Box and Jenkins Nonlinear System Modelling Using RBF Neural Networks Designed by NSGAII

Kheireddine Lamamra, Khaled Belarbi and Souaad Boukhtini

Abstract In this work, we use radial basis function neural network for modeling nonlinear systems. Generally, the main problem in artificial neural network is often to find a better structure. The choice of the architecture of artificial neural network for a given problem has long been a problem. Developments show that it is often possible to find architecture of artificial neural network that greatly improves the results obtained with conventional methods. We propose in this work a method based on No Sorting Genetic Algorithm II (NSGA II) to determine the best parameters of a radial basis function neural network. The NSGAII should provide the best connection weights between the hidden layer and output layer, find the parameters of the radial function of neurons in the hidden layer and the optimal number of neurons in the hidden layers and thus ensure learning necessary. Two functions are optimized by NSGAII: the number of neurons in the hidden layer of the radial basis function neural network, and the error which is the difference between desired input and the output of the radial basis function neural network. This method is applied to modeling Box and Jenkins system. The obtained results are very satisfactory.

Keywords NSGAII · Radial basis function (RBF) neural networks · Optimization · Modelling · Non linear system · Box and Jenkins system

K. Lamamra (✉)
Department of Electrical Engineering, University of Oum El Bouaghi, Oum El Bouaghi, Algeria
e-mail: l_kheir@yahoo.fr

K. Lamamra
Laboratory of Mastering of Renewable Energies, University of Bejaia, Bejaia, Algeria

K. Belarbi · S. Boukhtini
University of Constantine, Constantine, Algeria
e-mail: kbelarbi@yahoo.com

S. Boukhtini
e-mail: sou_boukh@yahoo.fr

A.T. Azar and S. Vaidyanathan (eds.), *Computational Intelligence Applications in Modeling and Control*, Studies in Computational Intelligence 575,
DOI 10.1007/978-3-319-11017-2_10

1 Introduction

The first phase of modelling is to bring together the knowledge which we have about the process behaviour, from experiments and/or theoretical analysis of physical phenomena. This knowledge leads to several model assumptions. Each of these dynamic models realizes nonlinear functions between its control variables, state, and output. In the case where these functions are unknown, a black box model is used [56]. If some functions can be fixed from the physical analysis, then we talk about knowledge model [21, 56].

The second phase is to select the best model. This phase is the identification; this involves estimating the parameters of competing models. The estimation of the model parameters is performed by minimizing a cost function determined from the difference between the measured process outputs and the predicted values (prediction error).

The quality of this estimate depends on the wealth of learning sequences and effectiveness of the used algorithm. After the identification of all hypothesis models, we use the hypothesis corresponding to the best obtained predictor; the final model validation is performed according to the performance of its intended use [6, 20, 58].

There are several modelling tools, among them artificial neural networks. Several studies are currently using artificial neural networks in the field of modelling. for example: Grasso et al. [31] proposed "a new neural architecture able to accomplish the identification task. It is based on a relatively new neural algorithm, the multi-valued neural network with complex weights. The main idea is to use a set of measurements or simulations made on the system, taken at different values of geometrical parameters and at different frequencies, to train a multilayer architecture with multi-valued neurons, able to estimate the electrical parameters of the lumped model".

Badkar et al. [7] presented "a study the Laser transformation hardening of commercially pure titanium, nearer to ASTM grade 3 of chemical composition was investigated using continuous wave 2 kW, Nd: YAG laser. The effect of laser process variables such as laser power, scanning speed, and focused position was investigated using response surface methodology and artificial neural network keeping argon gas flow rate of 10 lpm as fixed input parameter". They described in their work, "the comparison of the heat input (HI) and ultimate tensile strength (σ) (simply called as tensile strength) predictive models based on artificial neural network and response surface methodology. The performance of the developed artificial neural network models were compared with the second-order RSM mathematical models of HI and σ. There was good agreement between the experimental and simulated values of response surface methodology and artificial neural network".

Among the advantages of a neural network is its ability to adapt to the conditions imposed by any environment, and ease to change its parameters (weight, number of neurons, etc…) depending on the behaviour of its environment The neural networks

are used to model and control dynamic systems linear and nonlinear where conventional methods fail [18, 41].

Research in the field of neural networks is focused on architecture by which neurons are combined and methodologies by which the weight of interconnections are calculated or adjusted.

Currently researchers are divided into two groups, the first is made up of biologists, physicists and psychologists, this group is trying to develop a neural model able to mimic with a given accuracy, the behaviour of the brain, the second group consists of engineers who are concerned with how the artificial neurons are interconnected to form networks with powerful computing capabilities. Actually, studies of neural networks are expanding and their use is still growing rapidly [57, 65].

Usually we associate with an artificial neural network learning algorithm to modify the processing performed in order to achieve a given task. For artificial neural network, learning can be seen as the problem of updating the weights of the connections within the network, in order to succeed the requested task [8, 40].

Generally learning neural networks can be made in two ways. In supervised learning, we have a set of examples (input-output pairs) and we must learn to give the correct output of new inputs. In reinforcement learning: we have inputs describing a situation and we receive a punishment (or error) if we give out is not adequate [11, 16, 44].

In supervised learning, the identification of the parameters of neural network is often performed by the algorithms of back-propagation, based on minimizing the training error and the chaining rule [12, 66]. This algorithm is a gradient descent on a differentiable error function.

This algorithm showed several disadvantages such as slow convergence, sensitivity to local minima and the difficulty to adjust the learning parameters (the number of neurons in the hidden layers, learning step etc…). In some networks using Hebbian learning where the synaptic weights can be adjusted during a learning phase patterns through Hebb's formula leads to a formula expressing the weights based on grounds recognized [4, 28, 51].

Several approaches have been proposed to improve the method of back-propagation as the modification of learning step, decentralization of learning step algorithms using quasi-Newton [37], genetic algorithms [59] …etc.

In this work we propose the use of No Sorting Genetic Algorithm II (NSGA II) to construct a model using a RBF neural network with optimal structure. In this approach the NSGA II is used to optimize the number of neurons in the hidden layer of neural network, find the best connection weights between the hidden layer and output layer, find the parameters of the radial function of neurons in the hidden layer and ensure learning of neural network.

This paper arbitrary small on a compact region [15, 29, 33, 49] is organized as follows: in the second section we briefly recall the basic principles of neural networks, in the third section we present the NSGA II algorithm, its operating principle and its application in our method, and the fourth section we present the learning of radial basis function neural networks by the NSGAII, and finally we present the simulation results of the developed method in the fifth section.

2 Neural Networks "NN"

Artificial neural networks, is a branch of artificial intelligence research that aims to simulate intelligent behaviour by mimicking the way that biological neural networks work. Most artificial intelligence methods seek to reproduce human intelligence by imitating "what we do". Artificial neural networks seek to reproduce it by imitating "the way that we do it". The origins of artificial neural networks proceeded computers by some decades, but it was not until computers became generally available that real progress could be made in the development of these methods [5, 9].

There was a slight "glitch" of a decade or so following the publication of a book that heavily criticized the possibility of artificial neural networks developing into anything useful; since then, progress has been dramatic and these tools have moved on from being oddities used by specialists to general-purpose algorithms for data analysis and pattern recognition tasks [9].

A neural network is a computational model whose design is inspired schematically the functioning of real neurons [42]. Artificial neural networks are increasingly used and applied in various fields [47, 61, 64].

This concept, as well as genetic algorithms, is linked to the notion of learning that allows computers to learn by example, experience or analogy, forming the foundation for adaptive systems.

An artificial neural network is generally composed of a succession of layers, each of which takes its inputs to the outputs of the previous one. Each layer is composed of neurons, and each synapse is associated a synaptic weight.

A neural network has the ability to adapt to the conditions imposed by any environment, and easy replacement when a there are a change of the parameters of this environment, which allows him to solve problems previously qualified as difficult [24, 39, 63].

2.1 Radial Basis Function RBF NN

The radial basis function neural network is a two-layer network in which the hidden layer performs a nonlinear transformation usually a Gaussian function to map the input space (Fig. 1), then the output layer combines the outputs of the intermediate layer linearly as the outputs of whole network. They may be considered as linearly parameterized networks.

RBF's are Gaussian functions have the form:

$$f_i(x) = e^{\frac{-d(x)}{\sigma_i^2}}$$

With:

$d(x)$ the Euclidian distance given by: $d(x) = \|x - c_i\|$

c_i centers of Gaussian functions

σ_i widths of Gaussian functions

The radial basis function network approximation error can be reduced by increasing the number of the adjustable weights. Universal approximation results for neural networks indicate that, if the number of RBF is large enough, then the error can be made arbitrary small on a compact region [15, 29, 33, 49].

The adjustable parameters of a radial basis function neural network are thus the centers c_i and the variance sigma of the radial basis functions and the weights of the connections.

The centers and the variance are usually adjusted off line using for instance the k-means based on the data sets to be approximated while the weights of the connections are adjusted on line, during the approximation process using various methods such as least mean square [36], genetic algorithm [17], particle swarm optimization [32] …etc.

Radial basis functions are powerful techniques for interpolation in multidimensional space. Radial basis functions have been applied in the area of neural network where they may be used as a replacement for the sigmoid hidden layer transfer characteristic in multi-layer Perceptron. Radial basis function network have two layers of processing: In the first, input is mapped onto each RBF in the 'hidden' layer. The RBF chosen is usually a Gaussian.

