in every comparison between the control algorithm and the other algorithms for horizon 50. The column labeled as z establishes the ranking between the algorithms, α is equal to 0.05, and the hypothesis that the control algorithm does not yield results significantly better than algorithm i is rejected when $p < \alpha/i$.

Measure	$Alg_{Control}$	i	Algorithm	z	p	α/i	Hypothesis
MAPE	L-Co-R	8	NNEP2	6.990	2.754E-12	0.006	**Rejected**
		7	EvRBF	6.575	4.863E-11	0.007	**Rejected**
		6	NNEP	6.338	2.326E-10	0.008	**Rejected**
		5	Fuzzy-WM	5.450	5.048E-08	0.01	**Rejected**
		4	PolCuaLMS	4.087	4.366E-05	0.013	**Rejected**
		3	RBFN	3.613	3.023E-04	0.017	**Rejected**
		2	RBFN2	3.436	5.912E-04	0.025	**Rejected**
		1	ARIMA	1.362	1.731E-01	0.05	**Non reject.**
MASE	L-Co-R	8	EvRBF	6.220	4.982E-10	0.006	**Rejected**
		7	NNEP2	5.272	1.350E-07	0.007	**Rejected**
		6	NNEP	4.146	3.377E-05	0.008	**Rejected**
		5	Fuzzy-WM	3.613	3.023E-04	0.01	**Rejected**
		4	RBFN2	2.369	1.782E-02	0.013	**Rejected**
		3	RBFN	2.132	3.297E-02	0.017	**Rejected**
		2	PolCuaLMS	1.540	1.235E-01	0.025	**Non reject.**
		1	ARIMA	1.362	1.731E-01	0.05	**Non reject.**

population and individuals of RBFNs population can cooperate together to produce global solutions.

To test the performance of L-Co-R forecasting with a variable horizon, thirty-four different time series were used. The results of L-Co-R have been compared with another six methods found in the literature, regarding five different quality measures (MAPE, MASE, MAE, MdAPE, and sMdAPE) and for seven considered horizons (1, 2, 4, 8, 10, 20 and 50).

In order to draw conclusions from the results obtained, statistical study has been carried out. First of all, we used the Friedman and Iman-Davenport tests to see if the differences observed between methods are significant. Then we applied the Holm procedure in order to find out the control algorithm (L-Co-R in all cases) which presents statistical differences regarding the rest of the methods.

Thus, we can conclude that L-Co-R achieves better results than the other methods, taking into account the large set of time series and the context of variable horizon.

Acknowledgments

This work has been supported by the regional project TIC-3928 (Feder Founds), the Spanish project TIN 2012-33856 (Feder Founds), TIN 2011-28627-C02 (Feder Founds).

Appendix A. Supplementary data

Supplementary data associated with this article can be found in the online version at http://dx.doi.org.10.1016/j.neucom.2013.08.023.

References

[1] G. Box, G. Jenkins, Time Series Analysis: Forecasting and Control, Holden Day, San Francisco, 1976.
[2] E. Parras-Gutierrez, M. Garcia-Arenas, V. Rivas, M. del Jesus, Coevolution of lags and RBFNs for time series forecasting: L-Co-R algorithm, Soft Comput. 16 (6) (2012) 919–942.
[3] D. Broomhead, D. Lowe, Multivariable functional interpolation and adaptive networks, Complex Syst. 2 (1988) 321–355.
[4] J. De Gooijer, R. Hyndman, 25 years of time series forecasting, Int. J. Forecast. 22 (3) (2006) 443–473.

