# Using Genetic Algorithms to Optimize Wavelet Neural Networks Parameters for Function Approximation

**Mohammed Awad**[1]
[1]**Department of Computer System Engineering, Faculty Engineering and Information Technology,**
**Arab American University-Palestine**

**Abstract**

This paper deals with the problem of function approximation from a given set of input/output data. This paper presents a new approach for solving the problem of function approximation from a given set of I/O data using Wavelet Neural Networks (WNN) and Genetic Algorithms (GAs). GAs has the property of global optimal search algorithm and WNNs are universal approximations, it's achieved faster convergence than Radial Basis Function Neural Networks (RBFN) and avoids stocking in local minimum. This approach is based on a new efficient method of optimizing WNNs parameters using GAs, it uses GA to optimize scale parameter Aj and the translation parameter Bj of the WNN such that each individual of the population represents scale parameter and translation parameter of WNNs. Orthogonal least squares (OLS) is used to optimize weights w of WNNs. Finally Levenberg–Marquardt Algorithm (LMA) is used to train the WNN to speed up the training process. The performance of the proposed approach has been evaluated on cases of one and two dimensions. The results show that the function approximation using GAs to optimize WNN parameters can achieve better normalized-root-mean-square-error than those achieved by traditional algorithms that use RBFN.

***Keywords:*** *Wavelet Neural Networks, Genetic Algorithms and Function Approximation.*

.

## 1. Introduction

The need for function approximations arises in many branches of applied mathematics, computer science and engineering communities in particular [1]. Approximating multi-dimensional functions by neural networks (NN) is a convenient way of representation, because learning in NN corresponds to the approximation of an underlying function due to the built-in capacity to adapt synaptic weights to changes in the surrounding environment [2]. Function Approximation consists of synthesizing a complete model from samples of the function and its independent variables [3]. In supervised learning, the task is to map from one vector space to another with the learning based on a set of instances of such mappings. We assume that a function F does exist and we endeavor to synthesize a computational model of that function. As a general mathematical problem, function approximation has been studied for centuries. For example, in pattern recognition, a function mapping is made whose objective is to assign each pattern in a feature space to a specific label in a class space [4, 5].

Most of reinforcement-learning algorithms consist in evaluating a value function that estimates the outcome of acting from a particular state. When the system to be controlled can be in a very large number of states, this value function has to be estimated by a generalizing function approximation [2]. Universal approximation only does not generalize well to higher dimensional spaces, well-known Wavelets are convenient tools when dealing with input spaces with up to multi-dimensions [2]. It is known that the functions can be represented as a sum of orthogonal basis functions. Such extensions can be easily represented as neural networks having the basis function selected as activation functions in every node, and the coefficients of extension as the weight of the output. Several classic orthogonal functions suffer, therefore, from the disadvantages of the approximation that uses global functions. What is necessary is a set of local basis functions and orthogonal [7]. Special class of functions, known as wavelet, possesses properties of good localization and they are orthonormal basis. Although, wavelet can be used as activation functions of a neuronal network, called wavelet neuronal network (WNN) [8]. The idea of using wavelets in neuronal networks has been proposed recently by Zhang and Benveniste [9] and Pati and Krishnaprasad [10].

Different neural networks such as multilayer perceptrons (MLP) and radial basis function neural networks (RBFN) have been used to solve the problem of function approximation. Wavelet Neural Networks (WNN) has recently attracted great interest, because they are universal approximations, it's achieved faster convergence than Radial Basis Function Neural Networks (RBFN) and is capable to deal with the problems of "curse of

dimensionality" [24]. In addition, WNN are generalized RBFN [6]. Due to the above advantages of wavelets over other basis functions it used as basis functions.

The idea of combining genetic algorithms and neural networks occurred initially at the end of the 1980s. The problem of neural networks is that the number of parameters has to be determined before any training begins and there is no clear rule to optimize them, even though these parameters determine the success of the training process [11, 28]. Genetic algorithms (GAs) [24], on the other hand, are very robust and explore the search space more uniformly, is a directed random search technique that is widely applied in optimization problems, since every individual is evaluated independently, which makes GAs very suitable to the optimization of Neural Networks. However, the choice of the basic parameters (network topology, initial weights) often determines the success of the training process. The selection of these parameters is practically determined by accepted rules of thumb, but their value is at most arguable. GAs are global search methods, that are based on the principles of selection, crossover and mutation [11]. GAs increasingly have been applied to the design of neural networks in several ways, such as optimization of the topology of neural networks by optimizing the number of hidden layers and the number of nodes in each hidden layer, and the optimization of neural network parameters by optimizing the weights [12, 13].

The structure of the WNNs is similar to the RBFNs, except the radial basis function is replaced with orthonormal basis functions. The efficiency of this type of networks is in learning of the function and its evaluation [10]. Wavelet Neural Networks use a three-layer structure and wavelet activation functions in hidden layer. The structure of a WNN with output *F(x)*, inputs p $\{x_1,x_2,\ldots,x_p\}$ and *m* mother wavelet (Neurones) as given in Figure.1

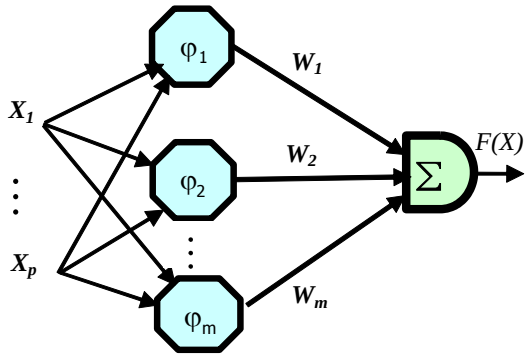



Fig. 1. Wavelet Neural Network

The WNNs possesses a unique attribute, besides the formation of an orthogonal basis they are also capable of explicit and represent the behaviour of a function in several resolutions of input variables [7]. The fundamental concept in the formulation and the design of WNN as basis function is the representation of multi-resolution of functions that use wavelet. This provides the essential frame for completely learning for WNN [7]. The wavelets are a family of functions where each one comes defined by one parameter of dilation $A_j$ that controls the scaling parameter and translation $B_j$ which controls the position of a single function, called mother wavelet $\varphi(x)$ [14]. The position of functions in a space of input data make that a WNN can reflect the properties of the function more exactly than the RBFNs. Considering a set of learning of n-elements, The output of the WNN is given by the following expression:

$$F(\vec{x},\varphi,w) = \sum_{j=1}^{m} w_j \cdot \varphi_j \left( \frac{\vec{x} - B_j}{A_j} \right) \quad (1)$$

where *m* the number of the mother wavelet nodes in the hidden layer, $w_j$ the weight connecting the $j_{th}$ unit of the hidden layer to the output layer unit. $\varphi(x)$ is the wavelet activation function (mother wavelet) of $j_{th}$ unit of the hidden layer. In terms of wavelet transformation theory, mother wavelets given by the following form:

$$\varphi = \left|A_j\right|^{-1/2} \varphi\left( \frac{\vec{x} - B_j}{A_j} \right) \quad (2)$$