In regression problems the output layer is then a linear combination of hidden layer values representing mean predicted output. In classification problems the output layer is typically a sigmoid function of a linear combination of hidden layer values, representing a posterior probability [9, 42].

Radial basis function network have the advantage of not suffering from local minima in the same way as Multi-Layer Perceptron. Generally this is because the only parameters that are adjusted in the learning process are the linear mapping

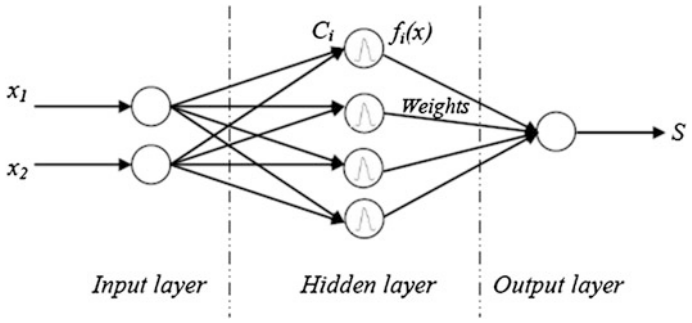

Fig. 1 General structure of a RBF neural network. Example of RBF neural network with two neurons in the input layer, four neurons in the hidden layer and one neuron in the output layer

from hidden layer to output layer. RBF network have the disadvantage of requiring good coverage of the input space by radial basis functions. RBF centers are determined with reference to the distribution of the input data, but without reference to the prediction task. As a result, representational resources may be wasted on areas of the input space that are irrelevant to the learning task. In this work, it is to the NSGAII algorithm to find the best RBF centers.

Currently, the RBF neural network are used in many works for different tasks, for example in network traffic identification, where "a method of network traffic identification based on RBF neural network is proposed by analysis of the current status of the network environment. The public data set and the real-time traffic are used for a combination of supervised learning" [62].

Gutiérrez et al. [34] used in their work, the RBF neural networks in a hybrid multi-logistic methodology, called logistic regression, "the process for obtaining the coefficients is carried out in three steps. First, an evolutionary programming (EP) algorithm is applied, in order to produce an RBF neural network with a reduced number of RBF transformations and the simplest structure possible. Then, the initial attribute space (or, as commonly known as in logistic regression literature, the covariate space) is transformed by adding the nonlinear transformations of the input variables given by the RBFs of the best individual in the final generation. Finally, a maximum likelihood optimization method determines the coefficients associated with a multilogistic regression model built in this augmented covariate space".

In the work of Pendharkar [48], radial basis function neural networks are used for classification problems, "a hybrid radial basis function network-data envelopment analysis (RBFN-DEA) neural network is proposed, the procedure uses the radial basis function to map low dimensional input data from input space R to a high dimensional R^+ feature space where DEA can be used to learn the classification function".

Sheikhan et al. [54, 55] presented a "RBF-based Active queue management (AQM) controller is used, RBF as a nonlinear controller is suitable as an AQM scheme to control congestion in transmission control protocol (TCP) communication networks since it has nonlinear behaviour. Particle swarm optimization algorithm is also employed to derive RBF output weights such that the integrated-absolute error is minimized. Furthermore, in order to improve the robustness of RBF controller, an error-integral term is added to RBF equation. The output weights and the coefficient of the integral error term in the latter controller are also optimized by Particle swarm optimization algorithm". Sheikhan et al. [54, 55] "makes the Lorenz hyper-chaos synchronization and its application to improve the security of communication systems. Two methods are proposed to synchronize the general forms of hyper-chaotic systems, and their performance in securing communication application is verified. The first method uses a standard RBF neural controller. Particle swarm optimization algorithm is used to derive and optimize the parameters of the RBF controller. In the second method, with the aim of increasing the robustness of the RBF controller, an error integral term is added to the equations of RBF neural network".

In the work of Xia et al. [60] "an energy-based controller is incorporated with RBF neural network compensation, which is used to swing up the pendubot and raise it to its uppermost unstable equilibrium position".

Radial basis function neural networks are also used in adaptive control, Jafarnejadsani et al. [38] proposed "an adaptive control based on radial-basis-function neural network for different operation modes of variable-speed variable-pitch wind turbines including torque control at speeds lower than rated wind speeds, pitch control at higher wind speeds and smooth transition between these two modes The adaptive neural network control approximates the nonlinear dynamics of the wind turbine based on input/output measurements and ensures smooth tracking of the optimal tip-speed-ratio at different wind speeds. The robust neural network weight updating rules are obtained using Lyapunov stability analysis".

The radial basis function neural networks are used also in identification of nonlinear systems, in the work of Chai and Qiao [13], RBF neural networks are used to "model the non linear system when the system runs without a fault, after some input and output data of the system are obtained, the center of the hidden nodes are chosen using clustering technology. Assuming that the system noise and approximation error are unknown but bounded, the output weights of RBF neural network model of the system are determined by a linear-in-parameter set membership estimation. An interval containing the actual output of the system running without a fault can be predicted based on the result of the estimation. If the measured output is out of the predicted interval, it can be determined that a fault has occurred".

Dos Santos Coelho et al. [52], used a "Radial Basis Function neural network with training combining the Gustafson-Kessel clustering method and a modified differential evolution in order to perform the swimmer velocity profile identification. The main idea is to obtain the dynamic of the velocity profile and to use it to improve the athletes' swim style. To achieve good performance with differential evolution algorithm, the tuning of control parameters is essential as its performance is sensitive to the choice of the mutation and crossover settings".

In this work authors "combines the two strategies described above proposing a modified differential evolution algorithm based on the association of a sinusoidal signal and chaotic sequences generated by logistic map for the mutation factor tuning. By using data collected from breaststroke and crawl swim style of an elite female swimmer; the validity and the accuracy of the RBF neural network model have been tested by simulations".

RBF neural network are used in optimization, in the work of Mukhopadhyay et al. [46] it's introduced "Discrete Hilbert Transform (DHT)-Neural Model which provides better result than the ARMA-Neural Model. a signal and its' DHT produces the same Energy Spectrum. Based on this concept DHT is used for Wind Speed forecasting purpose. Thereafter the RBF neural network is used on this to forecast wind power".

Chen et al. [14] proposed "a novel online modelling algorithm for nonlinear and non-stationary systems using a radial basis function neural network with a fixed number of hidden nodes. Each of the RBF basis functions has a tunable center vector and an adjustable diagonal covariance matrix. A multi-innovation recursive

least square (MRLS) algorithm is applied to update the weights of RBF online, while the modelling performance is monitored. When the modelling residual of the RBF network becomes large in spite of the weight adaptation, a node identified as insignificant is replaced with a new node, for which the tunable center vector and diagonal covariance matrix are optimized using the quantum particle swarm optimization (QPSO) algorithm".

2.2 Learning of a Neural Network

Learning is to determine the weights for the output of the neural network to be as close as possible to the target. The main problem is how to build a neural network, how many layers of covers and the number of units (or neurons) in hidden layer required achieving a good approximation. Since a wrong choice can lead to poor network performance matching [23].

The first attempts to solve the problem of determining the architecture have been to test several networks with different architectures to achieve the desired performance [27].

In recent years, many studies have been devoted to developing methods for optimizing the architecture of the neural network. The main algorithms have been proposed can be classified into three families:

1. Pruning algorithms: detect and remove the weights or units that contribute little to the network performance [45].
2. Ascending or constructive algorithms: start from an approximate solution to the problem with a simple network and add if necessary unit or hidden layers to improve network performance [26].
3. The direct algorithms: define a suitable architecture and perform learning or perform both operations simultaneously, such as genetic algorithms [3].

3 Non-dominated Sorting Genetic Algorithm II: NSGA II

The Multi Objective Genetic Algorithm that we used in this work is the NSGAII (Non-dominated Sorting Genetic Algorithm) introduced and enhanced by Deb and Goel [19].

It is one of the most used and most cited in the literature algorithms [53]. It is widely used by many authors, not only in the context of multi-objective optimization, but also for comparison with other algorithms, it is considered as a benchmark by several researchers, for example: Hashmi et al. [35] used this algorithm in "a negotiation Web service that would be used by both the consumer and provider Web services for conducting negotiations for dependent QoS parameters".

Min et al. [43] used it in a "multi-objective history matching model to predict the individual performance".

In the work of Gossard et al. [30] NSGAII is "coupled with an artificial neural network to optimize the equivalent thermo-physical properties of the external walls (thermal conductivity kwall and volumetric specific heat (ρc) wall) of a building in order to improve its thermal efficiency".

Adham et al. [1] used NSGAII "as an optimization technique in combination with a multi-objective general optimization scheme with the thermal resistance model as an analysis, for a potential improvement in the overall performance of a rectangular micro-channel heat sink using a new gaseous coolant namely ammonia gas".

In the work of Prasad and Singru [50] NSGA-II is used to "select the optimum design of turbo-alternator (TA), a real-life TA used in an industry is considered". Domínguez et al. [22] proposed "a high-performance architecture for the NSGA-II using parallel computing, for evaluation functions and genetic operators. In the proposed architecture, the Mishra Fast Algorithm for finding the Non Dominated Set was used; it's proposed a modification in the sorting process for the NSGA-II that improves the distribution of the solutions in the Pareto front".

