Prediction of Time Series Using RBF Neural Networks: A New Approach of Clustering

Mohammed Awad[2], Héctor Pomares[1], Ignacio Rojas[1], Osama Salameh[2] and Mai Hamdon[2]
[1]Department of Computer Architecture and Technology, University of Granada, Spain
[2]Faculty of Information Technology, Arab American University, Palestine

Abstract: *In this paper, we deal with the problem of time series prediction from a given set of input/output data. This problem consists of the prediction of future values based on past and/or present data. We present a new method for prediction of time series data using radial basis functions. This approach is based on a new efficient method of clustering of the centers of the radial basis function neural network; it uses the error committed in every cluster using the real output of the radial basis function neural network trying to concentrate more clusters in those input regions where the error is bigger and move the clusters instead of just the input values of the I/O data. This method of clustering, improves the performance of the time series prediction system obtained, compared with other methods derived from traditional algorithms.*

Keywords: *Clustering, time series prediction, RBF neural networks.*

Received April 21, 2007; accepted December 14, 2007

1. Introduction

Time series is widely used in many aspects of our lives. Daily temperature, electrical load and river flood forecasting [21],…*etc.* The problem consists of predicting the next value of a series known up to a specific time, using the known past values of the series. Basically, time series prediction can be considered a modeling problem. The first step is establishing a mapping between inputs/outputs. Usually, the mapping is nonlinear and chaotic. After such a mapping is set up, future values are predicted based on past and current observations [16, 21].

Radial Basis Function Neural Networks (RBFNNs) are characterized by a transfer function in the hidden unit layer having radial symmetry with respect to a center [9]. The basic architecture of an RBFNN is a 3-layer network as in Figure 1.

The output of the net is given by the following expression:

$$F(\vec{x}, \Phi, w) = \sum_{i=1}^{m} \phi_i(\vec{x}) \cdot w_i \qquad (1)$$

where $\Phi = \{\phi_i : i = 1,...,m\}$ are the basis functions set and w_i the associate weights for every Radial Basis Function (RBF). The basis function ϕ can be calculated as a Gaussian function using the following expression:

$$\phi(\vec{x}, \vec{c}, r) = \exp\left(\frac{\|\vec{x} - \vec{c}\|}{r}\right) \qquad (2)$$

where $\vec{c}$ is the central point of the function ϕ, r is its radius and $\vec{x}$ is the input vector.

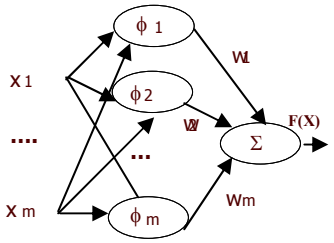

Figure 1. Radial basis function network.

A common learning method for RBFNNs is clustering. Every cluster has a centre, which can be chosen as the centre of a new RBF. The RBF centres can be obtained by many clustering algorithms. These algorithms are classified as unsupervised clustering algorithms such as k-means [6], fuzzy c-means [2], enhanced LBG [19], and supervised clustering algorithms such as the Clustering for Function Approximation method (CFA) [3], the Conditional Fuzzy Clustering algorithm (CFC) [11] and the Alternating Cluster Estimation method (ACE) [18]…*etc.* The clustering algorithm obtains the cluster centres by attempting to minimize the total squared error incurred in representing the data set by the m cluster centres. However, the clustering algorithm can only achieve a local optimal solution, which depends on the initial locations of cluster centres. A consequence of this local optimality is that some initial centres can become stuck in regions of the input domain with few or no input patterns. This wastes resources and results in a local optimal network.

RBFNNs are universal approximators and thus best suited for function approximation problems. In general an approximator is said to be universal if it can approximate any continuous function on a compact set to a desired degree of precision. The technique of finding the suitable number of radial functions is very complex since we must be careful of not producing excessively large networks which are inefficient and sensitive to over-fitting and exhibit poor performances. Figure 2 presents a functional approximation using several RBFs with different values of the radius *r*, where *r* is the radius of RBF.

