Optimization RBFNNs Parameters Using Genetic Algorithms: Applied on Function Approximation

Mohammed Awad m.awad@aauj.edu
Faculty Engineering and Information Technology /CIT Dept.
Arab American University
Jenin, 240, Palestine

Abstract

This paper deals with the problem of function approximation from a given set of input/output (I/O) data. The problem consists of analyzing training examples, so that we can predict the output of a model given new inputs. We present a new approach for solving the problem of function approximation of I/O data using Radial Basis Function Neural Networks (RBFNNs) and Genetic Algorithms (GAs). This approach is based on a new efficient method of optimizing RBFNNs parameters using GA, this approach uses GA to optimize centres c and radii r of RBFNNs, such that each individual of the population represents centres and radii of RBFNNs. Singular value decomposition (SVD) is used to optimize weights w of RBFNNs. The GA initial population performed by using Enhanced Clustering Algorithm for Function Approximation (ECFA) to initialize the RBF centres c and k-nearest neighbor to initialize the radii r. The performance of the proposed approach has been evaluated on cases of one and two dimensions. The results show that the function approximation using GA to optimize RBFNNs parameters can achieve better normalized-root-mean square-error than those achieved by traditional algorithms.

Keywords: Radial Basis Function Neural Networks, Genetic Algorithms and Function Approximation.

1. INTRODUCTION

Function approximation is the name given to a computational task that is of interest to many science and engineering communities [1]. Function Approximation consists of synthesizing a complete model from samples of the function and its independent variables [2]. 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 [3, 12].

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 [23]. Genetic algorithms (GAs), on the other hand, are very robust and explore the search space more uniformly, since every individual is evaluated independently, which makes GAs very suitable to the optimization of Neural Networks [4]. 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 [23, 25]. 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 [5, 6].

One type of neural network, called Radial Basis Function Neural Networks (RBFNNs) [24], has the property of universal approximation and has received some attention by other researchers, but its parameters have, so far, been only partially optimized using GAs [1, 12]. RBFNNs are characterized by a transfer function in the hidden unit layer having radial symmetry with respect to a centre [7]. The basic architecture of RBFNNs is a 3-layer network as in Figure 1. The output of the RBFNNs 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 $\Phi = \{\phi_i : i = 1, ..., m\}$ is the basis functions set, and w_i is the associate weights for every 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.

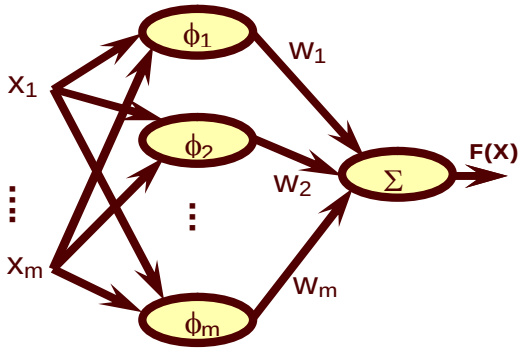


Fig.1. Radial Basis Function Network

Topology optimization is a common learning method for RBFNNs, but a big challenge is optimization that includes the full parameter sets of centres c, radii r and weights w along with the number of neurons per hidden layer. There are several possibilities of using GAs to configure RBFNNs. A straightforward approach is to fix a topology and use GA as an optimization tool to compute all free-parameters [8]. In [9] the author fixed the number of hidden neurons, and used GA to optimize only the location of the RBFNNs centres. The radii and output weights were computed by the K-nearest neighbor KNN and the singular value decomposition SVD, respectively. In [10] the author also fixed the number of centres, and evolved their locations and radii, instead of encoding a network in each individual, the entire set of chromosomes cooperates to constitute RBFNNs. Another idea is to hybridize the configuration process, using GA as a support tool. Chen et. al. [13] presented a two-level learning method for RBFNNs, where a regularized orthogonal least squares (ROLS) algorithm was employed to construct the RBFNNs at the inner level, while the two main parameters of this algorithm were optimized by a GA process at the outer level. In [14], GA was used to optimize the number and initial positions of the centres using the k-means clustering algorithm; the RBFNNs first training then proceeded as in [15].