[5] R. Fildes, K. Nikolopoulos, S. Crone, A. Syntetos, Forecasting and operational research: a review, J. Oper. Res. Soc. 59 (2008) 1150–1172.
[6] R. Brown, Statistical Forecasting for Inventory Control, McGraw-Hill, 1959.
[7] P. Winters, Forecasting sales by exponentially weighted moving averages, Manage. Sci. 6 (3) (1960) 324–342.
[8] R. Snyder, Recursive estimation of dynamic linear models, J. R. Stat. Soc. Ser. B (Methodological) 47 (2) (1985) 272–276.
[9] H. Tong, On a threshold model, in: Pattern Recognition and Signal Processing, NATO ASI Ser. E: Appl. Sci. 29 (1978) 575–586.
[10] H. Tong, Threshold Models in Non-linear Time Series Analysis, Springer-Verlag, 1983.
[11] K. Chan, H. Tong, On estimating thresholds in autoregressive models, J. Time Ser. Anal. 7 (3) (1986) 179–190.
[12] P. Brockwell, R. Hyndman, On continuous-time threshold autoregression, Int. J. Forecast. 8 (2) (1992) 157–173.
[13] M. Clements, P. Franses, N. Swanson, Forecasting economic and financial time-series with non-linear models, Int. J. Forecast. 20 (2) (2004) 169–183.
[14] B. Samanta, Prediction of chaotic time series using computational intelligence, Expert Syst. Appl. 38 (9) (2011) 11406–11411.
[15] S. Zhu, J. Wang, W. Zhao, J. Wang, A seasonal hybrid procedure for electricity demand forecasting in China, Appl. Energy 88 (11) (2011) 3807–3815.
[16] W. Qiu, X. Liu, H. Li, A generalized method for forecasting based on fuzzy time series, Expert Syst. Appl. 38 (8) (2011) 10446–10453.
[17] C. Wang, A comparison study between fuzzy time series model and ARIMA model for forecasting Taiwan export, Expert Syst. Appl. 38 (8) (2011) 9296–9304.
[18] T. Yu, K. Huarng, A neural network-based fuzzy time series model to improve forecasting, Expert Syst. Appl. 37 (4) (2010) 3366–3372.
[19] K. Kavaklioglu, Modeling and prediction of Turkey's electricity consumption using support vector regression, Appl. Energy 88 (1) (2011) 368–375.
[20] P.K. Dash, A.C. Liew, S. Rahman, G. Ramakrishna, Building a fuzzy expert system for electric load forecasting using a hybrid neural network, Expert Syst. Appl. 9 (3) (1995) 407–421.
[21] Z. Tang, C. de Almeida, P. Fishwick, Time series forecasting using neural networks vs. Box–Jenkins methodology, Simulation 57 (5) (1991) 303–310.
[22] G. Zhang, B. Patuwo, M. Hu, Forecasting with artificial neural networks: the state of the art, Int. J. Forecast. 14 (1) (1998) 35–62.
[23] A. Jain, A. Kumar, Hybrid neural network models for hydrologic time series forecasting, Appl. Soft Comput. 7 (2) (2007) 585–592.
[24] C.M. Arizmendi, J. Sanchez, N.E. Ramos, G.I. Ramos, Time series predictions with neural nets: application to airborne pollen forecasting, Int. J. Biometeorol. 37 (3) (1993) 139–144.
[25] V. Rivas, J. Merelo, P. Castillo, M. Arenas, J. Castellano, Evolving RBF neural networks for time-series forecasting with EvRBF, Inf. Sci. 165 (3–4) (2004) 207–220.
[26] A. Bezerianos, S. Papadimitriou, D. Alexopoulos, Radial basis function neural networks for the characterization of heart rate variability dynamics, Artif. Intell. Med. 15 (3) (1999) 215–234.
[27] B. Carse, T. Fogarty, Fast evolutionary learning of minimal radial basis function neural networks using a genetic algorithm, in: Proceedings of Evolutionary Computing, Lecture Notes in Computer Science, vol. 1143, Springer Berlin, Heidelberg, 1996, pp. 1–22.
[28] B. Whitehead, T. Choate, Cooperative–competitive genetic evolution of radial basis function centers and widths for time series prediction, IEEE Trans. Neural Networks 7 (4) (1996) 869–880.
[29] C. Harpham, C. Dawson, The effect of different basis functions on a radial basis function network for time series prediction: a comparative study, Neurocomputing 69 (16–18) (2006) 2161–2170.
[30] H. Du, N. Zhang, Time series prediction using evolving radial basis function networks with new encoding scheme, Neurocomputing 71 (7–9) (2008) 1388–1400.
[31] A. Chatterjee, P. Siarry, Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization, Comput. Oper. Res. 33 (3) (2006) 859–871.
[32] H. Hippert, J. Taylor, An evaluation of Bayesian techniques for controlling model complexity and selecting inputs in a neural network for short-term load forecasting, Neural Networks 23 (3) (2010) 386–395.
[33] C. Lee, C. Ko, Time series prediction using RBF neural networks with a nonlinear time-varying evolution PSO algorithm, Neurocomputing 73 (1–3) (2009) 449–460.
[34] M. Perez-Godoy, P. Pérez-Recuerda, M. Frías, A. Rivera, C. Carmona, M. Parras, Co2rbfn for short and medium term forecasting of the extra-virgin olive oil price, in: J. González, D. Pelta, C. Cruz, G. Terrazas, N. Krasnogor (Eds.), Proceedings of Nature Inspired Cooperative Strategies for Optimization, vol. 284, Springer, Berlin, Heidelberg, 2010, pp. 113–125.
[35] F. Takens, Detecting strange attractor in turbulence, in: Dynamical Systems and Turbulence, Lecture Notes in Mathematics, vol. 898, Springer, New York, NY, 1980, pp. 366–381.
[36] A. Guillén, H. Pomares, J. González, I. Rojas, O. Valenzuela, B. Prieto, Parallel multiobjective memetic {RBFNNs} design and feature selection for function approximation problems, Neurocomputing 72 (16–18) (2009) 3541–3555, http://dx.doi.org/10.1016/j.neucom.2008.12.037, Financial Engineering, Computational and Ambient Intelligence (IWANN 2007) 〈http://www.sciencedirect.com/science/article/pii/S0925231209001970〉.
[37] C. Stoean, R. Stoean, M. Lupsor, H. Stefanescu, R. Badea, Feature selection for a cooperative coevolutionary classifier in liver fibrosis diagnosis, Comput. Biol. Med. 41 (4) (2011) 238–246, http://dx.doi.org/10.1016/j.compbiomed.2011.02.006 〈http://www.sciencedirect.com/science/article/pii/S0010482511000308〉.