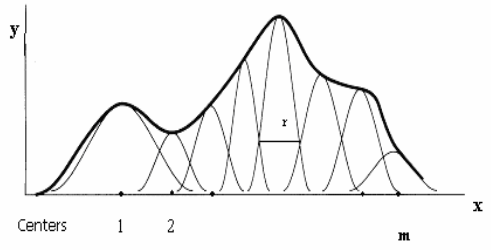

Figure 2. Function approximation using RBFNNs.

In this paper we present a new method of clustering the centres of RBFs for the prediction of time series and a new efficient clustering method for the initialization of the centres of the RBF network, this method uses the target output of the RBFN to migrate and fine-tune the clusters instead of just the input values of the I/O data. This method calculates the error committed in every cluster using the real output of the RBFN trying to concentrate more clusters in those input regions where the error is bigger, thus attempting to homogenize the contribution to the error of every cluster.

The organization of the rest of this paper is as follows. Section 2 presents an overview of the proposed algorithm. In section 3, we present in detail the proposed algorithm for the determination of the pseudo-optimal RBF parameters. Then, in section 4 we show some results that confirm the performance of the proposed methodology. Some final conclusions are drawn in section 5.

2. Proposed Approach

As mentioned before, the problem of time series prediction consists of the prediction of future values based on past and/ or present values. A time series is a sequence of vectors, $x(t)$, t = 0,1,…, where *t* represents elapsed time. Theoretically, *x* may be a value which varies continuously with *t*, such as temperature. In practice, for any given system, *x* will be sampled to give a series of discrete data points, equally spaced in time [7]. Formally this can be stated as: find a function $x(t+d)=f(x(t))$ such as to obtain an estimate of *x* at time *t* + *d*, from the *N* time steps back from time *t*, so that:

$$x(t+d)=f(x(t)),x(t-1),...,x(t-N+1) \quad (3)$$

The accuracy of the prediction process is measured by a cost function which takes into account the error between the output of the RBFNN and the real output. In this paper, the cost function we are going to use is the so-called Normalized/Root Mean Squared Error (N/RMSE). This performance index is defined as:

$$NRMSE=\sqrt{\sum_{i=1}^{P}(y_i-f(\vec{x}_i))^2/\sum_{i=1}^{P}(y_i-\bar{y})^2} \quad (4)$$

where $\bar{y}$ is the mean of the real output, and *p* is the data number. The objective of our algorithm is to increase the density of clusters in the input domain areas where the error committed in every cluster using the real output of the RBFN is bigger.

The RBFNN is completely specified by choosing the following parameters: the number *m* of radial basis functions, the centres $\vec{c}$ of every RBF, the radius *r*, and the weights *w*.

The number of RBFs is a critical choice. In our algorithm we have used a simple incremental method to determine the number of RBFs. We will stop adding new RBFs when the time series prediction error falls below a certain target error. As to the rest of the parameters of the RBFNN, in section 3 we present a new clustering technique. Figure 4 presents a flowchart with the general description of the proposed approach.

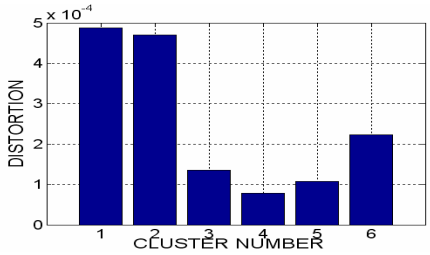

Figure 3. The distortion before the migration.

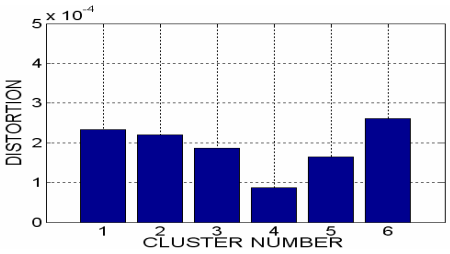

Figure 4. The distortion after the migration.