In our approach we present a different way that depends on optimizing the topology of RBFNNs and its parameters centres c, and radii r using GA. Weights w are calculated by means of methods of resolution of linear equations. In this proposed approach we use the singular values decomposition (SVD) to solve this system of linear equations and assign the

weights w for RBFNNs to calculate the output. Each individual is an entire set of chromosomes cooperate to constitute a RBFNNs. In our proposed approach we use the incremental method to determine the number of RBF (neurons) depending on the data-test-error that the system produces which means, an increase in each iteration will be only one RBF 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 RBFNNs 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. 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 $y = F(\vec{x})$ 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 RBFNNs and GA is based on the using of GA to optimize the RBFNNs parameters (centres c, and radii r) so that the neuron is put in a suitable place in input data space [11]. The form of combining RBFNNs with GAs appears in Figure 2.

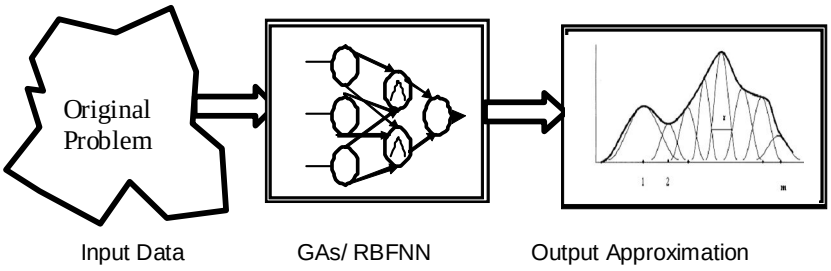


Fig.2. Combining GA and RBFNN

The process begins with an initial population generated using three techniques for the initialization of centres c, radii r, and weights w. The first technique is a clustering algorithm, designed for function approximation (ECFA) [16], which is used for initializing the RBF centres c. ECFA calculates the error committed in every cluster using the real output of the RBFNN, which is trying to concentrate more in those input regions where the approximation error is bigger, thus attempting to homogenize the contribution to the error of every cluster. Due to this fact, the cluster locations are located in different places depending on the paradigm used to model the internal relation in the I/O data [16]. The second technique is the k-nearest neighbors technique (Knn), which is used for the initialization of the radii r of each RBF. The Knn technique sets the radius of each RBF to a value equal to the mean of the Euclidean distance between the centres of their nearest RBF [1, 20]. The last technique is singular value decomposition (SVD), which is used to optimize directly the weights. The SVD technique is used to solve the problem of RBF misplacement by using singular matrix activation. If two functions are almost identical in the activation matrix, then two columns will be produced with equal weight, whereas if a RBF is not activated for any point, zero columns in the matrix will be produced [16, 20]. All these techniques are used once for the first configuration.

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 weights, centres and radii of RBFNNs. 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. PARAMETER OPTIMIZATION OF RBFNN 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 [17].

Our new proposed approach use GAs to construct optimal RBFNNs. The approach uses GAs evolving to optimize the two RBFNNs parameters (centres c, and radii r) and uses singular value decomposition (SVD) to optimize directly the weights w. The general process of our proposed approach can be depicted in Figure 3, and the pseudo-code of this algorithm reads:

Begin
Initialize population P {c [by ECFA], r [by Knn]}; and w [by SVD].
Evaluate each individual on population P by fitness function $F(x, \Phi, w)$;
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;