NSGAII is an algorithm establishing the dominance relationships between individuals and providing a fast sorting method of chromosomes [19]. This algorithm uses a measure of crowding around individuals to ensure diversity in the population. The principle of this algorithm is shown in Fig. 2.

At the beginning, an initial population is randomly generated, and then it undergoes a sorting using the concept of non-domination. Each solution is assigned a strength or rank equal to the level of non-dominance (1 for best, 2 for the next level, etc…). The reproduction step consists of a tournament for the selection of parents.

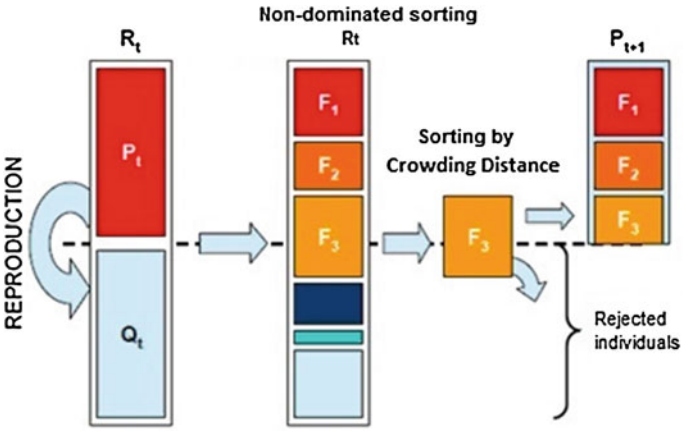

Fig. 2 Operating principle of NSGAII

When two individuals of the population are chosen randomly in the population, the tournament is based on a comparison of the domination with constraints of the two individuals. For a given generation t, we create $R_t = P_t U Q_t$, Q_t is children population of the previous population P_t (generated from the parents through the operators of crossover and mutation), R_t includes individuals of P_t, which ensures the elite nature of the algorithm NSGAII. Population R_t contains 2 N individuals (it is composed of N parents and N children). Then R_t undergoes a sorting using the concept of non-dominance of Pareto. Individuals are grouped into non-dominated fronts such as F_1 represents individuals of rank 1, F_2 individuals of rank 2, etc…

The next objective is to reduce the number of individuals in the 2 N population R_t for a population P_{t+1} of size N. If the size of F_1 is less than N, then all F_1 individuals are retained. It is the same for the other fronts as long as the number of individuals retained does not exceed the size N.

If we take the example of Fig. 2, the fronts F_1 and F_2 are fully retained but the conservation front F_3 will result in exceeding the size N of the population P_{t+1}. It must then make a selection of F_3 individuals to keep.

It is then necessary to make a selection of F_3 individuals to maintain. In this case, NSGAII involves a mechanism for preserving the diversity of the population based on the evaluation of the density of individuals around each solution through a procedure for calculating the "distance proximity".

A low value of the proximity distance for an individual is an individual "well surrounded". It then proceeds to a descending sorting according to this distance proximity to retain individuals F_3 front and eliminate individuals from the densest areas. This way we complete the population P_{t+1}. Individuals with extreme values for the criteria are preserved by this mechanism, thereby maintaining the external terminals of the Pareto front.

At the end of this phase, the population P_{t+1} is created. Then a new population Q_{t+1} is generated by reproduction from P_{t+1}. We continue iteratively the procedure described above to the satisfaction of stop criteria set by the user.

Generally, the NSGAII keeps elitism and diversity without adding additional parameters, while using an algorithm attractive in its simplicity with a minimum of parameters.

4 Learning of RBF NN by the NSGAII

The NSGA II is used to optimize the structure and parameters of the RBF NN. The following two objective functions are chosen:

- The first function to be optimized (f_1) is the number of neuron of the hidden layer of the RBF NN.
- The second function (f_2) is the quadratic error which is the difference between the desired input of the RBF NN and its output.

NSGAII must find the best number of neurons in the hidden layer (Nn) and provide the best connection weights between neurons in the hidden layer and the output layer, and find the parameters of the radial function of neurons hidden layer. In this work, we used the radial functions of Gaussian form, and NSGA II must find the best centers (C_i) and the best widths sigma (σ_i) for these functions.

The chromosome then contains the number of neurons in the hidden layer, Gaussian functions centers and widths of the hidden layer neurons, and the weights of connections between the hidden layer and the output layer.

The chromosome contains the following parameters:

$$[N_n C_1 C_2 \ldots C_{Nn} \sigma_1 \sigma_2 \ldots \sigma_{Nn} Z_1 Z_2 \ldots Z_{Nn}] \tag{1}$$

where:

N_n	is the number of neuron in the hidden layer
$C_2\ C_1 \ldots\ C_{Nn}$	Gaussian functions centers of the hidden layer neurons
$\sigma_1\ \sigma_2\ \ldots\ \sigma_{Nn}$	widths of the Gaussian functions of the hidden layer neurons
$Z_1\ Z_2 \ldots\ Z_{Nn}$	the connection weights between the neurons of the hidden layer and the neuron of the output layer

The length of the chromosome (*Lch*) depends only on the number of neurons in the hidden layer (N_n) and the number of neurons of the output layer (N_{ns}) because the inputs are fixed to two.

The general expression of the length of the chromosome (*Lch*) is given by [2]:

$$Lch = (2 + N_{ns}) * \max(N_n) + 1 \tag{2}$$

For example, for a neural network with one output and two neurons in the hidden layer, the length of the chromosome is *Lch* equal to 7: the number of neurons N_n (one allele), Gaussian functions centers neurons of the hidden layer C_1 C_2 (two alleles), the widths "σ" of the Gaussian functions of the hidden layer neurons σ_1 σ_2 (two alleles), the connection weights between the neurons of the hidden layer and the output layer neuron Z_1 Z_2 (two alleles).

The population size T_m (population matrix) is given by [3]:

$$Tm = N * ((2\ +\ N_{ns}) * \max(N_n) + 1) \tag{3}$$

N number of individuals in the population

For example, if the maximum number of neurons in the hidden layer is equal to 20, then the length of the population chromosomes is equal to 61. In this case if the neurons number in the chromosome i is equal to 6 ($N_{ni} = 6$), it will be organized as follows:

$$[N_{ni}(=6)\, C_1 \ldots C_6 \ldots \sigma_1 \ldots \sigma_6 Z_1 \ldots Z_6 0\, 0 \ldots 0]\, (Lch\ =\ 61) \tag{4}$$

N_{ni} neurons number in the hidden layer of the *i*th individual

Bellow an example of 3 individuals in a population composed of 80 individuals where N_{nmax} is 20:

$$N_{n1} = 5;\ N_{n30} = 20;\ N_{n80} = 2$$

$$\begin{vmatrix} Nn_1(=5)\ \ C_1 \ldots C_5 & \sigma_1 \ldots \sigma_5\ Z_1 \ldots Z_5\ 0\ 0 & 0 \\ \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots \\ Nn_{30}(=20)\ \ C_1 \ldots C_{20} & \sigma_1 \ldots \sigma_{20}\ Z_1 & Z_{20} \\ \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots \\ Nn_{80}(=2)\ \ C_1 C_2 & \sigma_1 \sigma_2\ \ Z_1 Z_2\ 0\ 0 & 0 \end{vmatrix}$$

5 Results of Simulation

This method is applied to modelling the BOX and JENKINS system, which is a time series. This process is a gas-fired boiler with the input the gas at the inlet and the output the concentration of released CO_2.

> A time series is a sequence of observations on a variable measured at successive points in time or over successive periods of time. The measurements may be taken every hour, day, week, month, or year, or at any other regular interval. The pattern of the data is an important factor in understanding how the time series has behaved in the past. If such behaviour can be expected to continue in the future, we can use the past pattern to guide us in selecting an appropriate forecasting method.
> To identify the underlying pattern in the data, a useful first step is to construct a time series plot. A time series plot is a graphical presentation of the relationship between time and the time series variable; time is on the horizontal axis and the time series values are shown on the vertical axis. Let us review some of the common types of data patterns that can be identified when examining a time series plot [2].

The data of BOX and JENKINS consist of 296 measurements of input and output [10, 25].

The RBF neural network has two inputs, one output and a hidden layer. The number of neurons in the hidden layer Nn is determined by the NSGA II as well as the centers and the widths of the Gaussian functions, and the weights of connection between the hidden layer and the output layer.