[38] J. Tian, M. Li, F. Chen, Dual-population based coevolutionary algorithm for designing {RBFNN} with feature selection, Expert Syst. Appl. 37 (10) (2010) 6904–6918, http://dx.doi.org/10.1016/j.eswa.2010.03.031 〈http://www.sciencedirect.com/science/article/pii/S0957417410002125〉.
[39] T. Ferreira, G. Vasconcelos, P. Adeodato, A new intelligent system methodology for time series forecasting with artificial neural networks, Neural Process. Lett. 28 (2) (2008) 113–129.
[40] K. Lukoseviciute, M. Ragulskis, Evolutionary algorithms for the selection of time lags for time series forecasting by fuzzy inference systems, Neurocomputing 73 (10–12) (2010) 2077–2088.
[41] R. Araújo, A quantum-inspired evolutionary hybrid intelligent approach for stock market prediction, Int. J. Intell. Comput. Cybern. 3 (10) (2010) 24–54.
[42] R. Araújo, Hybrid intelligent methodology to design translation invariant morphological operators for Brazilian stock market prediction, Neural Networks 23 (10) (2010) 1238–1251.
[43] R. García-Pajares, J. Benitez, G. Sainz Palmero, Feature selection form time series forecasting: a case study, in: Proceedings of 8th International Conference on Hybrid Intelligent Systems, 2008, pp. 555–560.
[44] A. Maus, J.C. Sprott, Neural network method for determining embedding dimension of a time series, Commun. Nonlinear Sci. Numer. Simul. 16 (8) (2011) 3294–3302.
[45] M. Potter, K. De Jong, A cooperative coevolutionary approach to function optimization, in: Proceedings of Parallel Problem Solving from Nature, Lecture Notes in Computer Science, vol. 866, Springer, Berlin, Heidelberg, 1994, pp. 249–257.
[46] M. Potter, K. De Jong, Cooperative coevolution: an architecture for evolving coadapted subcomponents, Evol. Comput. 8 (1) (2000) 1–29.
[47] R. Wiegand, W. Liles, K. De Jong, An empirical analysis of collaboration methods in cooperative coevolutionary algorithms, in: Proceedings of the Genetic and Evolutionary Computation Conference, 2001, pp. 1235–1242.
[48] L. Panait, R. Wiegand, S. Luke, Improving coevolutionary search for optimal multiagent behaviors, in: Proceedings of the International Joint Conference on Artificial Intelligence, Morgan Kaufmann, 2003, pp. 653–658.
[49] C. Au, H. Leung, Biasing mutations in cooperative coevolution, in: Proceedings of IEEE Congress on Evolutionary Computation, 2007, pp. 828–835.
[50] K. Tan, Y. Yang, C. Goh, A distributed cooperative co-evolutionary algorithm for multi-objective optimization, IEEE Trans. Evol. Comput. 10 (5) (2006) 527–549.
[51] N. García-Pedrajas, J.R. del Castillo, D. Ortiz-Boyer, A cooperative coevolutionary algorithm for instance selection for instance-based learning, Mach. Learn. 78 (3) (2010) 381–420.
[52] J. Derrac, S. García, F. Herrera, Ifs-coco: instance and feature selection based on cooperative coevolution with nearest neighbor rule, Pattern Recognition 43 (6) (2010) 2082–2105.
[53] N. García-Pedrajas, C. Hervas-Martínez, D. Ortiz-Boyer, Cooperative coevolution of artificial neural network ensembles for pattern classification, IEEE Trans. Evol. Comput. 9 (3) (2005) 271–302.
[54] M. Li, J. Tian, F. Chen, Improving multiclass pattern recognition with a co-evolutionary RBFNN, Pattern Recognition Lett. 29 (4) (2008) 392–406.
[55] X. Ma, H. Wu, Power system short-term load forecasting based on cooperative co-evolutionary immune network model, in: Proceedings of 2nd International Conference on Education Technology and Computer, 2010, pp. 582–585.
[56] M. Qian-Li, Z. Qi-Lun, P. Hong, Z. Tan-Wei, Q. Jiang-Wei, Multi-step-prediction of chaotic time series based on co-evolutionary recurrent neural network, Chinese Physics B 17 (2) (2008).
[57] B. Bowerman, R. O'Connell, A. Koehler, Forecasting: Methods and Applications, Thomson Brooks/Cole, Belmont, CA, 2004.
[58] S. Makridakis, A. Andersen, R. Carbone, R. Fildes, M. Hibon, R. Lewandowski, J. Newton, E. Parzen, R. Winkler, The accuracy of extrapolation (time series) methods: results of a forecasting competition, J. Forecast. 1 (2) (1982) 111–153.
[59] J. Armstrong, F. Collopy, Error measures for generalizing about forecasting methods: empirical comparisons, Int. J. Forecast. 8 (1) (1992) 69–80.
[60] R. Fildes, The evaluation of extrapolative forecasting methods, Int. J. Forecast. 8 (1) (1992) 81–98.
[61] S. Makridakis, M. Hibon, The m3-competition: results, conclusions and implications, Int. J. Forecast. 16 (4) (2000) 451–476.
[62] R. Hyndman, A. Koehler, Another look at measures of forecast accuracy, Int. J. Forecast. 22 (4) (2006) 679–688.
[63] G. Zhang, M. Qi, Neural network forecasting for seasonal and trend time series, Eur. J. Oper. Res. 160 (2) (2005) 501–514.
[64] L. Eshelman, The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination, in: Proceedings of 1st Workshop on Foundations of Genetic Algorithms, 1991, pp. 265–283.
[65] L. Wang, J. Mendel, Generating fuzzy rules by learning from examples, IEEE Trans. Syst. Man Cybern. 22 (6) (2002) 1414–1427.
[66] A. Martínez-Estudillo, F. Martínez-Estudillo, C. Hervás-Martínez, N. García-Pedrajas, Evolutionary product unit based neural networks for regression, Neural Networks 19 (4) (2006) 477–486.
[67] J. Rustagi, Optimization Techniques in Statistics, Academic Press, Boston, 1994.
[68] J. Alcalá-Fdez, L. Sánchez, S. García, M. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas, J. Fernández, F. Herrera, Keel: a software tool to assess evolutionary algorithms for data mining problems, Soft Comput.—Fusion Foundations Methodol. Appl. 13 (3) (2009) 307–318.