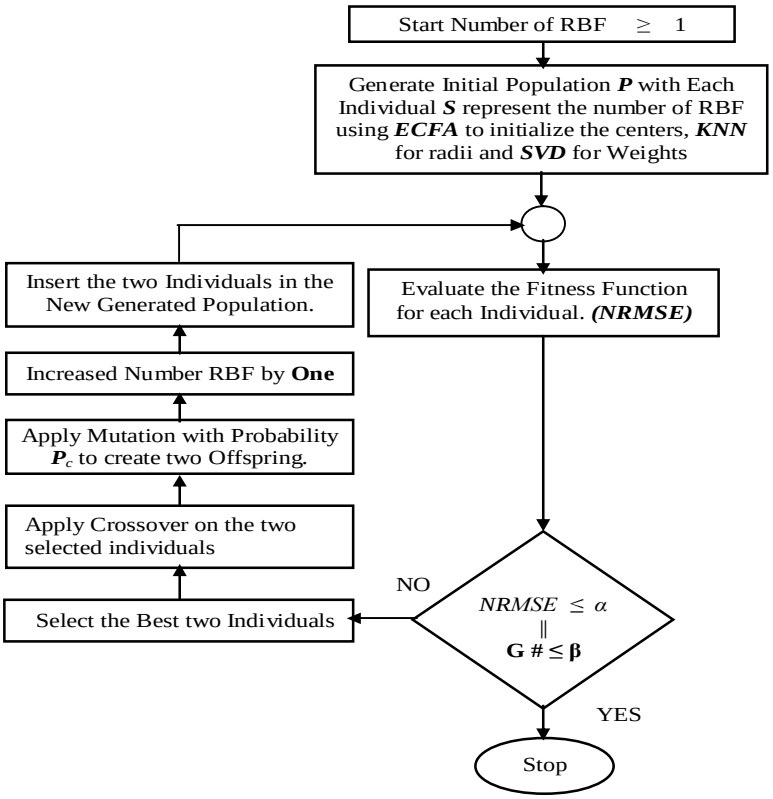


Fig. 3. General description of the proposed algorithm

3.1 Initialization

Each gene is constituted by a real vector representing centres, and a real value representing radii of RBFs m. Chromosomes have a variable length which defined as follow:

$$chrom = \left[\{c_{1m}, r_{1m}\}, \{c_{2m}, r_{2m}\} .... \{c_{im}, r_{im}\} \right] \quad (3)$$

In our approach the chromosome that consists of (centres c, radii r) is generated initially depending on classical algorithms so that initial centres will be generated once in the first configuration by an efficient method of clustering of the centres c of the RBF Network (ECFA) [16]. The K-nearest neighbors technique (Knn) used once in the first configuration for the initialization of the radii r of each RBF. The number of parameters in each chromosome calculated by [(# of RBF centres × # of dimensions) + # of RBF radii]. Singular value decomposition (SVD) is used directly to optimize the weights w.

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:

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

Where $\bar{y}$ is the mean of the target output, and *p* 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 RBFNN, 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 [18, 19]. The most common selection method depends on assignment of a probability *pj* 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 [18, 19]:

$$P\left[\text{individual selection-i}\right]=q^{+}(1-q)^{s-1} \quad (5)$$

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

$$q^{+}= \frac{q}{1-(1-q)^{P}} \quad (6)$$

Where P is the population size.

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:

$$\overline{X}^{!} = r\,\overline{X}+(1-r)\overline{Y} \quad (7)$$

$$\overline{Y}^{!} =(1-r)\,\overline{X}\ +r\overline{Y} \quad (8)$$

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 4.

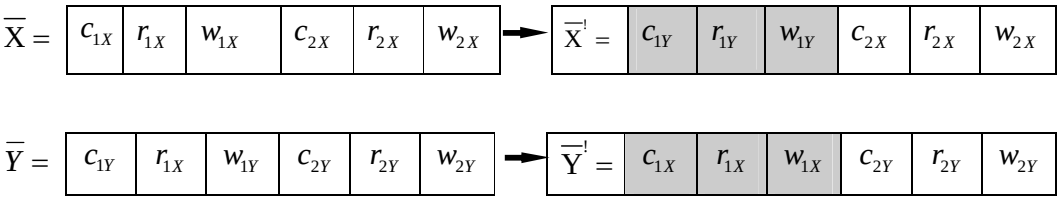


Fig.4. The process of the arithmetic crossover of three points in two neurons RBF

We can find many methods of mutation in [19], 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. 9):

$$x_i^{'} = \begin{cases} U(a_i, b_i) & if\ i = j \\ x_i & otherwise \end{cases} \quad (9)$$

Where a_i and b_i are down and top level, for every variable *i*. Figure 5 present the process of mutation that appears among the parameters of the RBFNNs.