Where $\varphi(\vec{x}) = \{x_1, x_2, \ldots, x_p\}$ , $A_j = \{A_{j1}, A_{j2}, \ldots, A_{jp}\}$ and $B_j = \{B_{j1}, B_{j2}, \ldots, B_{jp}\}$ are a family of functions generated from one single function $\varphi(x)$ . Note that for the *p*-dimensional input space, the mother wavelet can be calculated by the product of *p* single mother wavelets as follow [20]:

$$\varphi(\vec{x}) = \prod_{i=1}^{p} \varphi(\vec{x}_i) \quad (3)$$

A WNN can be considered as a function approximation which estimates an unknown functional using the following equation:

$$y = F(\vec{x},\varphi,w) + \varepsilon \quad (4)$$

where *F* is the regression function and $\varepsilon$ is error term is a zero-mean random variable of disturbance [14].

The localization of the $j_{th}$ units of the hidden layer is determined by the scale parameter *Aj* and the translation parameter *Bj*: these parameters are modified as well as the weight parameters during the process of training. This flexibility simplifies more reliability towards optimum learning solution. According to other models derived from traditional algorithms, the two parameters *Aj*, *Bj* can either be predetermined based upon the wavelet transformation theory or be determined by a learning algorithms [15]. Note that the above WNN is a kind of basis function neural network in the sense of that the wavelet consists of the basis functions [15].

The Topology optimization is a common learning method for WNNs, but a big challenge is optimization that includes the full parameter sets of scale parameter Aj and the translation parameter Bj and weights w along with the number of neurons per hidden layer. There are several possibilities of using GAs to optimize the WNN parameters and topology.

In [16] the author proposed new method for the design of WNNs in evolutionary way, by adding neurons one at a time, which allowed us to construct a parsimonious model to satisfy the design requirement. GA was used to select a wavelet basis, and the fitness of a wavelet is evaluated according to the residue reduction. In [17] the author proposed algorithm combines WNN with GA in order to optimize (case study intrusion detection) network parameters and improve network performance. All parameters of the WNN (*Aj* and the translation parameter *Bj* and weights *w* along with the number of neurons per hidden layer) optimize by GA. In [18] the author improved WNN to design a fault pattern recognition device. The training of the proposed network is performed through combining GA and gradient descent algorithm, where the initial parameters of the network are gained by the GA and the gradient descent algorithm is employed for local search procedure to find an optimal solution. In [19] the author presents an iterative method which combining the strength of back-propagation (BP) in weight learning and genetic algorithms' capability of searching the satisfying solution is proposed for optimizing wavelet neural networks (WNNs). these parameters are initialized to interval 0 and 1, GAs are used for optimal selection of them before the training procedure, parameters of the wavelet networks are trained during learning phase for approximation using gradient-descent method. System modelling consist, Input variables and wavelet network structure are determined; Initial values of parameters are randomly selected and these parameters are encoded into the chromosome in GAs; are optimized by using GAs, and the satisfying solution is obtained; Parameters are trained using gradient descent. In [20] genetic algorithm is proposed to design WNNs for nonlinear system identification. The model structure of a high dimensional system is decomposed into some sub-models of low dimensions. By introducing a connection switch to each link between a wavelet and an input node, the decomposition is done automatically during the evolutionary process. GA is used to train the wavelet parameters and the connection switches. In this way, both the structure and wavelet parameters of WNNs can be optimized simultaneously. In [21] the author propose the composition method of the activation function in the hidden layer with the scaling function which can represent the region where the several wavelet functions can represent, this can decrease the size of the network with a few wavelet functions. In addition to determine the parameters of the scaling function we can process rough approximation and then the network becomes more stable. The other wavelets can be determined by the global solution, the genetic algorithm which is suitable for the suggested problem and also, they use the back-propagation algorithm in the learning of the weights. In the proposed VTWNN in [22], wavelets are used as the transfer functions in the hidden layer. The network parameters, the translation parameters of the wavelets in the hidden layer, are variable depending on the network inputs. VTWNN has the ability to model the input-output function with variable network parameters. It works as if several individual neural networks are handling different groups of input data set. Effectively, it becomes an adaptive network capable of handling different input patterns, which exhibits a better performance. All network parameters are trained by real-coded GA with uni-modal normal distribution crossover and non-uniform mutation.

In our approach we present a different way that depends on optimizing the topology of WNNs parameters scale parameter *Aj* and translation parameter *Bj* by GA. Weights *w* are calculated by means of methods of resolution of linear equations. In this proposed approach we use the Orthogonal least squares (*OLS*) [29, 30] to solve this system of linear equations and assign the weights w for WNN to calculate the output. Each individual is an entire set of chromosomes cooperate to constitute a WNN. In our proposed approach we use the incremental method to determine the number of mother wavelets (neurons) depending on the data-test-error that the system produces which means an increase in each iteration will be only one mother wavelet until there is no improvement in test error during several iterations.

The organization of the rest of this paper is as follow: Section 2 presents an overview of the proposed approach. In Section 3, we present in detail the proposed approach for the determination of the pseudo-optimal WNN parameters. Then, in Section 4 we show some results that

confirm the performance of the proposed approach. Some final conclusions are drawn in Section 5.