The NSGA II optimizes simultaneously Nn and the quadratic cumulated error e_c given by:

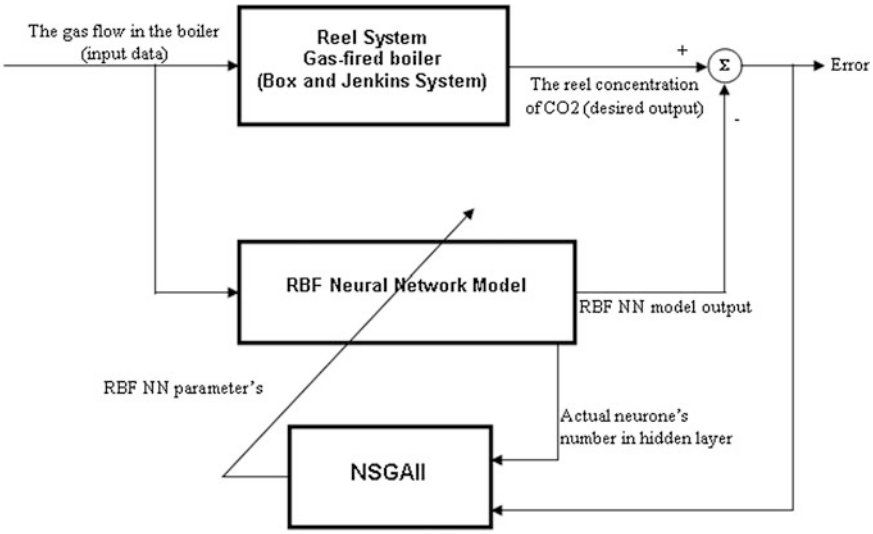

Fig. 3 Modelling schema of Box and Jenkins system using RBF NN

$$e_c = \sum_{i=1}^{N} e^2(i); \quad \text{with } e(i) = \sum_{i=1}^{N} (Y_d(i) - Y_r(i))$$

where:

e_c cumulative error. e(i): instantaneous error. N: length of the simulation sequence (the number of data N = 296)

Yd the desired out put,

Yr real output (output of the RBF NN model)

The modelling schema is presented in the Fig. 3,

The neural network learning is performed on 100 data of model of Box and Jenkins and validation is performed on the remaining data (196 data).

The results obtained which represent the first Pareto front (which containing the best individuals or non dominated individuals) of the last generation is given in Table 1.

By analyzing this table, we can notice that the Pareto front contains five non dominated individuals. There is an important difference between the global error of the individual 5 and the other individuals. It was selected as the best model. The structure of this RBF neural network model is composed of seven neurons in the hidden layer and an overall error 0.5259.

Figure 4 shows the gas flow in the boiler, which represent the global real input data.

Figure 5 shows the real input training data.

Figure 6 represents the real input validation data.

Table 1 Pareto front

No. of individual	Number of neurons in the hidden layer Nn	Training error (100 data) e_t	Validation error (196 data) e_V	Global error (296 data) e.g.
1	2	5.4241	11.0611	16.4852
2	3	4.7137	8.7448	13.4585
3	4	3.0042	6.1185	9.1227
4	6	1.0218	3.5033	4.5251
5	7	0.2043	0.3216	0.5259

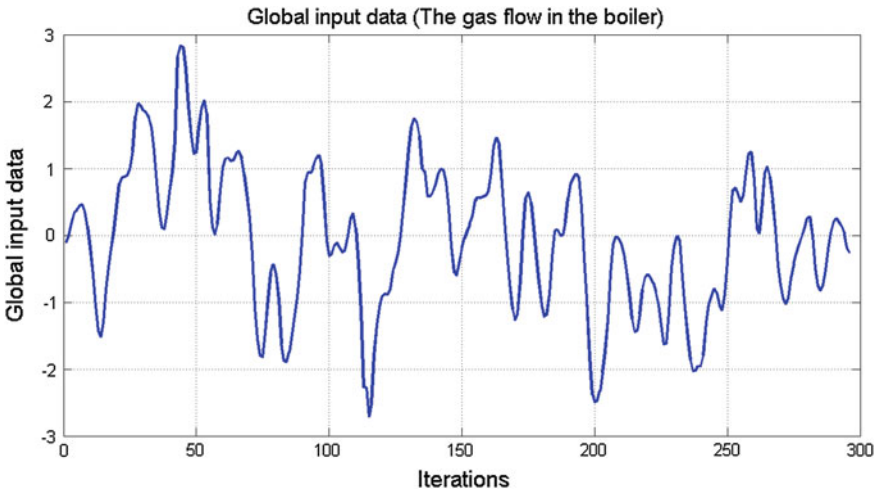

Fig. 4 Global input data, Gaz flow in the boiler

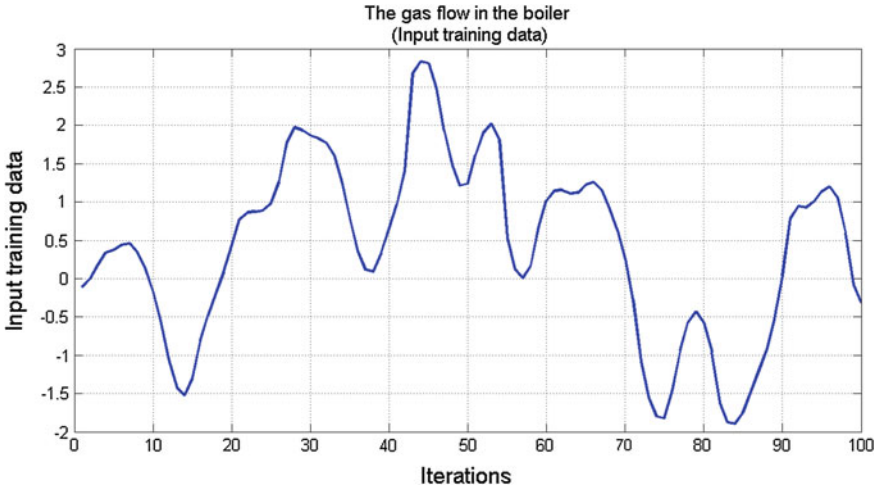

Fig. 5 Input training data

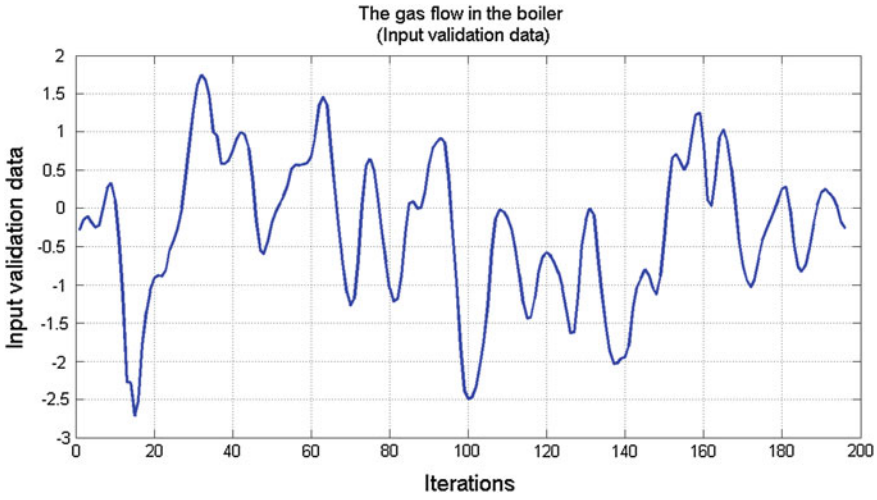

Fig. 6 Input validation data

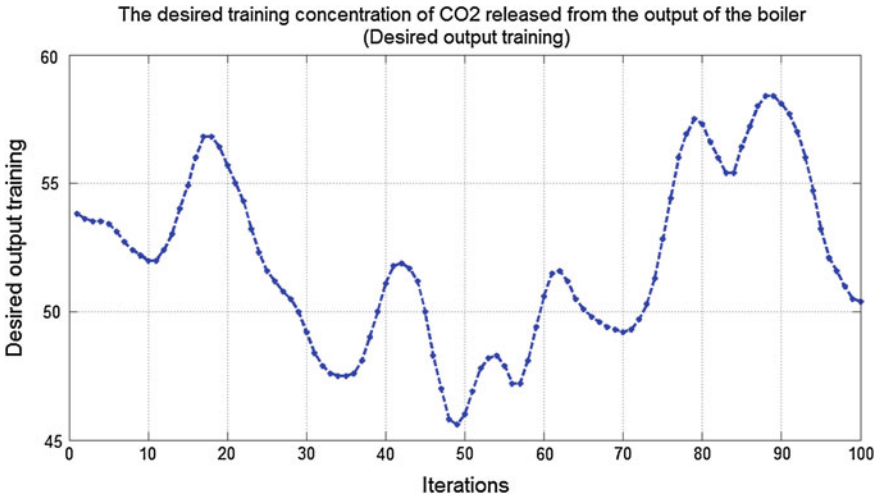

Fig. 7 Desired output training

Figure 7 represents the desired output training, which represent the desired training concentration of CO_2 released from the output of the boiler.

Figure 8 shows the RBF neural network model output during training phase.

Figure 9 represents the desired output and RBF neural network model output during training phase, and the Training error is shown in the Fig. 9.