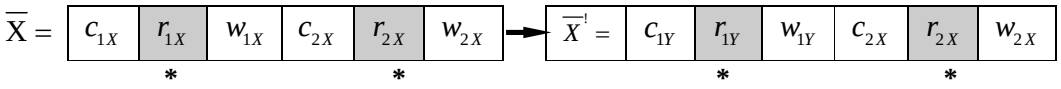


Fig.5. The uniform mutation of two points in two neurons RBF

4. SIMULATION EXAMPLES

The objective of this study is to develop and test an efficient approach that use to solve the problem of function approximation. Therefore, we assume different polynomial function to test the improvement of the approximation process depending on this approach. We have investigated three polynomial function problems, one function in one dimension and other two in two dimensions. The first function in figure 6 tests a case where there are many curves in the function structure. The numerical values in the function are created to proof that the proposed approach converges and dose not stuck in local minimums. Experiments have been performed to test the proposed approach. The system is simulated in MATLAB 7.0 under Windows XP with a Pentium IV processor running at 2.4 GHz. In this section we will compare the result of our approach with the results of other algorithms that approximate functions using GAs to optimize RBFNNs parameters. 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 [21, 22], and the approximation of function in two dimensions with the NRMSE and execution time. 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). The GA parameters that used are; the population-size = 100, crossover rate = 0.5 and mutation rate = 0.05.

4.1 One Dimension Examples $F_1(x)$

To test the effects caused by the proposed approach on initialization and avoiding local minimum of RBFs 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 test. This function is defined by the following expression:

$$F_1(x) = e^{-3x} \sin(10\pi x), \qquad x \in [0,1] \qquad (10)$$

We can note from figure 6 (a) that the error produces before the training process distributed in unhomogenized form along with the input data space. In figure 6 (b) the training process that depends on optimizing RBFNN parameters (centres and radii) by GA produce error distribution is homogenized form for each RBF along with the input data space

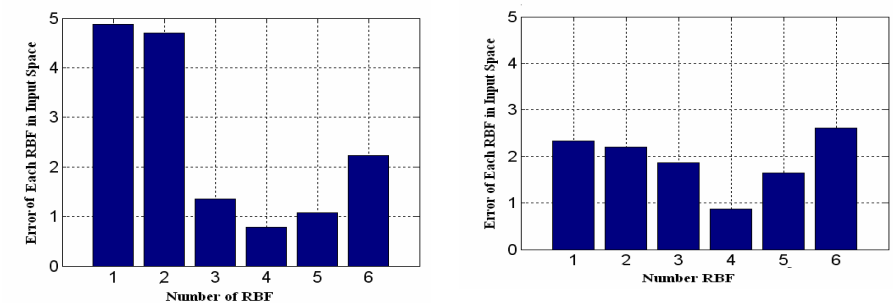


Fig. 6. (a) Error of each RBF in the input space Before the Training.

(b) Error of each RBF in the input space After the Training.

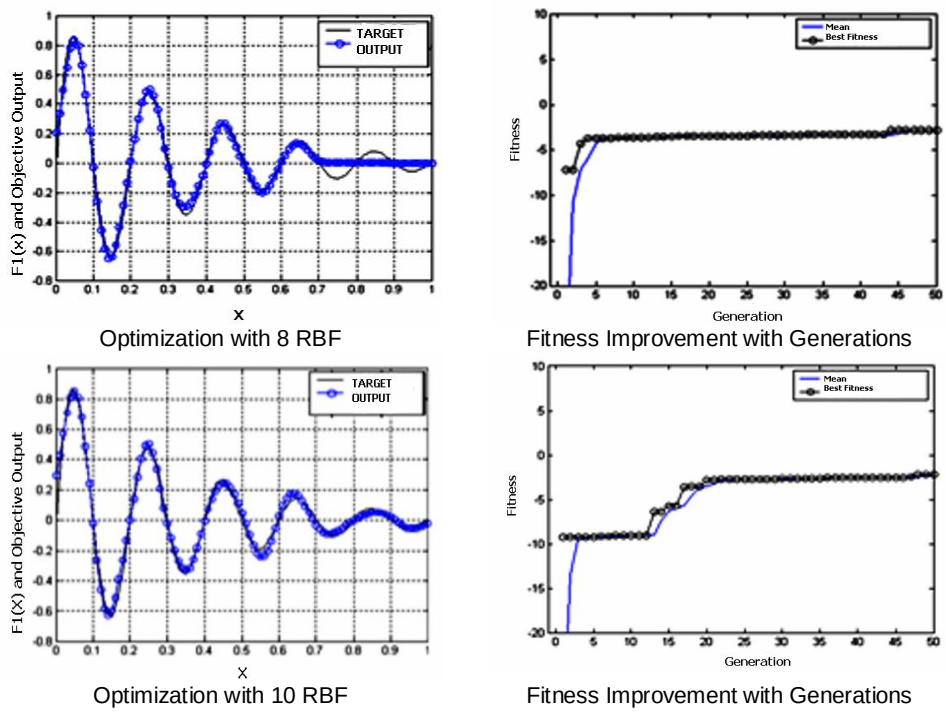
In Table 1, it can be seen that the proposed approach converge. This implies that RBFNN optimize not fall into local optimum solution. The $NRMSE_{Test}$ predicted by the proposed