## 2. General Description of the Proposed Approach

As mentioned before, the problem of function approximation consists of synthesizing a complete model from samples of the function and it is independent variables. Consider a function $y = F(\vec{x}, \varphi, w)$ where $\vec{x}$ is a vector $(x_1, \ldots, x_p)$ in k-dimensional space from which a set of input/output data pairs is available. The process of combining WNNs and GA is based on the using of GA to optimize the WNNs parameters (scale parameter *Aj* and translation parameter *Bj*) so that the neuron is put in a suitable place in input data space [17]. The process begins with an initial population generated randomly. This population consists of two values scale parameter *Aj* and translation parameter *Bj*. Weights *w* are estimated by means of methods of resolution of linear equations. In this proposed approach we use the orthogonal least squares (OLS) [23 29] to solve this system of linear equations and assign the weights w of the direct connections of the WNN to calculate the output, this algorithm is a simple and efficient algorithm for fitting the WNN. It also has the capability to select smaller weight and to create a parsimonious network model. OLS algorithm purifies the wavelets from their candidates, which is avoided using more wavelets than required and often resulted in an over-fitting of the data and a poor situation [23]. So that, each individual is an entire set of chromosomes cooperate to constitute a WNN. In our proposed approach we use the incremental method to determine the number of mother wavelets (neurons) depending on the data-test-error that the system produces which means, an increase in each iteration will be only one mother wavelet until there is no improvement in test error during several iterations.

The fitness function (NRMSE) that is used to evaluate the population will establish the fitness for every chromosome depending on its functions in the training set. The best population will be selected for promotion to the next generation, where the genetic operators of crossover and mutation produce a new population. The population leads the process of the selection to the best value of the fitness (small error). Crossover and mutation lead to exploring the unknown regions of the search space. Then, the population converges to the best parameters of optimization of scale parameter *Aj*, translation parameter *Bj* and weights *w*. The process repeats until it finds the best fitness or until the generation number reaches the maximum with the same genetic operators in every generation.

## 3. Optimization of WNN Parameters Using GAs

A GA is a search or an optimization algorithm, which is invented based on genetics and evolution. The initial population of individuals that have a digit string as the chromosome is usually generated randomly. Each element of a chromosome is called a gene. The fitness, which is a measure of improvement of approximation, is calculated for each individual. The selection operations choose the best individuals for the next generation depending on the fitness value. Then, crossover and mutation are performed on the selected individuals to create a new individual that replaces the worst members of the population offspring. These procedures are continued until the end-condition is satisfied. This algorithm confirms the mechanism of evolution, in which the genetic information changes for every generation, and the individuals that better adapt to their environment survives preferentially [24, 25].

Our new proposed approach use GAs to construct optimal WNN. The approach uses GAs evolving to optimize the two WNN parameters (scale parameter *Aj*, translation parameter *Bj*) and uses orthogonal least squares (OLS) to optimize directly the weights *w*. The general process of our proposed approach can be depicted in Figure 2, and the pseudo-code of this algorithm reads:

**Begin**
*Initialize* population Randomly
*Evaluate* each individual on population *P* by fitness function $F(x, \varphi, w)$;
While not (*stop criteria*) ([threshold of NRMSE] || [number of Generation β]) **do**
*Select* individual's $I_1$ and $I_2$;
$I_{p+1} \leftarrow$ *Crossover*($I_1$, $I_2$);
*Mutation* ($I_{p+1}$);
*Evaluate* ($I_{p+1}$);
if matches threshold → *stop*
else insert($I_{p+1}$, $P_{new}$);
**End;**

### 3.1 Initialization

Each gene is constituted by a real vector representing translation parameter *Bj* and, and a real value representing scale parameter *Aj* of mother wavelets *m*. Chromosomes have a variable length which defined as follow:

$$chrom = \left[ \{B_{1m}, A_{1m}\}, \{B_{2m}, A_{2m}\} \ldots \{B_{im}, A_{im}\} \right] \quad (5)$$

In our approach the chromosome that consists of (translation parameter *Bj*, scale parameter *Aj*) is generated

initially randomly. The number of parameters in each chromosome calculated by [(# of mother wavelets translation parameter × # of dimensions) + # of scale parameter]. Orthogonal least squares (OLS) are used directly to optimize the weights.

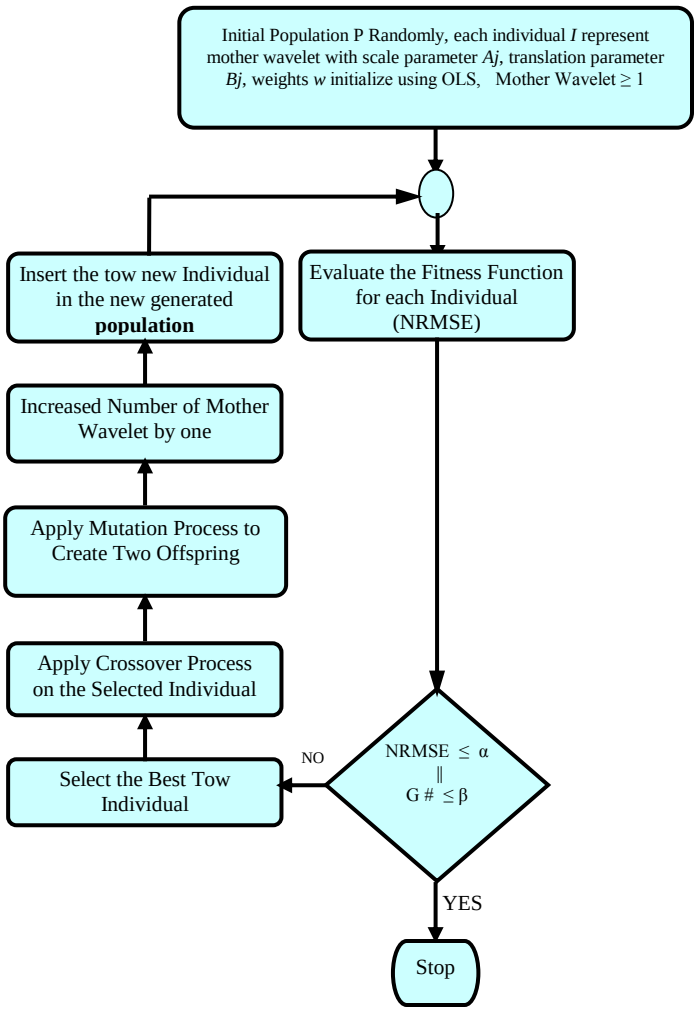



Fig. 2. General description of the proposed approach

## 3.2 The Evaluation Function

The evaluation function is the function that calculates the value of the fitness in each chromosome, in our case, the fitness function is the error between the target output and the current output, (Fitness = error). In this paper, the fitness function we are going to use is the so-called Normalized-Root-Mean-Squared-Error (NRMSE). This performance-index is defined as:

$$NRMSE = \sqrt{\sum_{i=1}^{P}(y_i - F(\vec{x}, \varphi, w))^2 / \sum_{i=1}^{P}(y_i - \bar{y})^2} \qquad (6)$$

Where $\bar{y}$ the mean of the target output, and p is is the input data number.