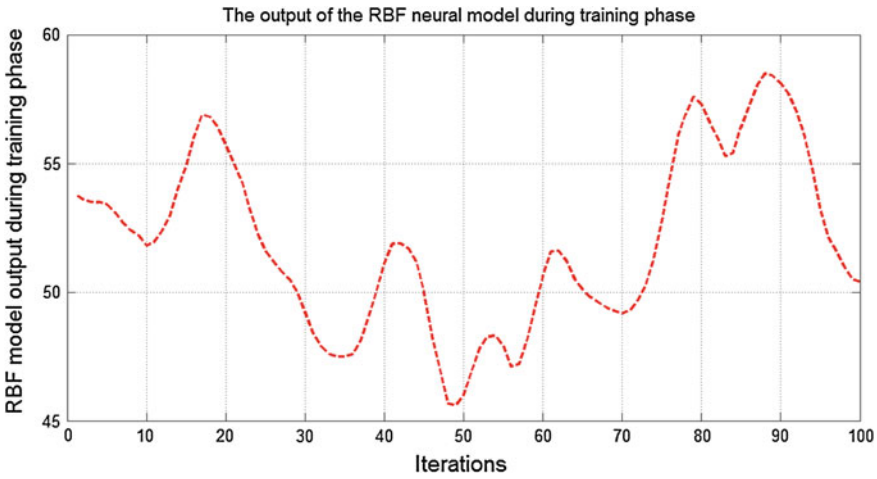

Fig. 8 RBF neural network model output during training phase

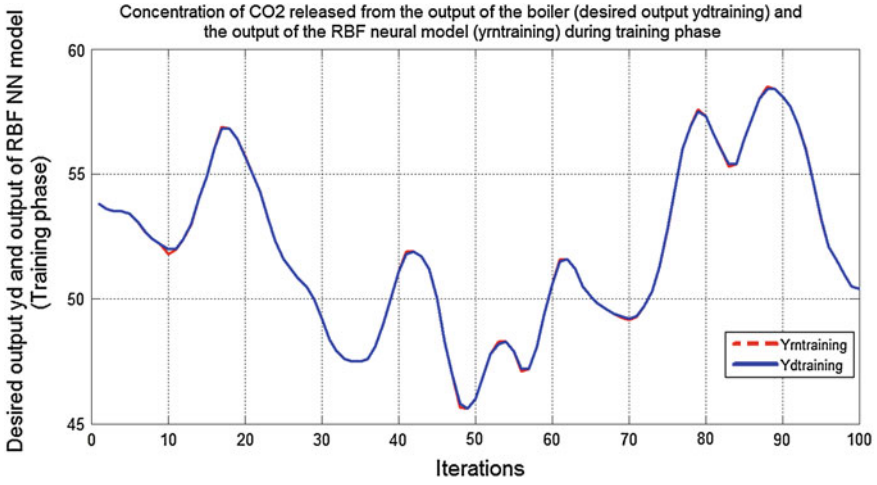

Fig. 9 Desired output and RBF neural network model output during training phase

By analyzing these figures, we can observe that the output of RBF neural network model has perfectly followed the desired output during the training phase with a training error of 0.2043.

In the training error figure (Fig. 10), the most significant peak appears at 10th iteration.

Figure 11, represents the desired validation concentration of CO_2 released from the output of the boiler (the desired output validation).

Figure 12, shows the RBF neural network model output during validation phase.

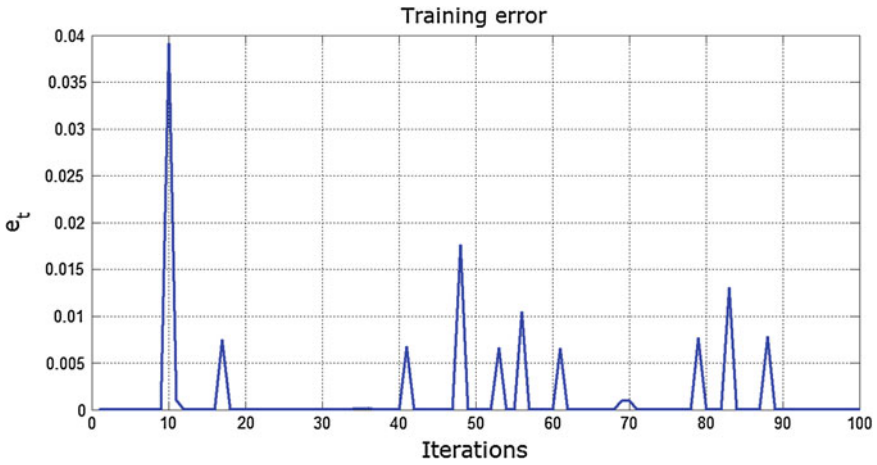

Fig. 10 Training error

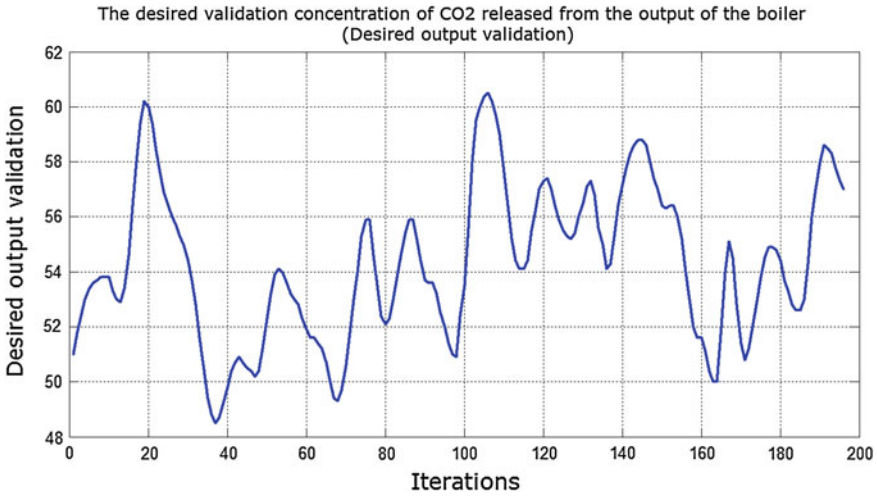

Fig. 11 Desired output validation

Figure 13, represents the desired output and the RBF neural network model output during validation phase, and the validation error is shown in the Fig. 14.

In these figures we can also observe that the output of RBF neural network model has followed the desired output during the validation phase with a training error of 0.3216.

This error is larger than the training error; this can be justified of the fact that the training data number chosen is lower than validation data number.

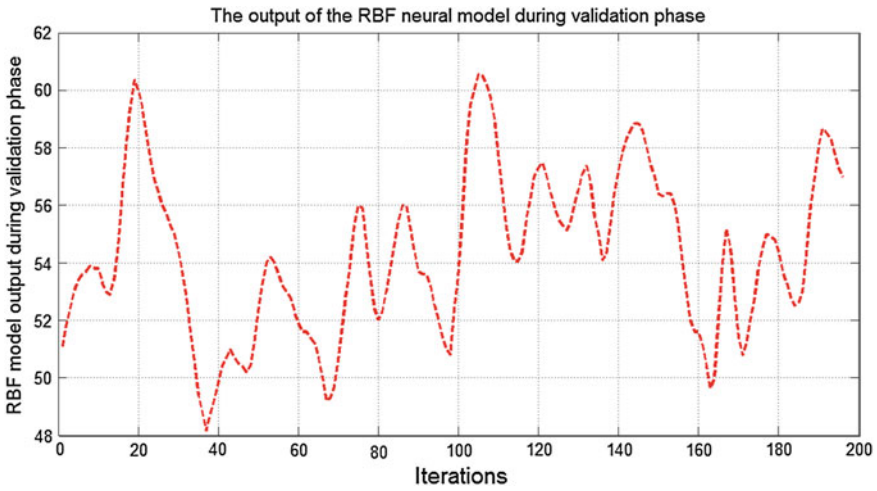

Fig. 12 RBF neural network model output during validation phase

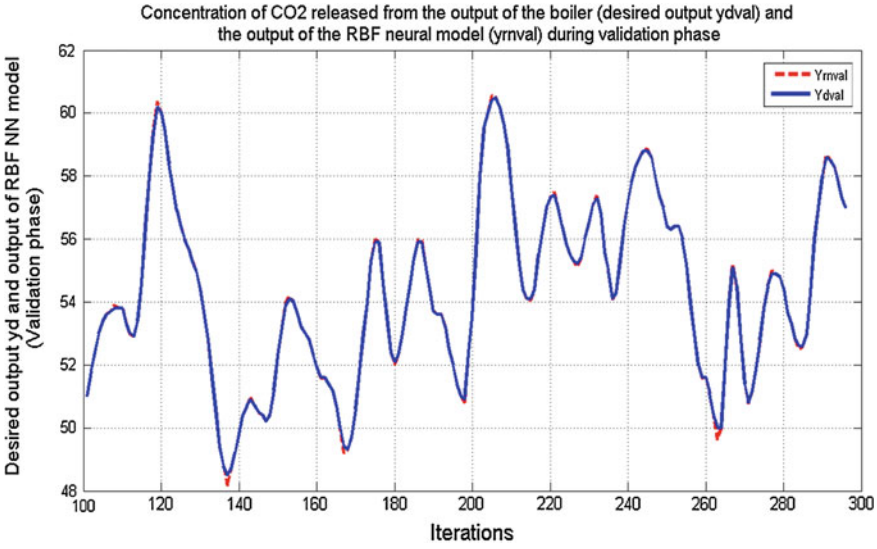

Fig. 13 Desired output and RBF neural network model output during validation phase

In the validation error figure (Fig. 14), the most significant peaks are appeared at 37th and 163th iterations.

The global desired output which is the global desired concentration of CO_2 released from the output of the boiler is shown in the Fig. 15.