approach shown that the proposed approach minimizes the approximation error with much accuracy than other algorithms.

Method	# RBF	$NRMSE_{Test}$	
González [22]	5	0.1771	
	6	0.1516	
	8	0.0674	
	10	0.0882	
Rivas [21]	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
Our Approach	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

TABLE1: Comparison Result of $NRMSE_{Test}$ Error of different approach

It's clear in figure 7 that the distribution of RBFs in the case of approximation with 8 RBF is not affected in the right part of the function, but when we increased the number of RBF as in approximation with 10 RBF, the approximation process is efficient, which is clear in the improvement of the fitness value with the increased number of generations. These results indicate that using GA to optimize RBFNN centres and radii give optimal performance.



Optimization with 8 RBF — Fitness Improvement with Generations

Optimization with 10 RBF — Fitness Improvement with Generations

Fig. 7. Approximation of the function and Improvement of fitness with Generations

A comparison between three approaches applied is shown in figure 8. We can see that the training precision of the algorithm presented in this paper is higher than other algorithms. The

$NRMSE_{Test}$ becomes smaller and the fitness becomes larger accompanying the increase of the generation; the fitness changes slowly when the generation number is between 20 and 50; we can judge that the convergence condition is satisfied when the generation number reaches 20, because the fitness does not increase any more.

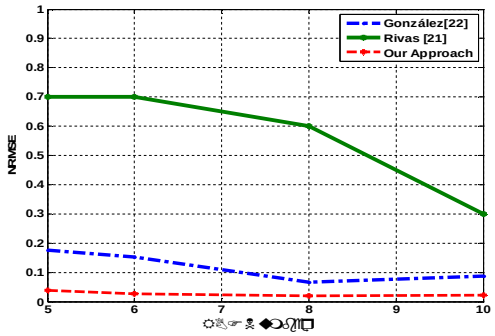


Fig. 8. Comparison the $NRMSE_{Test}$ with the increase of RBF numbers between different approaches.

4.2 Two Dimension Examples $F_1(x_1,x_2)$

In this part we used functions of two-dimensions (see Figure 9, Figure 11). 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.

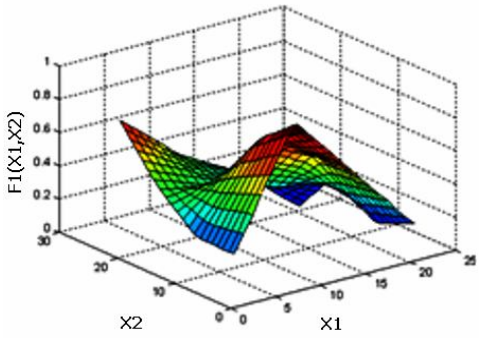


Fig. 9. Objective function $F1(x_1,x_2)$

Figure 10 presents different result of approximation of the function $F_1(x_1,x_2)$, and the improvement of fitness function ($NRMSE_{Test}$) with the increased generation numbers.