## 3.3 Stop Process

A GA evolves from generation to generation selecting and reproducing parents until reaching the end criterion. The criterion that is most used to stop the algorithm is a stated maximum number of generations. With this work we use the maximum number of generation β or the value of the fitness (NRMSE) threshold α as the criterion of End. This finishes the process when the fitness (NRMSE) value reaches the determined threshold value α or when the maximum number of performed generations exceeds the determined number of generations. In practice, however, the process of optimization can finish before approaching the termination conditions, which can happen when a GA moves from generation to generation without resulting in any improvement in the value of the fitness.

***If Current Generation ≥ Maximum Generation β || Fitness (NRMSE) ≤ Threshold value α → End the optimization***

## 3.4 Selection

The selection of the individuals to produce the consecutive generation is an important role in genetic algorithms. The probable selection arises the fitness of each individual. This fitness presents the error between the objective output and actual output of WNN, such that the individual that produces the smallest error has higher possibility to be selected. An individual in the population can be selected once in conjunction with all the individuals in the population who has a possibility of being selected to produce the next generation. There are many methods that are used for the process of the selection as: roulette wheel selection, geometric ranking method, and rank selection… etc [26, 27]. The most common selection method depends on assignment of a probability $p_j$ to every individual $j$ based on its value of fitness. A series of numbers $N$ is generated and compared against the accumulative probability $C_i = \sum_{j=1}^{i} P_j$ , of the population. The appropriate individual $j$, is selected and copied in the new population if $C_{i-1} < U(0,1) \leq C_i$ . In our work we use a Geometric Ranking method; in this method the function of the evaluation determines the solution with a partially

ordered set. By this we guarantee the minimization and the negative reaction of the geometric method of classification. It works by assigning $P_i$ based on the line of the solution $i$ when all solutions are classified. In this method the probability $P_i$ of the definite classification is calculated as in the following expressions [26, 27]:

$$P\left[\text{individual selection-i}\right]=q^{+}(1-q)^{s-1} \quad (7)$$

where $P$ is the size of population.

$$q^{+} = \frac{q}{1-(1-q)^{P}} \quad (8)$$

where $q$ is the probability of selecting the best individual. $s$ is the line of the individual, where one is the best.

### 3.5 Crossover and Mutation

Crossover and mutation provide the basic search mechanism of a GA. The operators create new solutions based on the previous solutions created in the population. Crossover takes two individuals and produces two new recombinant individuals, whereas the mutation changes the individual by random alteration in a gene to produce a new solution. The use of these two basic types of genetic operators and their derivatives depends on the representation of the chromosome. For the real values that we use in our work, we use the arithmetical crossover, which produces two linear combinations of the parents (two new individuals) as in the following equations:

$$\overline{X}^{!} = r\,\overline{X} + (1-r)\,\overline{Y} \quad (9)$$

$$\overline{Y}^{!} = (1-r)\,\overline{X} + r\,\overline{Y} \quad (10)$$

Where $\bar{X}$ *and* $\bar{Y}$ are two vectors of k-dimensional that denote to individuals (parents) of the population and r is the probability of crossover between (0, 1) in this work probability of crossover r = 0. 5. From these equations we can present the process of the arithmetic crossover as shown in Figure 3.

We can find many methods of mutation in [27], such as uniform mutation, non-uniform mutation (odd number - uniform mutation), and multi-non-uniform mutation. In our work we use the process of uniform mutation that changes one of the parameters of the parent. The uniform mutation selects one j element randomly and makes it equal to a uniform selected number inside the interval. The equation that presents the uniform mutation is shown in equation (Eq. 11):

$$x_i^{'} = \begin{cases} U(a_i, b_i) & if\ i = j \\ x_i & otherwise \end{cases} \quad (11)$$

Where $a_i$ and $b_i$ are down and top level, for every variable $i$. Figure 4 present the process of mutation that appears among the parameters of the WNN.

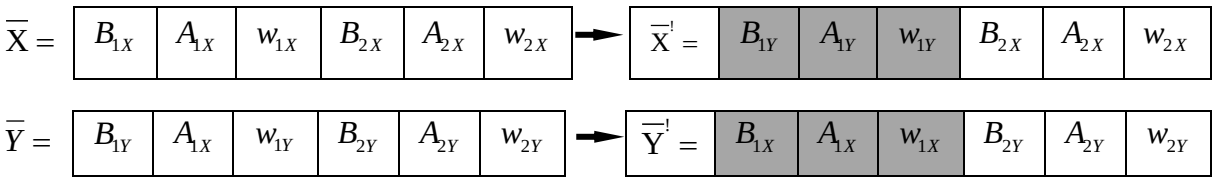



Figure.3. The process of the arithmetic crossover of three points in two neurons mother wavelet