3. Parameter Adjustment of the RBFNN

The locality property inherent to the RBF allows us to use a clustering algorithm to obtain the RBF centres. Clustering algorithms may get stuck in a local minimum ignoring a better placement of some of the clusters, i.e., the algorithm is trapped in a local minimum which is not the global one. For this reason we need a clustering algorithm capable to solve this local minimum problem. To avoid this problem we

endow our supervised algorithm with a migration technique. This modification allows the algorithm to escape from local minimum and to obtain a prototype allocation independent of the initial configuration.

To optimize the other parameters of the RBFNN (the radius *r* and the weights *w*) we used well-known heuristics; the k-Nearest Neighbour technique (kNN) [10] for the initialization of the radius of each RBF, Singular Value Decomposition (SVD) [13] to directly optimize the weights. Finally, the levenberg-marquardt algorithm is to fine-tune the obtained RBFNN [8].

Therefore, in this section we will concentrate on the proposed clustering algorithm. In Figure 4, we show a flowchart representing the general description of our clustering algorithm. As can be seen from this figure, the initial values of the clusters are calculated using the *k*-means clustering algorithm followed by a local displacement process which locally minimizes the distortion (*D*) within each cluster.

In Figure 3 we can see the initial distortion distribution for the case of 6 equally distributed RBFs, which is the first configuration whose approximation error falls under the target error. Figure 4 represents the same information when the clustering process has ended. We can now see the advantage that we expect from the fact of making each cluster to have an equal contribution to the total distortion, which is the objective of the proposed clustering algorithm. The distortion is defined as:

$$D = \frac{\sum_{j=1}^{m} \sum_{\vec{x}_i \in C_j} \left\| \vec{x}_i - \vec{c}_j \right\|^2 E_{ij}}{\sum_{j=1}^{m} \sum_{\vec{x}_i \in C_j} E_{ij}} \quad (5)$$

where m is the number of RBFs (clusters), $\vec{c}_j$ is the centre of cluster C_j and E_{ij} is the error committed by the net when the input vector $\vec{x}_i$ belongs to cluster C_j.

$$E = \left| y - f(\vec{x}, \Phi, w) \right| \quad (6)$$

In the local displacement of the cluster centres, we start by making a hard partition of the training set, just as in the k-means algorithm. The second step of the process of local displacement is the calculation of the error of the RBFNN using the the K-nearest neighbours algorithm to initiate the radii and the singular value decomposition to calculate the weights of the RBFs.This is carried out by an iterative process that updates each cluster centre as the weighted mean of the training data belonging to that cluster and we repeat this process until the total distortion of the net reaches a minimum.

$$\vec{c}_m = \sum_{\vec{x}_i \in C_m} E_{im} \cdot \vec{x}_i \; / \sum_{\vec{x}_i \in C_m} E_{im} \quad (7)$$

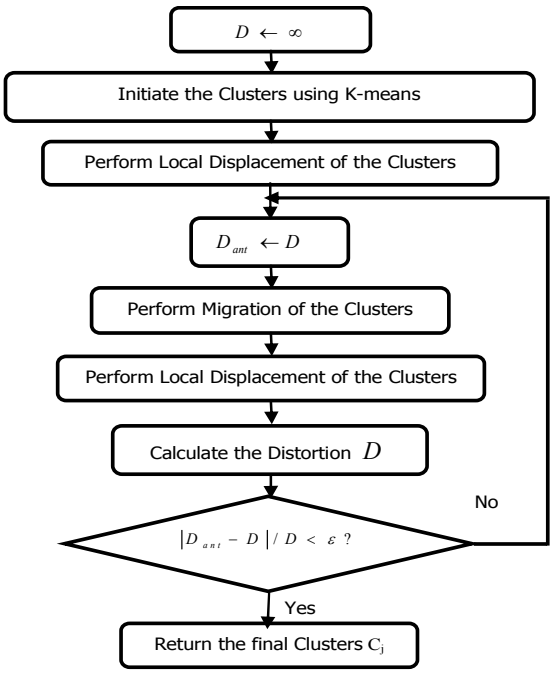

Figure 5. General description of the proposed clustering algorithm.