Nº RBF	NRMSE	Execution Time (sec)		
	Mean	Max	Min	Mean
2	0.224	130	122	127
4	0.176	164	144	156
6	0.124	169	147	157
8	0.115	192	181	186
10	0.27	203	184	192

TABLE3. Result of $NRMSE_{Test}$ and Execution Time of the proposed approach applied on 2D Function $F_1(x_1,x_2)$

Table 3 shows two results, the mean of $NRMSE_{Test}$ after 5 executions and the time of the approximation in seconds. The $NRMSE_{Test}$ of the RBFNN trained by GA is lower which means that the proposed approach converges and dose not stuck in local minimum. Although the RBFNN optimized by GA gives a lower $NRMSE_{Test}$ and higher approximation accuracy on the training data, it requires small computation time to converge.

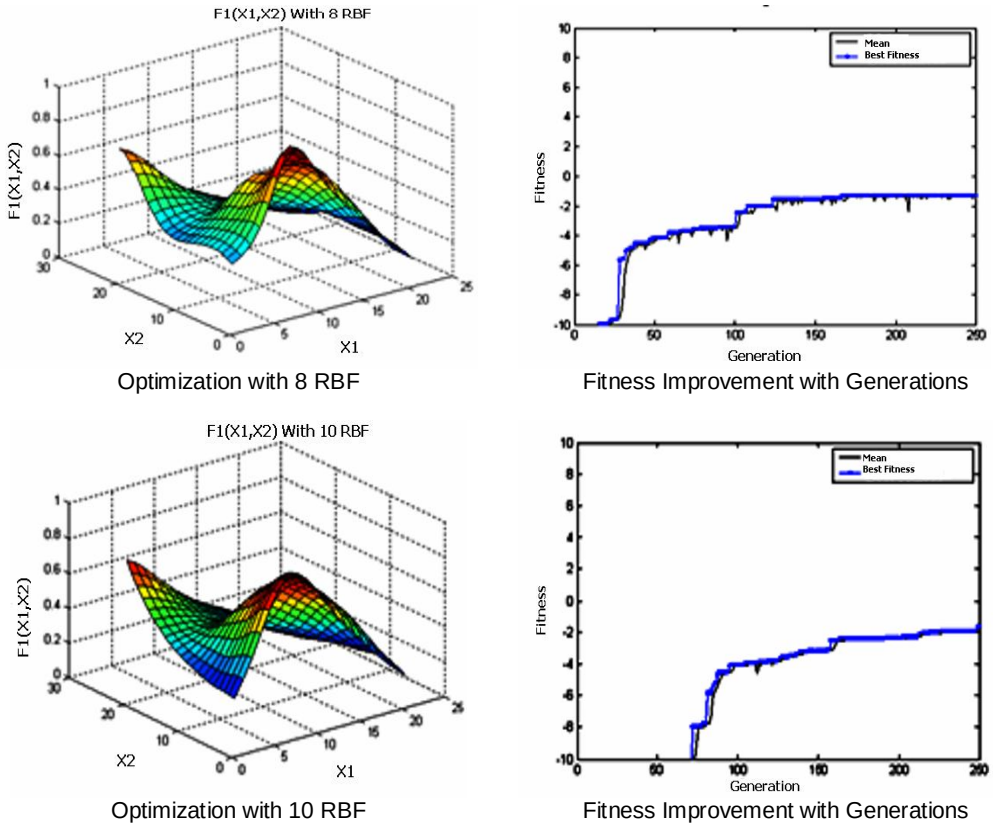


Fig. 10. Approximation of the function $F_1(x1,x2)$ and Improvement of fitness with Generations

The $NRMSE_{Test}$ becomes smaller and the fitness becomes larger accompanying the increase of the generation; the fitness changes slowly when the generation number is between 175 and 250; we can judge that the convergence condition is satisfied in this study case of 2 dimensions when the generation number reaches 175, because the fitness does not increase any more.