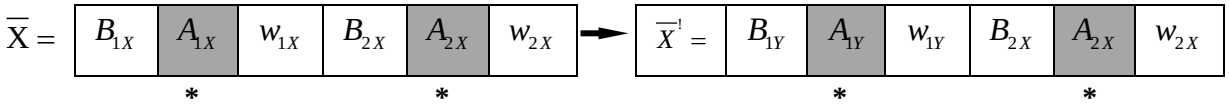



Figure.4. The uniform mutation of two points in two neurons mother wavelet

## 4. Simulation Examples

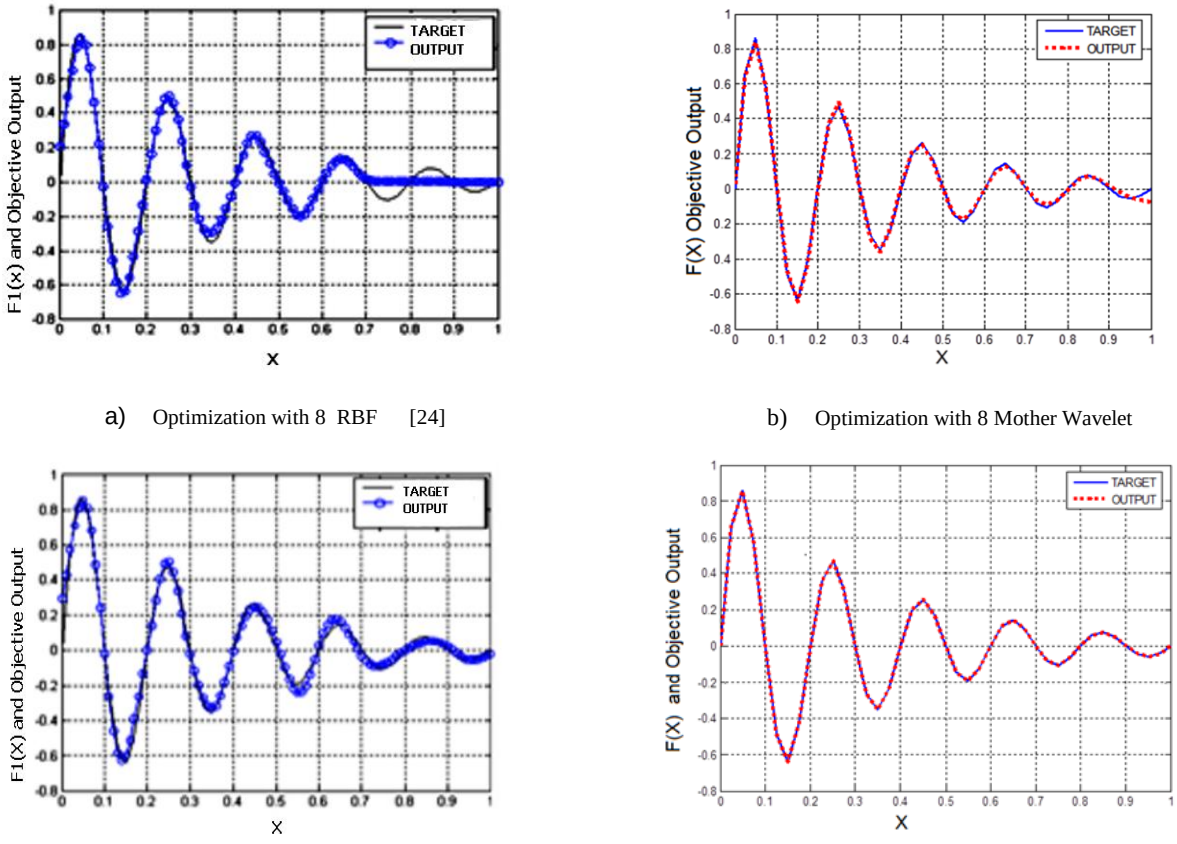
Experiments have been performed to test the proposed approach. The system is simulated in MATLAB 7.3 under Windows XP with a Pentium IV processor running at 2.8 GHz. In this section, different examples are given to verify the procedure in the proposed approach. We will compare the result of our approach that depends on WNN with the results of other algorithms that approximate functions use GAs to optimize RBFNs parameters as in [24, 31, 32]. Two types of results are presented: The results of the validity of the algorithm in approximate functions from samples of I/O data of one dimension compared with other algorithms as [24, 31, 32], and the approximation of function in two dimensions with the NRMSE and execution time compared with traditional algorithm [24]. The results are obtained in five executions. $NRMSE_{Test}$ is the mean of normalized mean squared error of the test index (for 1000 test data). For the GAs parameters we used. The population-size = 100, crossover rate = 0.5 and mutation rate = 0.05.

### 4.1 One Dimension Example F(x)

To test the effects caused by the proposed approach on initialization and avoiding local minimum of mother wavelet placement. Training set of 2000 samples of the function was generated by evaluating inputs taken uniformly from the interval [0, 1], from which we have removed 1000 points for validation. This function is defined by the following expression:

$$F(x) = e^{-3x} \sin(10\pi x), \qquad x \in [0,1] \qquad (12)$$

Figure 5 presents different result of approximation of the function $F_1(x)$ comparing with the result of optimizing RBFN parameters using GAs [24]. It's clear that the approximation depending on the proposed approach *(Figure 5 b,d)* is better than the approximation depending on RBFN using GAs *(Figure 5 a,c)* with the same number of neurons.




a) Optimization with 8 RBF [24]

b) Optimization with 8 Mother Wavelet

c) Optimization with 10 RBF [24]

d) Optimization with 10 Mother Wavelet

Fig. 5. Approximation of the function F(x) comparing with RBF.

The $NRMSE_T$ of the points predicted by the proposed approach is shown in Table 1. It is clear that the proposed approach minimizes the approximation error with much accuracy than other algorithms. Approximation result shows that the proposed approach use WNN produce better result than RBFN optimized by GAs [24].

Table. 1. Comparison Result of NRMSETEST Error of different approach

| Method | # RBF | $NRMSE_{Test}$ |
|---|---|---|
| González [32] | 5 | 01771 |
| | 6 | 0.1516 |
| | 8 | 0.0674 |
| | 10 | 0.0882 |
| Rivas [31] | 4 ± 7 | 0.7 ± 0.2 Generation = 10 |
| | 5 ± 6 | 0.7 ± 0.2 Generation = 25 |
| | 8 ± 9 | 0.6 ± 0.3 Generation = 50 |
| | 23 ± 7 | 0.2 ± 0.3 Generation = 75 |
| | 22 ± 11 | 0.4 ± 0.3 Generation = 100 |
| Awad [24] | 2 | 0.059 Generation = 50 |
| | 4 | 0.0485 Generation = 50 |
| | 6 | 0.0274 Generation = 50 |
| | 8 | 0.0205 Generation = 50 |
| | 10 | 0.0223 Generation = 50 |
| Proposed Approach | 2 | 0.0497 Generation = 50 |
| | 4 | 0.0319 Generation = 50 |
| | 6 | 0.0182 Generation = 50 |
| | 8 | 0.0137 Generation = 50 |
| | 10 | 0.0078 Generation = 50 |

## 4.2 Two Dimension Examples $F1(x_1,x_2)$

In this part we used functions of two-dimensions (see Figure 6, Figure 8). These functions of two-dimension use a set of training data formed by 441 points distributed as *21 x 21* cells in the input space. These examples of two dimensions are used to demonstrate the ability of the proposed approach in approximating two dimension examples. In this example we use number of Generations =250.

Figure 7 presents different result of approximation of the function $F_1(x_1, x_2)$ produced by the proposed approach, comparing with the result of optimizing RBFN parameters using GAs [24]. It's clear that the approximation depending on the proposed approach *(Figure 7 b,d)* is better than the approximation depending on RBFN using GAs *(Figure 7 a,c)* with the same number of neurons, and the improvement of Fitness function ($NRMSE_T$) with the increased generation numbers.