After this process we must update the cluster centres in order to minimize the total distortion. The algorithm stops when the value of the distortion is less than the value of a threshold ε. Figure 6 presents a flowchart with the general description of the local displacement process.

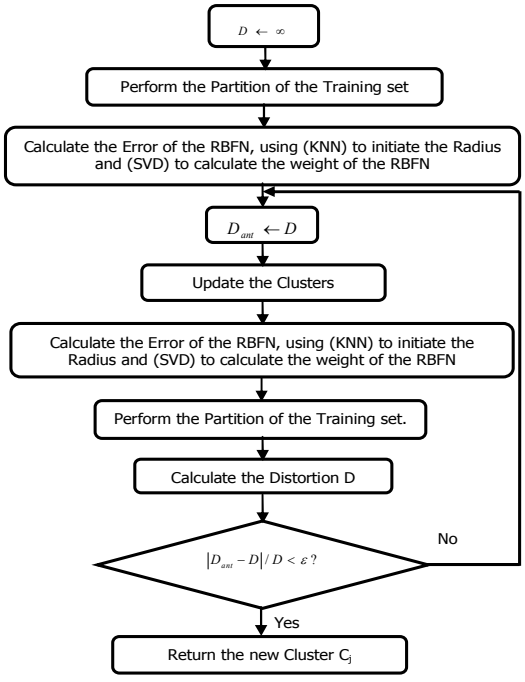

Figure 6. Local displacement of the clusters.

The migration process migrates clusters from the better zones toward those zones where the error is worse, thus attempting to make equal their contribution to the total distortion.

Our main hypothesis is that the best initial cluster configuration will be the one that equalizes the error committed by every cluster. The probability of

choosing a given cluster inversely proportional to what we call *the utility* of that cluster, which is defined as:

$$U_j = D_j / \bar{D} \quad j = 1,\ldots,m \tag{8}$$

In this way, the proposed algorithm selects one cluster that has utility less than one and moves this cluster to the zone nearby a new selected cluster having utility more than one as shown in Figure 7. This migration step is necessary because the local displacement of clusters only moves clusters in a local manner.

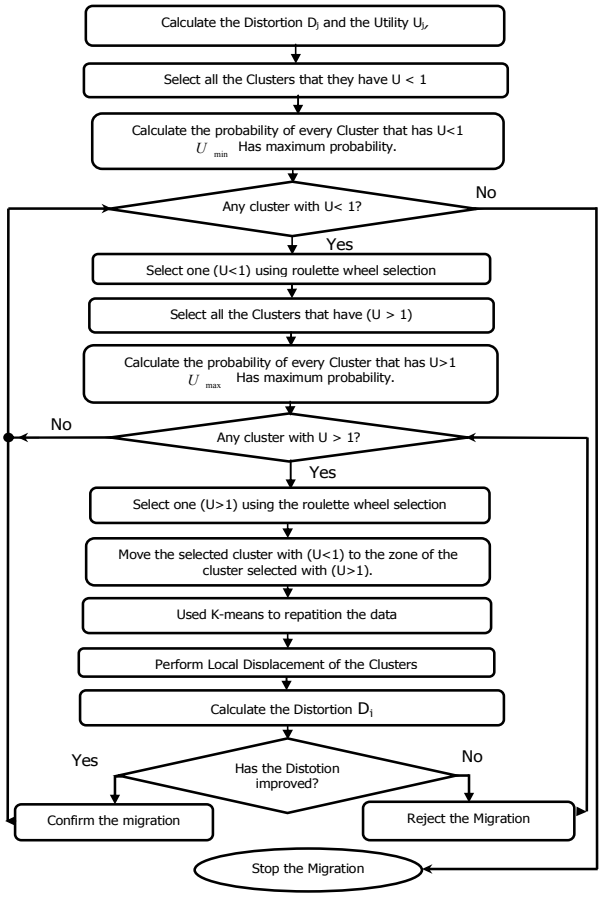

Figure 7. The migration process.

