2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

PSO-Based Update Memory for Improved Harmony Search Algorithm to the Evolution of FBBFNT' parameters

S. Bouaziz, Adel M. Alimi, A. Abraham

Abstract—**In this paper, a PSO-based update memory for Improved Harmony Search (PSOUM-IHS) algorithm is proposed to learn the parameters of Flexible Beta Basis Function Neural Tree (FBBFNT) model. These parameters are the Beta parameters of each flexible node and the connected weights of the network. Furthermore, the FBBFNT's structure is generated and optimized by the Extended Genetic Programming (EGP) algorithm. The combination of the PSOUM-IHS and EGP in the same algorithm is so used to evolve the FBBFNT model. The performance of the proposed evolving neural network is evaluated for nonlinear systems of prediction and identification and then compared with those of related models.**

Keywords—**PSO-based update memory for Improved Harmony Search algorithm; Extended Genetic Programming; Flexible Beta Basis Function Neural Tree; nonlinear prediction systems, nonlinear identification systems.**

I. Introduction

The Harmony Search (HS) algorithm proposed by Geem [1], was inspired from the jazz musical improvisation. The latter is done by a skilled musician (corresponding to the decision variable) who plays (generate) a note (value) to obtain a best harmony state (global optimum). The HS method is so considered as an evolutionary stochastic global optimization technique like genetic algorithm [38], bacterial foraging optimization algorithm [39, 40]. According to [1, 2], the HS algorithm outperforms the conventional mathematical optimization algorithms and the Genetic Algorithm [3]. Such algorithm has successfully been applied to solve some engineering applications such as robotics [4], routing problems [5], Transport energy modeling [6], etc.

On the other hand, Artificial Neural Network (ANN) is a growing interdisciplinary field which considers the systems as adaptive, distributed and mostly nonlinear, three of the elements found in the real applications. It is placed at the crossroads of various biological-inspired approaches where it is considered as an abstract simulation of a real nervous system. ANN is composed of an interconnected group of artificial neurons distributing in layers to model complex relationships between inputs and outputs of the studied problem. It mimics also the learning behavior of biological systems by updating its parameters (including interconnection weights and in certain cases transfer function parameters).

S. Bouaziz and Adel M. Alimi is with REsearch Groups in Intelligent Machines (REGIM), University of Sfax, National School of Engineers (ENIS), BP 1173, Sfax 3038, Tunisia, (souhir.bouaziz@ieee.org, adel.alimi@ieee.org)
Ajith Abraham is with[2]IT4Innovations, VSB-Technical University of Ostrava, Czech Republic and[1]Machine Intelligence Research Labs, WA, USA; (ajith.abraham@ieee.org).
Corresponding author: S. Bouaziz.

The know-how accumulated, so, through advanced work in the process of ANN creating and learning showed that its reliability can be conditioned by the appropriate structure; the connection ways between the nodes; the chosen transfer function; and the training algorithm. Many efforts have been provided in the literature to address these issues. Yao [7] is one of the first researchers who has exploited possible benefits arising from the interactions between ANNs and evolutionary computation, to design and evolve ANN, and in such case the model is noted Evolving Artificial Neural Network (EANN).

Recently several studies have proposed using HS algorithm for adjusting connected weights of artificial neural networks [8-11].

In this context is located the works presented in this paper. Indeed, a PSO-based update memory for Improved Harmony Search (PSOUM-IHS) algorithm is proposed to optimize the parameters of the Flexible Beta Basis Function Neural Tree (FBBFNT) model [12-15]. These parameters are the flexible Beta node parameters' and the weights encoded in the best structure found by the Extended Genetic Programming (EGP) [12, 13]. The new model is applied for nonlinear identification and prediction systems.

The paper is organized as follows: Section 2 describes the basic concepts of FBBFNT model. The PSOUM-IHS algorithm will be detailed in Section 3. A hybrid FBBFNT evolving algorithm, which combines EGP and PSOUM-IHS is the subject of Section 4. The set of some simulation results for nonlinear prediction and identification systems are provided in Section 5. Finally, some concluding remarks are presented in Section 6.

II. Flexible Beta Basis Function Neural Tree Model

The initiative of using Beta function for designing Artificial Neural Network was introduced by Alimi [16]. This function has several advantages over the Gaussian function, such as its ability to generate more rich shapes (linearity, asymmetry, etc.) [17] and its great flexibility.