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

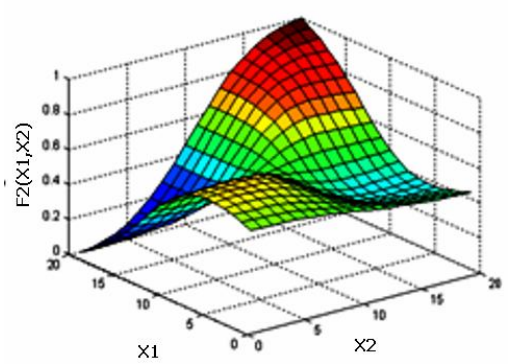
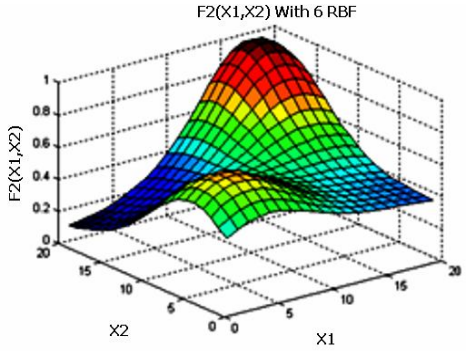


Fig. 11. Objective function $F_2(x1,x2)$

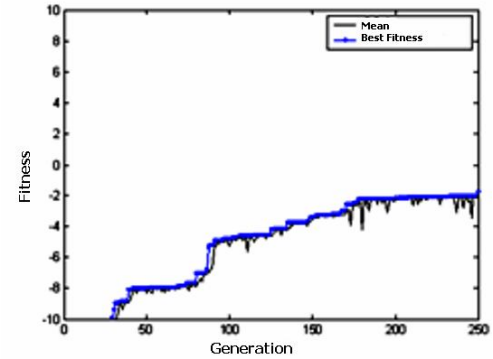
Figure 12 presents different result of approximation of the function $F_2(x_1, x_2)$ and the improvement of Fitness function ($NRMSE_{Test}$) with the increased generation numbers.

Nº RBF	NRMSE	Execution Time (sec)		
	Mean	Max	Min	Mean
2	0.53	122	112	117
4	0.37	132	121	127
6	0.28	169	147	158
8	0.22	188	175	178

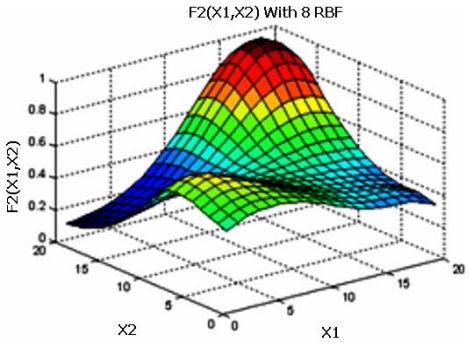
TABLE4. Result of $NRMSE_{Test}$ and Execution Time of the proposed approach applied on 2D Function $F_2(x_1,x_2)$



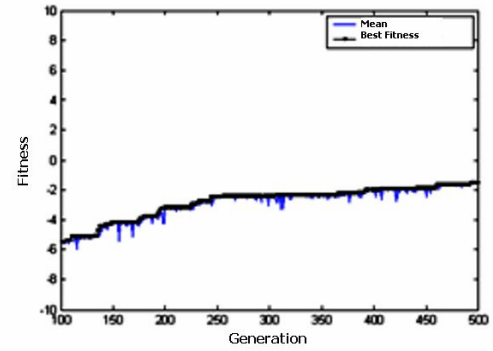
Optimization with 6 RBF



Fitness Improvement with Generations



Optimization with 8 RBF



Fitness Improvement with Generations

Fig. 12. Approximation of the function $F_2(x1,x2)$ and Improvement of fitness with Generations

5. CONCLUSION

In our paper an efficient way of applying GA to RBFNNs configuration has been presented. The approach optimizes centres c and Radii r parameters of RBFNN using GAs. The weights w are optimized by using singular value decomposition SVD. The initialization of the centres depends on an efficient algorithm of clustering (ECFA) [16] which means less complexity of calculation to optimize each parameter alone. This approach was compared to two approaches to optimize RBFNNs. The proposed approach is accurate as the best of the others approaches and with significantly less number of RBFs 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 using algorithms for the initialization of the RBFNN parameters. We have also shown that it is possible to use this approach to find the minimal number of RBF (Neurones) that satisfy a certain error target for a given function approximation problem.