4. Example of the Proposed Procedure

Experiments have been performed to test the proposed algorithm. The system is simulated in MATLAB 7.0 under Windows XP with processor Pentium IV running at 2.4 Ghz. In this section we attempt a short-term and large-term prediction of the algorithm presented in the above section with regard to the Mackey-glass time series data [5]. The Mackey-glass time series is commonly used to test the performance of neural networks [5].

The series is a chaotic time series making it an ideal representation of the nonlinear oscillations of many physiological processes [17]. To make the comparisons with earlier work, we chose the parameters presented in [4]. Figure 8 shows the Mackey- glass time series. Tables 1 and 2 compares the prediction accuracy of different computational paradigms presented in the bibliography for this benchmark problem (including our proposed approach). Mackey-glass time series generated with the following expression:

$$\frac{ds(t)}{dt} = a \cdot \frac{s(t-t)}{1+s^{10}(t-t)} - bs(t) \tag{9}$$

where *x(t)* is the value of the time series at time *t*. The time series was constructed with parameter values $a = 0.2$ and $b = 0.1$. Here, initial conditions used in our test bench are set as $s(0) = 1.2$ and $s(t) = 0$ when $t < 0$, doing $t = 17$. 1000 samples of the Mackey-glass time series are depicted Figure 8. The first 500 points are used as a training set and the last 500 are used as the test set. The tables present results of the normalized root mean-square error $NRMSE_{test}$ obtained by testing set of 500 test points after the application of the Levenberg–Marquardt method. As can be seen from the Tables 1 and 2, the proposed algorithm reaches better prediction error.

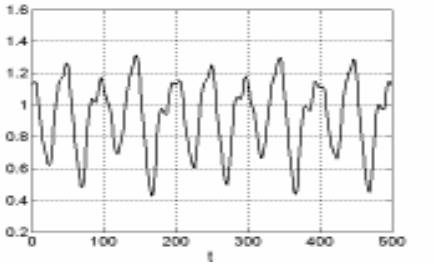

Figure 8. Mackey- glass time series.

4.1. Short-Term Prediction

Following the conventions established to predict this time series in short-term, the execution of the algorithm is considered to look for networks that predict the value *s(s+6)* from current value *s(t)* and of past values *s(s-6), s(s-12),* and *s(s-18),* using values of training of the form

$$[s(t-18), s(t-12), s(t-6), s(t); s(t+6)] \tag{10}$$

The NRMSE of the points predicted by the algorithms is shown in Table 1. It is clear that the proposed algorithm has predicted the time series in short-term with much greater accuracy than other algorithms.

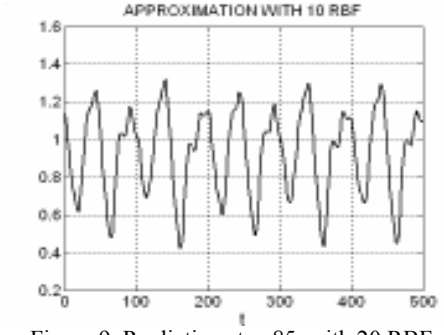

Figure 9. Prediction step 85, with 20 RBF.

Table 1. Comparison result of the prediction error of different methods for prediction step 6 (500 test data).