[69] D. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman & Hall/CRC, 2004.
[70] J. Zar, Biostatistical Analysis, Prentice Hall, Englewood Cliffs, 1999.
[71] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Comput. 13 (10) (2009) 959–977.
[72] S. Holm, A simple sequentially rejective multiple test procedure, Scand. J. Stat. 6 (2) (1979) 65–70.

Elisabet Parras Gutierrez received the Ph.D. degrees in Computer Science from the University of Jaén, Jaén, Spain, in 2011. She is a member of the research group Intelligent Systems and Data Mining that belongs to the Department of Computer Science of the University of Jaén, Spain. Her current research interests include radial basic function neural networks, time series forecasting, evolutionary algorithms, and data mining.

Víctor Manuel Rivas Santos was born in Menen (Belgium), although he rapidly moved (was moved) to Granada (Spain). He finished his studies on Computer Sciences in 1994, and obtained its Ph.D., also in Computer Sciences, in 2003, both of them from the University of Granada. He is currently a professor in the University of Jaén, where he works since 1996 at the Department of Computer Sciences. His research is centered on the Soft Computing area, especially neural networks as well as genetic and co-evolutionary algorithms applied to time-series prediction.

María Isabel García Arenas has Ph.D. in Computer Science and is a lecturer at the University of Granada; her main interests include parallel and distributed computation, and she has been the leader of several local research and innovation projects. She has also worked in the areas of neural networks or evolutionary computation.

Maria Jose del Jesus Díaz received the M.Sc. and Ph.D. degrees in Computer Science from the University of Granada, Granada, Spain, in 1994 and 1999, respectively. She is an Associate Professor with the Department of Computer Science, University of Jaén, Spain. Her current research interests include fuzzy rule based systems, genetic fuzzy systems, subgroup discovery, data preparation, feature selection, evolutionary radial basis neural networks, knowledge extraction based on evolutionary algorithms, and data mining.