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

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

[4] B. Carse, A.G. Pipe, T.C. Forgarty and T. Hill, "Evolving radial basis function neural networks using a genetic algorithm", IEEE International Conference on Evolutionary Computation, Vol. 1, page 300 (1995)

[5] 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.

[6] 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.

[7] A. Topchy, O. Lebedko, V. Miagkikh, “Fast Learning in Multilayered Neural Networks by Means of Hybrid Evolutionary and Gradient Algorithm”. in Proc. of the First Int. Conf. on Evolutionary Computations and Its Applications, ed. E. D. Goodman et al., (RAN, Moscow), pp.390–399, 1996.

[8] B. A. Whitehead and T.D. Choate. “Cooperative - Competitive Genetic Evolution of Radial Basis Function Centers and Widths for Time Series Predictio”. IEEE Transactions on Neural Networks, vol. 7, no. 8, pp.869-880, 1996.

[9] Fogel L.J., Owens A.J. and Walsh M.J. “Artificial Intelligence through Simulated Evolution”. John Wiley & Sons, 1966.

[10] M. W. Mak and K. W. Cho. “Genetic evolution of radial basis function centers for pattern classification”. In Proc. Of The 1998 IEEE International Joint Conference on Neural Networks, pages 669 – 673, 1998. Volume 1.

[11] A. F. Sheta and K. D. Jong. “Time-series forecasting using GA-tuned radial basis functions”. Information Sciences, Special issue, 2001.

[12] M. Awad, H. Pomares, F. Rojas, L.J. Herrera, J. González, A. Guillén. “Approximating I/O data using Radial Basis Functions:A new clustering-based approach”. IWANN 2005, LNCS 3512, pp. 289– 296, 2005.© Springer-Verlag Berlin Heidelberg 2005.

[13] 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.

[14] B. Burdsall and C. Giraud-Carrier. “GA-RBF: A selfoptimising RBF network”. In Proc. of the Third International Conference on Artificial Neural Networks and Genetic Algorithms, pages 348–351. Springer-Verlag, 1997.

[15] Y. Hwang and S. Bang. “An efficient method to construct a radial basis function neural network classifier”. Neural Networks, 10(8):1495–1503, 1997.

[16] M. Awad, H. Pomares, I. Rojas, Member, IEEE. “Enhanced Clustering Technique in RBF Neural Network for Function Approximation”. INFOS2007, Fifth International Conference 24-26 March 2007, Cairo University Post Office, Giza, Egypt.

[17] 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.

[18] P. T. Rodríguez-Piñero. “Introducción a los algoritmos genéticos y sus aplicaciones”. Universidad Rey Juan Carlos, España, Madrid. (2003)

[19] Z. Michalewickz. Univ. of North Carolina, Charlotte “Genetic Algorithms + Data Structures = Evolution Programs”. Springer-Verlag London, UK (1999).

[20] Gonzalez, J.; Rojas, H.; Ortega, J.; Prieto, A. “A new clustering technique for function approximation”. Neural Networks, IEEE .Transactions on, Volume: 13 Issue: 1, Jan. 2002. Page(s): 132 -142. “Conditional fuzzy C-means,” Pattern Recognition Lett., vol. 17, pp. 625–632, 1996

[21] Rivas. A. “Diseño y optimización de redes de funciones de base radial mediante técnicas bioinspiradas”. .PhD Thesis. University of Granada. 2003.

[22] González. J, “Identificación y optimización de redes de funciones de base radiales para aproximación funcional”. PhD Thesis. University of Granada. 2001.

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

[24] Sambasiva, R. Baragada, S. Ramakrishna, M.S. Rao, S. P. “Implementation of Radial Basis Function Neural Network for Image Steganalysis”, International Journal of Computer Science and Security, Vol. 2, Issue 1, pp. 12 – 22, March 2008

[25] Sufal D. Banani Saha, “*Data Quality Mining using Genetic Algorithm*”, International Journal of Computer Science and Security, ISSN: 1985-1553, 3(2): pp 105-112, 2009.