Method	m	$RMSE_{test}$
Lineal model prediction	-	0.55
Auto Regressive model	-	0.19
Cascade correlation NN	-	0.06
6^{th}-order polynomial	-	0.04
Back-Prop NN	-	0.02
Kim and Kim (GA A & Fuzzy System) [3]	5MFs	0.049
	7MFs	0.042
	9MFs	0.038
ANFIS & Fuzzy Svstem [3]	16 rules	0.007
New RBFNs Structure [16]	12 RBF	0.003
Pomares [13]	3 × 3 × 3 ×3	0.011
	3 × 4 × 4 ×4	0.007
	4 × 4 × 5 ×5	0.006
González [7]	4	0.015 ± 0.0019
	7	0.007 ± 0.0009
	10	0.005 ± 0.0010
	13	0.004 ± 0.0011
	16	0.004 ± 0.0002
Rivas [15]	4	0.014 ± 0.0021
	7	0.009 ± 0.0008
	10	0.007 ± 0.0009
	13	0.006 ± 0.0011
	16	0.005 ± 0.0003
Our Approach	4	0.012 ± 0.0080
	7	0.007 ± 0.0008
	10	0.005 ± 0.0006
	13	0.004 ± 0.0012
	16	0.003 ± 0.0006

4.2. Large-Term Prediction

In large-term, the execution of the algorithm is considered to look for networks that predict the value *s(s+85)* from current value *s(t)* and of past values *s(s-6), s(s-12),* and *s(s-18),* using values of training of the form

$$[s(t-18), s(t-12), s(t-6), s(t); s(t+85)] \quad (11)$$

5. Conclusion

In this paper, a new modified approach is presented to predict chaotic time series. We have proposed an algorithm of clustering especially suited for function approximation problems. This method calculates the error committed in every cluster using the real output of the RBFNN, and not just an approximate value of that output, trying to concentrate more clusters in those input regions where the approximation error is bigger, thus attempting to homogenize the contribution to the error of every cluster. Simulations, in this paper have demonstrated that the proposed method produces more accurate prediction. This algorithm is easy to implement and is superior in both performance and computation time to other algorithms.

Table 2. Comparison result of the prediction error of different methods for prediction step 85 (500 test data).

Method	m	$NRMSE_{test}$
RAN-P-GQRD [1]	14	0.206
	24	0.174
	31	0.160
	38	0.183
Fuzzy system [1]	10	0.108
	11	0.109
	12	0.103
	13	0.223
	14	0.159
	15	0.103
Whitehead [20]	25	0.29
	50	0.18
	75	0.11
	125	0.05
González [7]	5	0.389 ± 0.0194
	10	0.251 ± 0.0246
	14	0.198 ± 0.0164
	17	0.147 ± 0.0178
	20	0.126 ± 0.0174
Rivas [15]	5	0.397 ± 0.0238
	10	0.249 ± 0.0207
	14	0.168 ± 0.0210
	17	0.128 ± 0.0091
	20	0.113 ± 0.0125
Our Approach	5	0.388 ± 0.0227
	10	0.243 ± 0.0130
	14	0.150 ± 0.0303
	17	0.111 ± 0.0156
	20	0.097 ± 0.0074

References

[1] Bersini H., Duchateau A., and Bradshaw N., "Using Incremental Learning Algorithms in the Search for Minimal Effective Fuzzy Models," *in Proceedings of 6th International Confrances on Fuzzy Systems*, Barcelona, pp. 819-825, 1997.

[2] Bezdek C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum, New York, 1981.

[3] Duda O. and Hart E., *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[4] Frank J., Davey N., and Hunt P., *Time Series Prediction and Neural Networks*, Hatfield, UK, 1997.

[5] Gonzalez J., Rojas I., and Pomares H., "A New Clustering Technique for Function Approximation," *IEEE Transaction Neural Networks*, vol. 13, no. 1, pp. 132 -142, 2002.

[6] González J., "Identificación y Optimización de Redes de Funciones de Base Radiales Para Aproximación Funcional," *PHD Thesis*, University of Granada, 2001.

[7] Kanjilal P. and Banerjee N., "On the Application of Orthogonal Transformation for the Design and Analysis of Feedforward Networks," *IEEE Transactions on Neutral Networks*, vol. 6, no. 2, pp. 1061-1070, 1995.