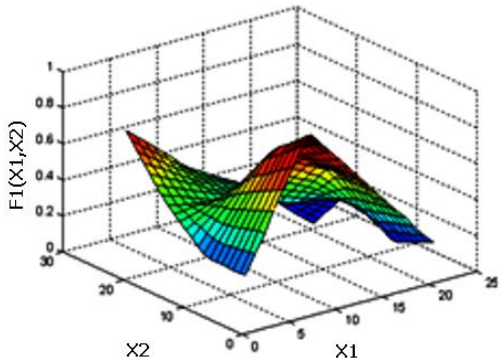


Fig. 6. Objective function $F1(x_1,x_2)$

In Table 2, $NRMSE_T$ is the final mean Error after 5 time of execution of the proposed approach comparing with the result of optimizing RBFN parameters using GAs [24].

## 4.3 Two Dimension Example $F_2(x_1,x_2)$

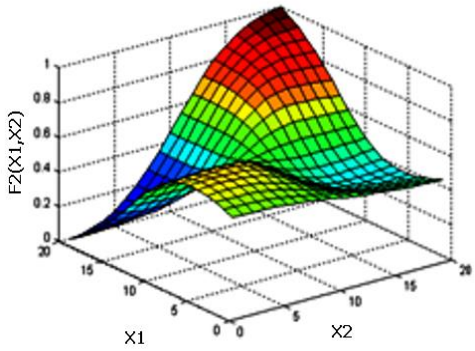
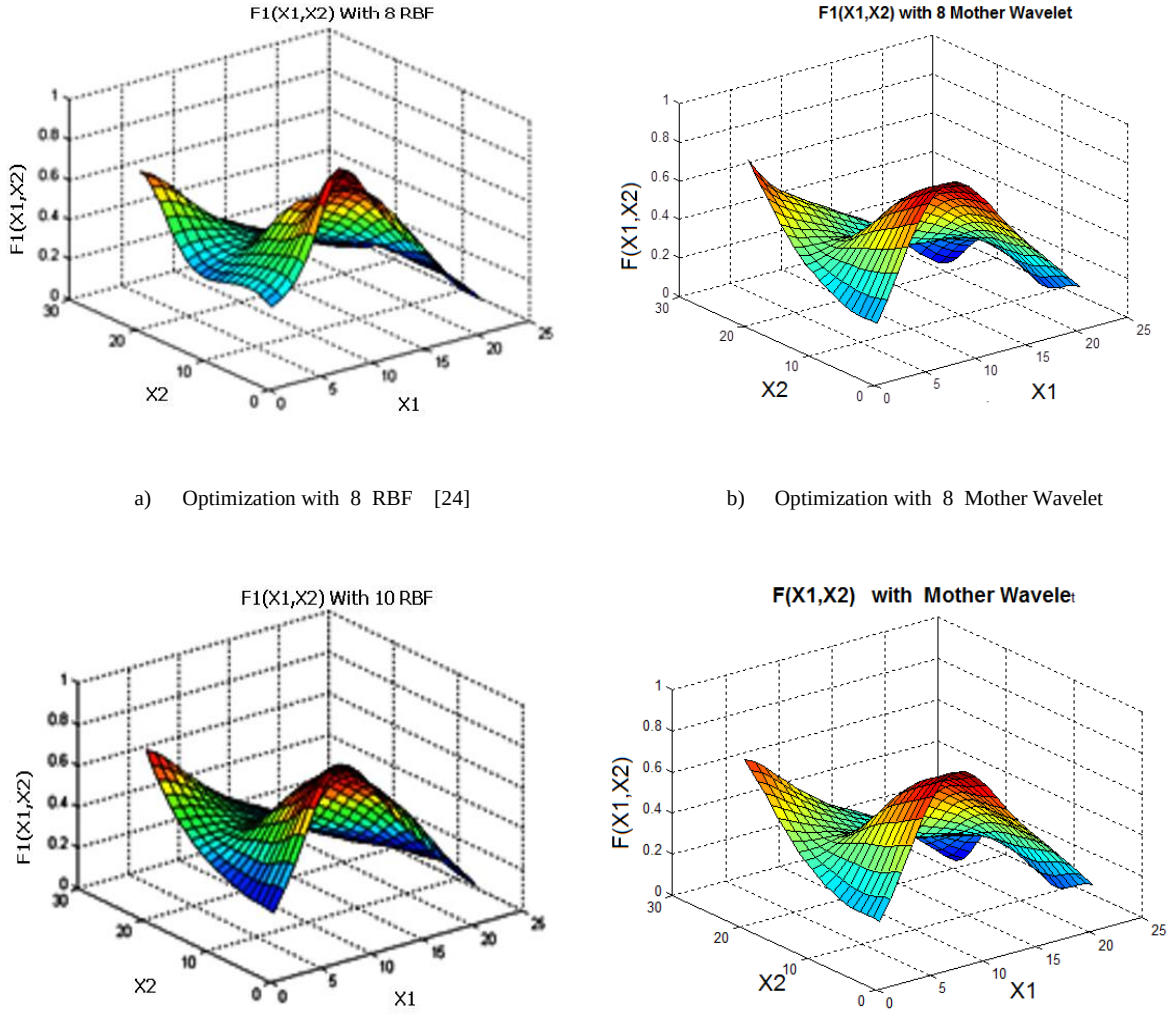


Fig. 8. Objective function $F1(x_1,x_2)$

Figure 9 presents different result of approximation of the function $F_1(x_1, x_2)$ produced by the proposed approach, comparing with the result of optimizing RBFN parameters using GAs [24]. It's clear that the approximation depending on the proposed approach *(Figure 9 b,d)* is better than the approximation depending on RBFN using GAs *(Figure 9 a,c)* with the same number of neurons, and the improvement of Fitness function ($NRMSE_T$) with the increased generation numbers.