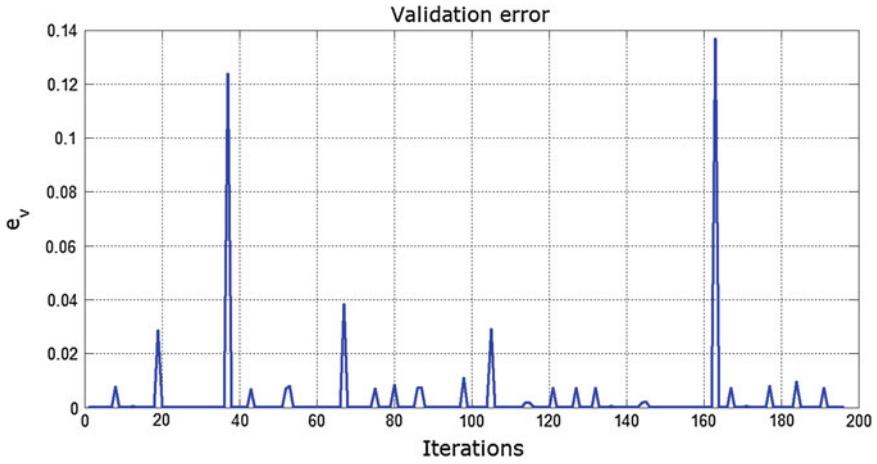

Fig. 14 Validation error

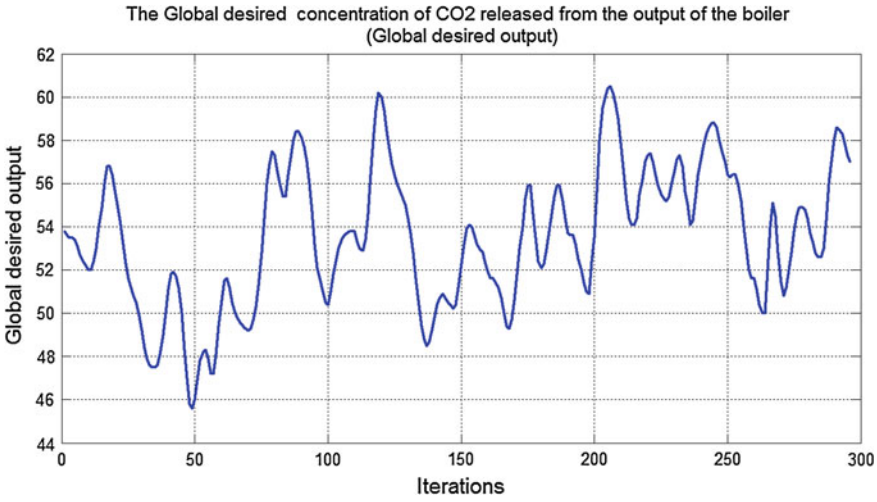

Fig. 15 Global desired output

Figure 16 represents the global RBF neural network model and the Fig. 17 shows the concentration of CO_2 released from the output of the boiler (desired output yd) and the output of the RBF neural model (y_r).

Figure 18 represents the global error, and finally, the Pareto front "global cumulated error function of the number of neurons" is shown in the Fig. 19.

Based on these results, we can conclude that the multi-objective genetic algorithm NSGAII gave a good structure of radial basis function neural network model, with a good number of neurons in the hidden layer and good connection weights

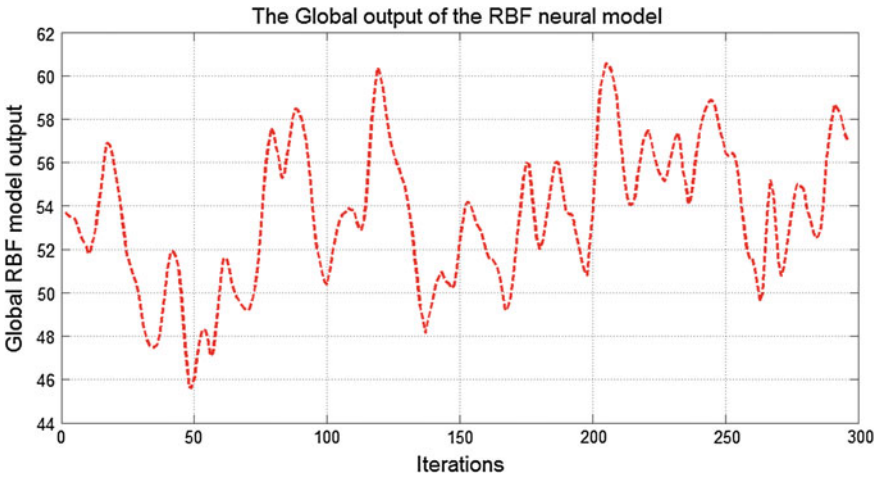

Fig. 16 Global RBF neural network model

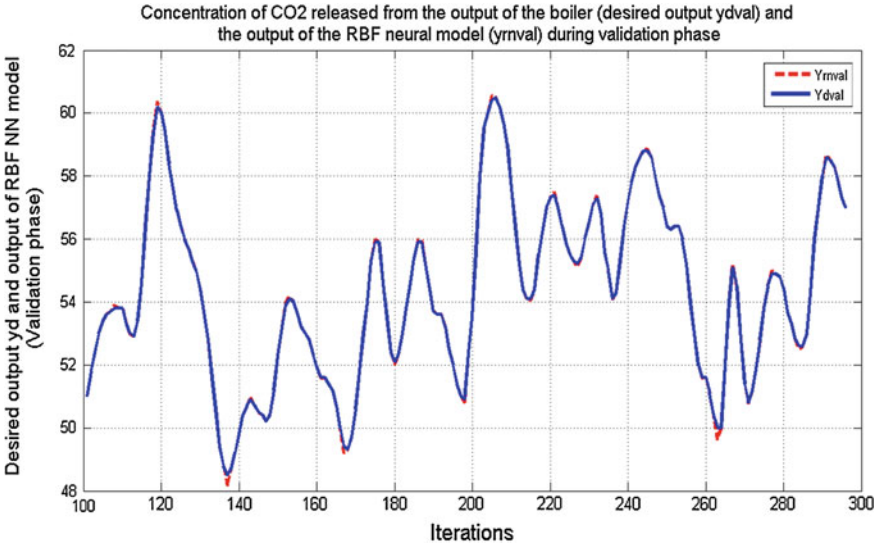

Fig. 17 The concentration of CO_2 released from the output of the boiler (desired output yd) and the output of the RBF neural model (year)

between the hidden layer and the output layer, and it also found the best parameters of the radial function of hidden layer neurons, because we see that the radial basis function neural network model output is very close to that of the desired output, with training error equal to 0.2043 and validation error equal to 0.3216 and a global error equal to 0.5259.

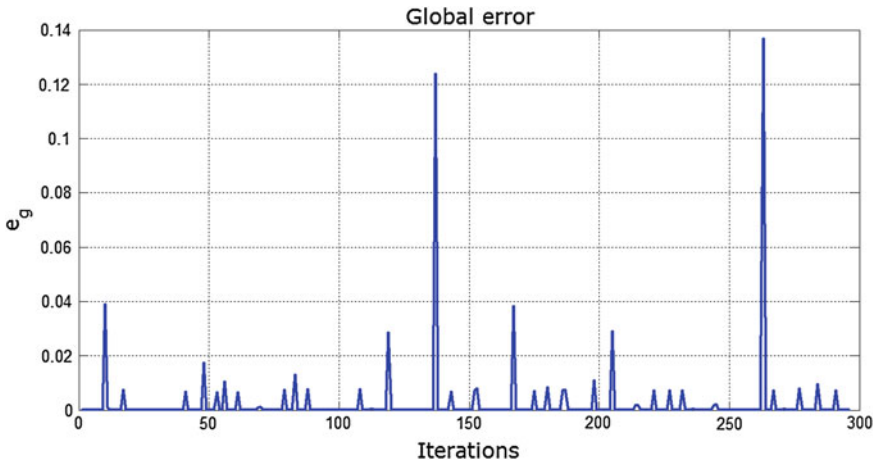

Fig. 18 The global error

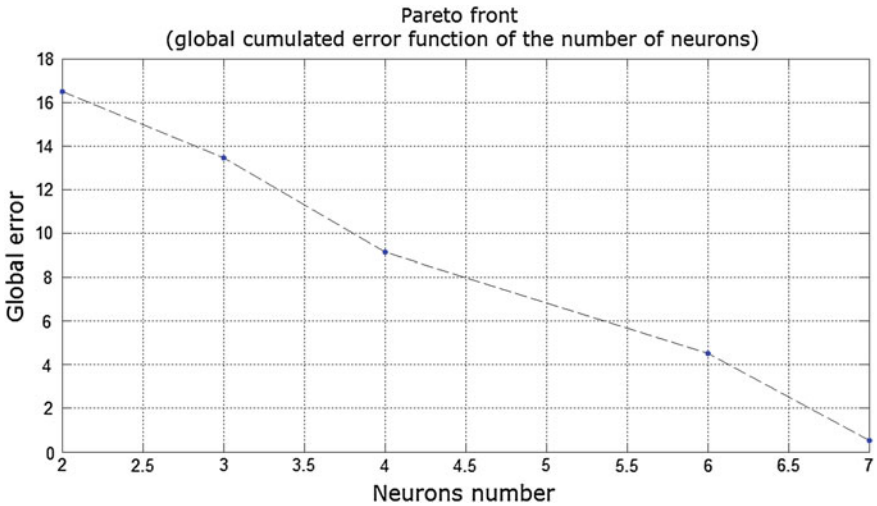

Fig. 19 Pareto front "global cumulated error function of the number of neurons"

We are currently working on the improvement of this technique; however we try to improve a new optimization method for Radial Basis Function and Multi Layer Perceptron neural networks. To do so, we did modelling by RBF and MLP neural networks with the optimization of three objectives.