[8] Kim D. and Kim C., "Forecasting Time Series with Genetic Fuzzy Predictorensemble," *IEEE*

Transactions on Fuzzy Systems, vo1. 5, no. 4, pp. 523-535, 1997.

[9] Mackey C. and Glass L., "Oscillation and Chaos in Physiological Control Systems," *Science Report*, 1977.

[10] Marquardt W., "An Algorithm for Least-Squares Estimation of Nonlinear Inequalities," *SIAM Journal on Applied Mathematics*, vol. 11, no. 4, pp. 431-441, 1963.

[11] Moody E. and Darken C., "Fast Learning in Networks of Locally Tuned Processing Units, *Computer Journal Neural Computation*, vol. 2, no. 1, pp. 281-294, 1989.

[12] Orr L., "Regularization in the Selection of Radial Basis Function Centers," *Computer Journal Neural Computation*, vol. 7, no. 3, pp. 606-623, 1995.

[13] Pedrycz W., "Conditional Fuzzy C-means," *Pattern Recognition Letters*, vol. 17, no. 2, pp. 625-632, 1996.

[14] Pomares H., "Nuevo Metodología Para El Diseño Automático De Sistemas Difusos," *PHD Thesis*, University of Granada, 2000.

[15] Rivas A., "Diseño Y Optimización de Redes de Funciones de Base Radial Mediante Técnicas Bioinspiradas," *PHD Thesis*, University of Granada, 2003.

[16] Rojas I., Pomares H., Gonzalez J., and Ros A., "A New Radial Basis Function Networks Structure: Application to Time Series Prediction," *in Proceedings of IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN-2000)*, Italy, pp. 449-445, 2000.

[17] Rosipal R., Koska M., and Farkas I., "Prediction of Chaotic Time-Series with a Resource Allocating RBF Network, *Computer Journal Neural Processing Letters*, vol. 7, no. 3, pp. 1-13, 1998.

[18] Runkler A. and Bezdek C., "Alternating Cluster Estimation: A New Tool for Clustering and Function Approximation," *IEEE Transactions on Fuzzy Systems*, vol. 7, no. 3, 377- 393. 1999.

[19] Russo M. and Patanè G., "Improving the LBG Algorithm," *Lecture Notes in Computer Science*, vol. 1606, no. 4, pp. 621-630, 1999.

[20] Whitehead A. and Choate D., "Cooperative-Competitive Genetic Evolution of Radial Basis Function Centers and Widths for Time Series Prediction," *IEEE Transactions on Neural Networks*, vol. 7, no. 4, pp. 869-80, 1996.

[21] Xiaoyu L., Bing K., and Simon Y., *Time Series Prediction Based on Fuzzy Principles*, Heidelberg, Florida, 2003.

Mohammed Awad received the BSc degree in industrial automation engineering in 2000, from the Palestine Polytechnic University and the PhD degree in 2005, from the University of Granada, Spain. He is currently assistant professor in the Faculty of Information Technology and Chair of the Department of computer Information Technology at the Arab American University, Palestine.

Héctor Pomares received the MSc degree in electronic engineering in 1995, the MSc degree in physics in 1997, and the PhD degree in 2000, all from the University of Granada, Granada, Spain. He is currently an associate professor in the Department of Computer Architecture and Computer Technology at the University of Granada.

Ignacio Rojas received the MSc degree in physics and electronics in 1992 and the PhD degree in 1996, both from the University of Granada, Spain. He was at the University of Dortmund, Germany, as invited researcher from 1993 to 1995.

Osama Salameh received the MSc degree in computer engineering in 1990 and the PhD degree in 1996, from Odessa State Polytechnic University, Ukraine. He is currently assistant professor in the Faculty of Information Technology and acting dean of the same faculty at the Arab American University, Palestine. His current areas of research interest include artificial neural networks and aspect oriented software development.

Mai Hamdon is currently a student in Arab American University in the Department of Computer Science. Her area of interests includes algorithms and artificial Intelligent.