In Table 3, $NRMSE_T$ is the final mean Error after 5 time of execution of the proposed approach comparing with the result of optimizing RBFN parameters using GAs [24].




a) Optimization with 8 RBF [24]

b) Optimization with 8 Mother Wavelet

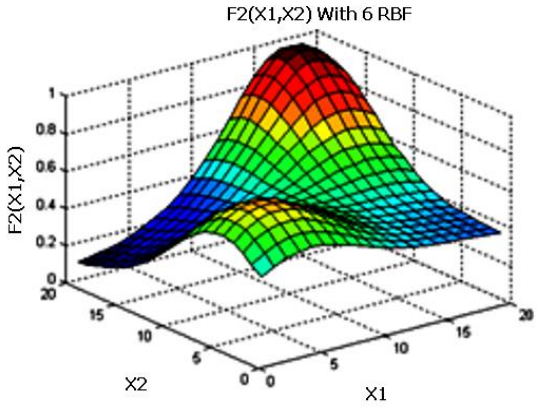
c) Optimization with 10 RBF [24]

d) Optimization with 10 Mother Wavelet

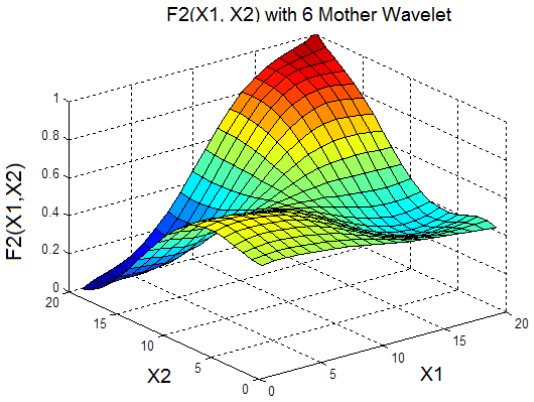
Fig. 7. Approximation of the function $F1(x_1,x_2)$ comparing with RBF

Table.2. Result of $NRMSE_{TEST}$ and Execution Time of the proposed approach applied on 2D Function $F1(x_1,x_2)$

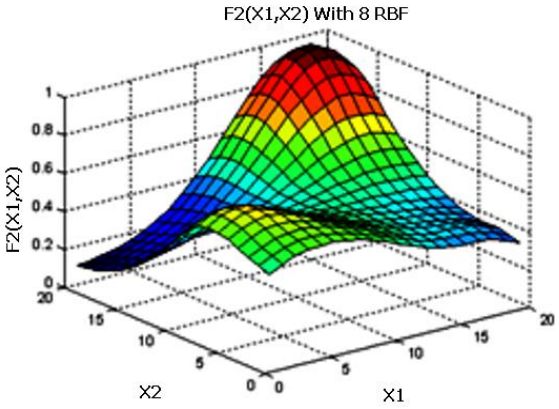
| N° Mother Wavelet | NRMSE | Execution Time (sec) | | | N° RBF | NRMSE | Execution Time (sec) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Mean | Max | Min | Mean | | Mean | Max | Min | Mean |
| **2** | 0.206 | 142 | 132 | 137 | **2** | 0.224 | 130 | 122 | 127 |
| **4** | 0.149 | 153 | 141 | 148 | **4** | 0.176 | 164 | 144 | 156 |
| **6** | 0.108 | 197 | 152 | 173 | **6** | 0.124 | 169 | 147 | 157 |
| **8** | 0.0637 | 215 | 184 | 196 | **8** | 0.115 | 192 | 181 | 186 |
| **10** | 0.0513 | 236 | 193 | 207 | **10** | 0.27 | 203 | 184 | 192 |



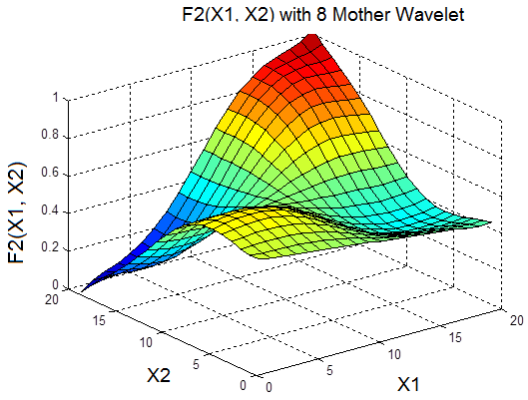

a) Optimization with 6 RBF [24]




b) Optimization with 6 Mother Wavelet




c) Optimization with 8 RBF [24]




d) Optimization with 8 Mother Wavelet

Fig. 9. Approximation of the function $F1(x_1,x_2)$ comparing with RBF

Table.3. Result of $NRMSE_{TEST}$ and Execution Time of the proposed approach applied on 2D Function $F2(x_1,x_2)$

| Nº Mother Wavelet | NRMSE | Execution Time (sec) | | | Nº RBF | NRMSE | Execution Time (sec) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Mean | Max | Min | Mean | | Mean | Max | Min | Mean |
| 2 | 0.224 | 130 | 122 | 127 | 2 | 0.53 | 122 | 112 | 117 |
| 4 | 0.176 | 164 | 144 | 156 | 4 | 0.37 | 132 | 121 | 127 |
| 6 | 0.124 | 169 | 147 | 157 | 6 | 0.28 | 169 | 147 | 158 |
| 8 | 0.115 | 192 | 181 | 186 | 8 | 0.22 | 188 | 175 | 178 |

## 5. Conclusion

In our paper an efficient way of applying GA to WNNs configuration has been presented. The approach optimizes translation parameter *Bj*, scale parameter *Aj* of WNN using GAs. The weights w are optimized by using Orthogonal least squares (OLS). The initialization of the translation parameter *Bj* and scale parameter *Aj* of WNN depends on random way. This approach was compared with other traditional approaches and with the approach that optimize RBFNs parameters using GAs. The proposed approach is accurate as the best of the others approaches and with significantly less number of mother wavelet in all experiments. Simulations have demonstrated that the approach can produce more accurate prediction. This approach is easy to implement and is superior in both performance and computation time compared to other algorithms. Normally, GAs took a long training time to achieve results, but in the proposed approach the time taken is suitable and that because of the WNN converge fast than RBFNs. We have also shown that it is possible to use this approach to find the minimal number of mother wavelet (Neurones) that satisfy a certain error target for a given function approximation problems.

## 6. References

[1] M. J. D. Powell. "The Theory of Radial Basis Functions Approximation, in Advances of Numerical Analysis". pp. 105–210, Oxford: Clarendon Press, 1992.

[2] I. Valova, N.Gueorguieva, G. Georgiev. "Function Approximation Through Growing Neural Network Based On RBF And Potential Function". 3rd International Conference on Neural, Parallel and Scientific Computations Morehouse College, IFNA, Atlanta, GA, USA. August 9-12, 2006

[3] Z. Zainuddin O. Pauline. "Function approximation using artificial neural networks". 12th WSEAS International Conference on Applied Mathematics, 2007 Cairo, Egypt pp: 140-145.

[4] Gen .M, Cheng .R. "Genetic algorithms and Engineering Optimization". A Wiley-Interscience Publication, Johan Wiley and Sons, Inc. 2000.

[5] Sheng. Li and Shu-Ching. Ch., "Function Approximation using Robust Wavelet Neural Networks," ICTAI '02, pp, 483-488, 2002.