The structure of neural network is hereby amended. In this technique, the input variables and the number of input neurons are not fixed and they will be included in the multi-objective optimization process carried out by the NSGAII algorithm. We have to minimize the following three functions:

- The number of neuron of the hidden layer of the radial basis function neural network.
- The quadratic error which is the difference between the desired input of the RBF neural network and its output.
- The regressor's number at the input of the RBF neural network.

6 Conclusion

Neural networks are increasingly used and applied in various fields, mainly in the problems of modelling of complex systems. This is due to their simplicity and their universal approximation properties and the ability of information parallel treatment. These properties make that these networks are well used for modeling and controlling linear and nonlinear dynamics systems, where conventional methods fail.

The most difficult problem to solve for neural networks is to obtain the best and right architecture. The networks established in most of practical applications are built with an experimental way.

This difficulty can be highlighted by a number of issues, such as the number of hidden layers to be used in a multilayer network, the optimal number of neurons in each layer and the initial values of connection weights during the learning phase … etc. A bad choice can lead to poor performances of the corresponding network.

In this work we present a technique to solve the problems mentioned above. This technique is to treat these problems as a multi-criteria optimization problem. We considered in this work the designing of RBF neural network using the multi-objective genetic algorithms type NSGAII, by optimizing simultaneously two objectives functions: the first function is the quadratic cumulatively error, which is the difference between the desired signal and the RBF neural network model output signal, and the second is the number of neurons in the hidden layer, thereby the NSGA II chromosome contains the number of neurons in the hidden layer, Gaussian functions centers and widths of the hidden layer neurons, and the weights of connections between the hidden layer and the output layer. At the end of the evolution of this algorithm off-line, we have a set of RBF models which forming the final Pareto front, and includes all allowed results, and ensuring the predefined criteria.

This optimization technique is applied to modelling a nonlinear system which is the BOX and JENKINS process. It's a gas-fired boiler with the input the gas at the inlet and the output the concentration of released CO_2. The results show that using NSGAII to optimize the RBF neural network provides a good model, these results are very satisfying.

At the end of the NSGAII algorithm evolution, it converges to a set of solutions (Pareto front), it is a set of solutions respecting the optimization criteria, which is generally difficult to choose one solution from the set, to resolve this problem we proposed in the future works, the uses of selection method such as the multi-criteria decision analysis approach.

In our future work, we are working on the improvement of this technique; nevertheless we try to improve a new optimization method for RBF and MLP neural network by the multi objectives genetic algorithms NSGAII and others and also using Particle Swarm Optimization. These techniques are applied in different areas, such as modelling and control of complex nonlinear systems, modelling of transistors, modelling and control in the field of renewable energies.

References

1. Adham, A.M., Mohd-Ghazali, N., Ahmad, R.: Optimization of an ammonia-cooled rectangular microchannel heat sink using multi-objective non-dominated sorting genetic algorithm (NSGA2). Heat Mass Transf. **48**(10), 1723–1733 (2012). doi:10.1007/s00231-012-1016-8
2. Anderson, D., Sweeney, D., Williams, T., Camm, J., Martin, R.: An introduction to management science: quantitative approaches to decision making, revised. Cengage Learning (2011)
3. Angeline, P.J., Saunders, G.M., Pollack, J.B.: An evolutionary algorithm that constructs recurrent neural networks. IEEE Trans. Neural Netw. **5**(1), 54–65 (1994). doi:10.1109/72.265960
4. Azar, A.T.: Adaptive neuro-fuzzy system as a novel approach for predicting post-dialysis urea rebound. Int. J. Intell. Syst. Technol. Appl. **10**(3), 302–330 (2011). doi:10.1504/IJISTA.2011.040352
5. Azar, A.T.: Fast neural network learning algorithms for medical applications. Neural Comput. Appl. **23**(3–4), 1019–1034 (2013). doi:10.1007/s00521-012-1026-y
6. Azar, A.T., Yashiro, M., Schneditz, D., Roa, L.M.: Double pool urea kinetic modeling. In: Azar, A.T. (ed.) Modelling and Control of Dialysis Systems. Studies in Computational Intelligence, vol. 404, pp. 627–687. Springer, Berlin (2013)
7. Badkar, D.S., Pandey, K.S., Buvanashekaran, G.: Development of RSM- and ANN-based models to predict and analyze the effects of process parameters of laser-hardened commercially pure titanium on heat input and tensile strength. Int. J. Adv. Manuf. Technol. **65**(9–12), 1319–1338 (2013). doi:10.1007/s00170-012-4259-0
8. Bermejo, D.M.A.V.: A mathematical model to predict δ- ferrite content in austenitic stainless steel weld metals. Weld. World **56**(9–10), 48–68 (2012). doi:10.1007/BF03321381
9. Binder, M.D., Hirokawa, N., Windhorst, U. (eds.) Artificial neural networks. In: Encyclopedia of Neuroscience, pp. 185–185. Springer, Berlin (2009)
10. Box, G.E.P., Jenkins, G.M., Reinsel, G.C.: Time Series Analysis: Forecasting and Control. Wiley, New York (2013)
11. Cantley, K.D., Subramaniam, A., Stiegler, H.J., Chapman, R.A., Vogel, E.M.: Hebbian learning in spiking neural networks with nanocrystalline silicon TFTs and memristive synapses. IEEE Trans. Nanotechnol. **10**(5), 1066–1073 (2011)
12. Carrasco, R., Sanchez, E.N., Carlos-Hernandezy, S.: Neural network identification for biomass gasification kinetic model. In: The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 1888–1893. Available at http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6033454 (2011). Accessed: 2 May 2014
13. Chai, W., Qiao, J.: Non-linear system identification and fault detection method using RBF neural networks with set membership estimation. Int. J. Model. Ident. Control **20**(2), 114 (2013). doi:10.1504/IJMIC.2013.056183
14. Chen, H., Gong, Y., Hong, X.: Online Modeling With Tunable RBF Network. IEEE Trans. Cybern. **43**(3), 935–947 (2013). doi:10.1109/TSMCB.2012.2218804

15. Chen, T., Chen, H.: Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks. IEEE Trans. Neural Netw. **6**(4), 904–910 (1995). doi:10.1109/72.392252
16. Cherkassky, V., Friedman, J.H., Wechsler, H.: From statistics to neural networks: theory and pattern recognition applications. Springer Publishing Company, Incorporated (2012)
17. Cook, D.F., Ragsdale, C.T., Major, R.L.: Combining a neural network with a genetic algorithm for process parameter optimization. Eng. Appl. Artif. Intell. **13**(4), 391–396 (2000). doi:10.1016/S0952-1976(00)00021-X
18. Dahl, G.E., Sainath, T.N. and Hinton, G.E.: Improving deep neural networks for LVCSR using rectified linear units and dropout. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8609–8613. IEEE (2013)
19. Deb, K., Goel, T.: Controlled Elitist non-dominated sorting genetic algorithms for better convergence. In: Zitzler, E., Thiele, L., Deb, K., Coello, C.A.C., Corne, D. (eds.) Evolutionary Multi-Criterion Optimization. Lecture Notes in Computer Science, pp. 67–81. Springer, Berlin (2001)
20. Deichmueller, M., Denkena, B., de Payrebrune, K.M., Kröger, M., Wiedemann, S., Schröder, A., Carstensen, C. (2013) Modeling of process machine interactions in tool grinding. In: Process Machine Interactions, pp. 143–176. Springer, Berlin
21. Deutschmann, O.: Modeling and Simulation of Heterogeneous Catalytic Reactions: From the Molecular Process to the Technical System. Wiley, New York (2013)
22. Domínguez, J., Montiel-Ross, O., Sepúlveda, R.: High-performance architecture for the modified NSGA-II. In: Melin, P., Castillo, O. (eds.) Soft Computing Applications in Optimization, Control, and Recognition. Studies in Fuzziness and Soft Computing, vol. 294, pp. 321–341. Springer, Berlin Heidelberg (2013)
23. Urbani, D., Marcos, S., Thiria, S.: (1995) Statistical methods for selecting neural architectures: application to the design of models of dynamic processes. PhD thesis of University Pierre and Marie Curie
24. Furtuna, R., Curteanu, S., Leon, F.: Multi-objective optimization of a stacked neural network using an evolutionary hyper-heuristic. Appl. Soft Comput. **12**(1), 133–144 (2012). doi:10.1016/j.asoc.2011.09.001
25. Box, G.E.P., Jenkins, G.M.: Time series analysis: forecasting and control, p. 575. Holden-Day, San Francisco (1976)
26. Giles, C.L., Chen, D., Sun, G.-Z., Chen, H.-H., Lee, Y.-C., Goudreau, M.W.: Constructive learning of recurrent neural networks: limitations of recurrent cascade correlation and a simple solution. IEEE Trans. Neural Netw. **6**(4), 829–836 (1995). doi:10.1109/72.392247
27. Giles, C.L., Miller, C.B., Chen, D., Chen, H.H., Sun, G.Z., Lee, Y.C.: Learning and extracting finite state automata with second-order recurrent neural networks. Neural Comput. **4**(3), 393–405 (1992). doi:10.1162/neco.1992.4.3.393
28. Gilson, M., Py, J.S., Brault, J.-J., Sawan, M.: Training recurrent pulsed networks by genetic and Taboo methods. In: Canadian Conference on Electrical and Computer Engineering, 2003, IEEE CCECE 2003, vol. 3, pp. 1857–1860 (2003). doi:10.1109/CCECE.2003.1226273
29. Girosi, F., Poggio, T.: Networks and the best approximation property. Biol. Cybern. **63**(3), 169–176 (1990). doi:10.1007/BF00195855
30. Gossard, D., Lartigue, B., Thellier, F.: Multi-objective optimization of a building envelope for thermal performance using genetic algorithms and artificial neural network. Energy Build. **67**, 253–260 (2013). doi:10.1016/j.enbuild.2013.08.026
31. Grasso, F., Luchetta, A., Manetti, S., Piccirilli, M.C.: System identification and modelling based on a double modified multi-valued neural network. Analog Integr. Circ. Sig. Process **78** (1), 165–176 (2014). doi:10.1007/s10470-013-0211-y
32. Guerra, F.A., dos Coelho, L.S.: Multi-step ahead nonlinear identification of Lorenz's chaotic system using radial basis neural network with learning by clustering and particle swarm optimization. Chaos, Solitons Fractals **35**(5), 967–979 (2008). doi:10.1016/j.chaos.2006.05.077