[6] Vachtsevanos. G and Wang. P., "A Wavelet Network Framework for Diagnostics of Complex Engineered Systems," Proceedings of MARCON 98 Maintenance and Reliability Conference, Knoxville, TN, invited paper. May, 1998.

[7] Wallich. P., "Wavelet theory: An analysis technique that's creating ripples," Scientific American, pp. 34-35. Jan. 1991.

[8] Zhang. Q and Benveniste.A., "Wavelet networks," IEEE Transactions on Neural Networks, vol. 3, no. 6, pp. 889-898, Nov. 1992.

[9] Krishnaprasad. P. S. and Pati. Y.C., "Analysis and Synthesis of Feed forward Neural Networks Using Discrete Affine Wavelet Transformations," IEEE Trans. on Neural Networks, vol. 4, pp,73-85 Jan, 1993.

[10] Ph. Koehn. "Combining Genetic Algorithms and Neural Networks". Master Thesis University of Tennessee, Knoxville, December 1994.

[11] D. Schaffer, D. Whitley and L.J. Eshelman, "Combinations of genetic algorithms and neural networks". A survey of the state of the art, in Combinations of Genetic Algorithms and Neural Networks, pp. 1-37, IEEE Computer Society Press, 1992.

[12] D. Prados. "A fast supervised learning algorithm for large multilayered neural networks". in Proceedings of 1993 IEEE International Conference on Neural Networks, San Francisco, v.2, pp.778-782, 1993.

[13] P. P. Kanjilal and D. N. Banerjee, "On the application of orthogonal transformation for the design and analysis of feedforward networks," IEEE Trans. Neural Networks, vol. 6, pp. 1061–1070, 1995.

[14] Chen. Y, Yang. B, and Dong. J., "Time-series prediction using a local linear wavelet neural network," Neurocomputing. v69, pp. 449-465. Jan. 2006.

[15] J. Xu: A Genetic Algorithm for Constructing Wavelet Neural Networks. 286-291 ICIC (1) 2006:

[16] Ch. Ning, H. Yujing, L Huifang, R Hui. A Study on GA-based WWN Intrusion Detection. Management and Service Science, 2009. MASS '09. International Conference on 20-22 Sept. 2009

[17] L. Jinru, L. Yibing, Y. Keguo. "Fault Diagnosis of Piston Compressor Based on Wavelet Neural Network and Genetic Algorithm". 7th World Congress on Intelligent Control and Automation. WCICA. 25-27 June 2008

[18] Feng Lu, Jianhua Xu, Zhanyong Wang, "Application of GA Optimized Wavelet Neural Networks for Carrying Capacity of Water Resources Prediction," esiat, vol. 1, pp.308-311, 2009 International Conference on Environmental Science and Information Application Technology, 2009.

[19] Jinhua Xu: GA-Optimized Wavelet Neural Networks for System Identification. 214-217. ICICIC (1) 2006:

[20] K. Seong-Joo, K. Taek-Yong, S. Jae-yong, J. Hong-Tae, Design of the scaling-wavelet neural network using genetic algorithm, in: Proceedings of the International Joint Conference on Neural Networks, Honolulu, HI, 2002, pp. 2174-2179.

[21] S.H. Ling, F.H.F. Leung, Genetic algorithm-based variable translation wavelet neural network and its application, in: Proceeding of International Joint Conference on Neural Networks, Montreal, Canada, 2005, pp. 1365-1370.

[22] Billings, S.A., Wei, H. : A New Class of Wavelet Networks for Nonlinear System Identification. IEEE Trans. on Neural Networks, VOL.16, NO.4, (2005) 862-874

[23] M. Awad "Optimization of Radial Basis Function Neural Networks Parameters using Genetic Algorithms: Applied on Function Approximation. International Journal of Computer Science and Security (IJCSS), Volume (4): Issue (3). Pp.295-307. August 2010.

[24] T. Hatanaka, N. Kondo and K. Uosaki. "Multi–Objective Structure Selection for Radial Basis Function Networks Based on Genetic Algorithm". Department of Information and Physical Science Graduate School of Information Science and Technology, Osaka University 2–1 YamadaOka, Suita, 565–0871, Japan.
[25] P. T. Rodríguez-Piñero. "Introducción a los algoritmos genéticos y sus aplicaciones". Universidad Rey Juan Carlos, España, Madrid. (2003)
[26] Z. Michalewickz. Univ. of North Carolina, Charlotte "Genetic Algorithms + Data Structures = Evolution Programs". Springer-Verlag London, UK (1999).
[27] D. Schaffer, D. Whitley and L.J. Eshelman, "Combinations of genetic algorithms and neural networks". A survey of the state of the art, in Combinations of Genetic Algorithms and Neural Networks, pp. 1-37, IEEE Computer Society Press, 1992.
[28] S. Chen, Y. Wu, and B. L. Luk. "Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks". IEEE-NN, 10(5):1239, September 1999.
[29] Zhang Qinghua, "A Using Wavelet Network for Nonparametric Estimation," IEEE Trans. Neural Networks, Vol. 8, No.2, 1997, pp. 227-236.
[30] Rivas. A. "Diseño y optimización de redes de funciones de base radial mediante técnicas bioinspiradas". .PhD Thesis. University of Granada. 2003.
[31] González. J, "Identificación y optimización de redes de funciones de base radiales para aproximación funcional". PhD Thesis. University of Granada 2001

**Mohammed Awad** holds PhD degree in June 2005, from the University of Granada, Spain, Dept. of Computer Architecture and Computer Technology. Inside the Ph.D program: Computer Engineering: Perspectives and Applications, Master Degree in Computer Engineering in June 2003 from University of Granada and BSc degree in industrial automation Engineering in 2000, from the Palestine Polytechnic University. From June 2005 to December 2005 he was a contracture as researcher in Dept. of Computer Architecture and Computer Technology, University of Granada. From 12/2010 - Till now; he is Associate Professor in the Faculty of Engineering and Information Technology, Arab American University, Palestine. He is now the Vice Dean of Scientific Research. From 2/2006 – 12/2010; he is assistant professor in the Faculty of Engineering and Information Technology at the Arab American University, Palestine. Chairperson of the Department of Computer Information Technology (2006- 2010, Chairperson of the Department of Computer system Engineering 2010- 2012 at the Arab American University, Palestine. member of the AUUJ research committee 2009 - 2014. His current areas of research interest include artificial Neural Networks and evolutionary computation, function approximation using radial basis function neural networks, input variable selections.