33. Gupta, M.M., Rao, D.H.: Neuro-control systems: theory and applications. IEEE, New York (1993)
34. Gutiérrez, P.A., Hervas-Martinez, C., Martínez-Estudillo, F.J.: Logistic regression by means of evolutionary radial basis function neural networks. IEEE Trans. Neural Netw. **22**(2), 246–263 (2011). doi:10.1109/TNN.2010.2093537
35. Hashmi, K., Alhosban, A., Najmi, E., Malik, Z., Rezgui (2013) Automated Web service quality component negotiation using NSGA-2. In: 2013 ACS International Conference on Computer Systems and Applications (AICCSA), pp. 1–6. doi:10.1109/AICCSA.2013.6616502
36. Haykin, S., Widrow, B. (2003) Least-Mean-Square Adaptive Filters. Wiley, New York (2003)
37. Jacek M.Z.: Introduction to Artificial Neural Systems. Jaico Publishing House, Mumbai (1992)
38. Jafarnejadsani, H., Pieper, J., Ehlers, J.: Adaptive control of a variable-speed variable-pitch wind turbine using radial-basis function neural network. IEEE Trans. Control Syst. Technol. **21**(6), 2264–2272 (2013). doi:10.1109/TCST.2012.2237518
39. Lamamra, K., Belarbi, K., Bosche, J., Hajjaji, A.E.L.: A neural network controller optimised with multi objective genetic algorithms for a laboratory anti-lock braking system. Sci. Technol. J. Constantine 1 Univ **35** (2012)
40. Kasabov, N., Dhoble, K., Nuntalid, N., Indiveri, G.: Dynamic evolving spiking neural networks for on-line spatio-and spectro-temporal pattern recognition. Neural Networks **41**, 188–201 (2013)
41. Levine, D.S., Aparicio I.V.M.: Neural networks for knowledge representation and inference. Psychology Press, Rouledge (2013)
42. Mallot, H.A.: Artificial neural networks. In: Computational Neuroscience, Springer Series in Bio-/Neuroinformatics, vol. 2, pp. 83–112. Springer International Publishing, Berlin (2013)
43. Min, B.H., Park, C., Jang, I.S., Lee, H.Y., Chung, S.H., Kang, J.M.: Multi-objective history matching allowing for scale-difference and the interwell complication. doi:10.3997/2214-4609.20130172
44. BG, Mirta: Dynamics of complex systems and applications to SHS: models, concepts, methods. Leibniz-IMAG Laboratory, Grenoble (2004)
45. Morse, J.N.: Reducing the size of the nondominated set: pruning by clustering. Comput. Oper. Res. **7**(1–2), 55–66 (1980). doi:10.1016/0305-0548(80)90014-3
46. Mukhopadhyay, S., Panigrahi, P.K., Mitra, A., Bhattacharya, P., Sarkar, M., Das, P.: Optimized DHT-RBF model as replacement of ARMA-RBF model for wind power forecasting. In: 2013 International Conference on Emerging Trends in Computing, Communication and Nanotechnology (ICE-CCN), pp. 415–419. doi:10.1109/ICE-CCN.2013.6528534 (2013)
47. Nikdel, N., Nikdel, P., Badamchizadeh, M.A., Hassanzadeh, I.: Using neural network model predictive control for controlling shape memory alloy-based manipulator. IEEE Trans. Industr. Electron. **61**(3), 1394–1401 (2014). doi:10.1109/TIE.2013.2258292
48. Pendharkar, P.C.: A hybrid radial basis function and data envelopment analysis neural network for classification. Comput. Oper. Res. **38**(1), 256–266 (2011). doi:10.1016/j.cor.2010.05.001. (Project Management and Scheduling)
49. Poggio, T., Girosi, F.: Networks for approximation and learning. Proc. IEEE **78**(9), 1481–1497 (1990). doi:10.1109/5.58326
50. Prasad, K.V.R.B., Singru, P.M.: Optimum design of turbo-alternator using modified NSGA-II algorithm. In: Bansal, J.C., Singh, P., Deep, K., Pant, M., Nagar, A. (eds.) Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012), Advances in Intelligent Systems and Computing, vol. 202, pp. 253–264. Springer, India (2013)
51. Roberto, B., Ubaldo, C., Stefano, M., Roberto, I., Elisa, S., Paolo, M.: Graybox and adaptative dynamic neural network identification models to infer the steady state efficiency of solar thermal collectors starting from the transient condition. Sol. Energy **84**(6), 1027–1046 (2010)

52. Dos Santos Coelho, L., Ferreira da Cruz, L., Zanetti Freire, R.: Swim velocity profile identification by using a modified differential evolution method associated with RBF neural network. In: 2013 Third International Conference on Innovative Computing Technology (INTECH), pp. 389–395. doi:10.1109/INTECH.2013.6653721 (2013)
53. Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J.J., Schwefel, H.-P. (eds.) (2000) A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. Parallel Problem Solving from Nature PPSN VI. Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin (2000)
54. Sheikhan, M., Shahnazi, R., Garoucy, S.: Hyperchaos synchronization using PSO-optimized RBF-based controllers to improve security of communication systems. Neural Comput. Appl. **22**(5), 835–846 (2013). doi:10.1007/s00521-011-0774-4
55. Sheikhan, M., Shahnazi, R., Hemmati, E.: Adaptive active queue management controller for TCP communication networks using PSO-RBF models. Neural Comput. Appl. **22**(5), 933–945 (2013). doi:10.1007/s00521-011-0786-0
56. Syed, A.A., Pittner, A., Rethmeier, M., De, A.: Modeling of gas metal arc welding process using an analytically determined volumetric heat source. ISIJ Int. **53**(4), 698–703 (2013)
57. Tang, Y., Wong, W.K.: Distributed synchronization of coupled neural networks via randomly occurring control. IEEE Trans. Neural Netw. Learn. Syst. **24**(3), 435–447 (2013)
58. Teixidor, D., Grzenda, M., Bustillo, A., Ciurana, J.: Modeling pulsed laser micromachining of micro geometries using machine-learning techniques. J. Intell. Manuf. 1–14. doi:10.1007/s10845-013-0835-x (2013)
59. Whitley, D., Starkweather, T., Bogart, C.: Genetic algorithms and neural networks: optimizing connections and connectivity. Parallel Comput. **14**(3), 347–361 (1990). doi:10.1016/0167-8191(90)90086-O
60. Xia, D., Wang, L., Chai, T.: Neural-network-friction compensation-based energy swing-up control of pendubot. IEEE Trans. Industr. Electron. **61**(3), 1411–1423 (2014). doi:10.1109/TIE.2013.2262747
61. Xiao, Z., Liang, S., Wang, J., Chen, P., Yin, X., Zhang, L., Song, J.: Use of general regression neural networks for generating the GLASS leaf area index product from time-series MODIS surface reflectance. IEEE Trans. Geosci. Remote Sens. **52**(1), 209–223 (2014). doi:10.1109/TGRS.2013.2237780
62. Xu, Y., Zheng, J.: Identification of network traffic based on radial basis function neural network. In: Chen, R. (ed.) Intelligent Computing and Information Science. Communications in Computer and Information Science, vol. 134, pp. 173–179. Springer, Berlin (2011)
63. Yu, H., Xie, T., Paszczynski, S., Wilamowski, B.M.: Advantages of radial basis function networks for dynamic system design. IEEE Trans. Industr. Electron. **58**(12), 5438–5450 (2011). doi:10.1109/TIE.2011.2164773
64. Yuan, J., Yu, S.: Privacy preserving back-propagation neural network learning made practical with cloud computing. IEEE Trans. Parallel Distrib. Syst. **25**(1), 212–221 (2014). doi:10.1109/TPDS.2013.18
65. Zhang, H., Yang, F., Liu, X., Zhang, Q.: Stability analysis for neural networks with time-varying delay based on quadratic convex combination. IEEE Trans. Neural Netw. Learn. Syst. **24**(4), 513–521 (2013)
66. Zhi, C., Guo, L.H., Zhang, M.Y., Shi, Y.: Research on dynamic subspace divided BP neural network identification method of color space transform model. Adv. Mater. Res. **174**, 97–100 (2011)