In this work, the Beta basis function neural network is encoded by the tree-based encoding method instead of the matrix-based encoding method [18], since this method is more flexible and gives amore modifiable and adjustable structure. The new model is called Flexible Beta Basis Function Neural Tree (FBBFNT). The FBBFNT is formed of a node set *NS* representing the union of function node set *F* and terminal node set *T*:

$$NS = F \cup T = \{\beta_2, \beta_3, \ldots, \beta_N, /_N\} \cup \{x_1, \ldots, x_M\} \quad (1)$$

Where:

- β_n (n = 2, 3, …, N) denote non-terminal nodes and represent flexible Beta basis neurons with n inputs and N is the maximum degree of the tree.
- $/_N$ is the root node and represents a linear transfer function.
- $x_1, x_2, \ldots, x_M$ are terminal nodes and define the input vector values.

The output of a hidden function node is calculated as a flexible neuron model (Figure 1).

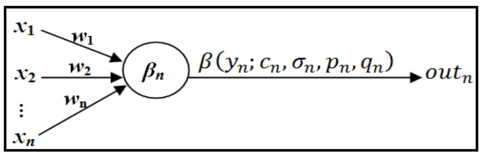

Fig. 1. A flexible Beta Basis function.

If a function node, i.e., β_n is selected, n real values are randomly generated to represent the weight connected between the selected node and its offspring. In addition, the Beta function has four adjustable parameters (the center c_n, the width σ_n and the form parameters p_n, q_n) are randomly generated as flexible Beta operator parameters. For each function node, its total excitation is calculated by:

$$y_n = \sum_{j=1}^{n} w_j * x_j \quad (2)$$

Where x_j (j = 1, …, n) are the inputs of the selected node and w_j (j = 1, …, n) are the weights.

The output of node β_n is then calculated by:

$$out_n = \beta_n(y_n, c_n, \sigma_n, p_n, q_n) =$$

$$\begin{cases} \left[1 + \frac{(p_n + q_n)(y_n - c_n)}{\sigma_n p_n}\right]^{p_n} \left[1 - \frac{(p_n + q_n)(c_n - y_n)}{\sigma_n q_n}\right]^{q_n} & \\ \quad if\, y_n \in \left]c_n - \frac{\sigma_n p_n}{p_n + q_n}, c_n + \frac{\sigma_n q_n}{p_n + q_n}\right[& \\ \quad 0 \qquad\qquad else & \end{cases} \quad (3)$$

The output layer yields a vector by linear combination of the node outputs of the last hidden layer to produce the final output.

A typical flexible Beta basis function neural tree model is shown in Figure 2. The overall output of flexible Beta basis function neural tree can be computed recursively by depth-first method from left to right.

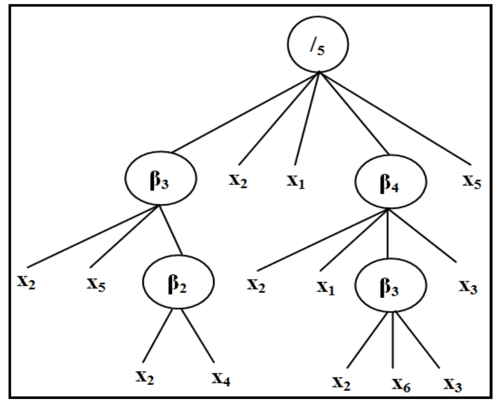

Fig. 2. A typical representation of FBBFNT: function node set F = {β_2, β_3,β_4,$/_5$}, and terminal node set T = {x_1, x_2, x_3, x_4, x_5, x_6}.

III. PSO-BASED UPDATE MEMORY FOR IMPROVED HARMONY SEARCH ALGORITHM: PSOUM-IHS

The Harmony Search (HS) algorithm searches the solution area as a whole to find the optimum element, which optimizes the fitness function. The steps in the procedure of harmony search are as follows [1]:

- **Step 1:** Problem formulation and parameter settings.
- **Step 2:** Initialize randomly the harmony memory.
- **Step 3:** Improvise a new harmony.
- **Step 4:** Update the harmony memory.
- **Step 5:** Checking stopping criterion

When the HS algorithm generates a new element, it considers all of the existing elements in the harmony memory (population) with fewer mathematical requirements. This characteristic makes the HS more flexible, the implementation easier and it is very versatile to combine HS with other meta-heuristic algorithms such as Particle Swarm Optimization (PSO) algorithm [19].

Moreover, in order to improve the adjusting characteristic of HS algorithm, Mahdavi *et al.*[20] suggested evolving the parameters instead of being fixed during the iterations. In fact, the authors suggested that *PAR* (Pitch Adjustment Rate) increase linearly and FW (width of the fret or bandwidth) decrease exponentially with iterations. Therefore, mathematic expressions were adapted into these parameters to follow the iteration change:

$$PAR = \frac{PAR_{max} - PAR_{min}}{MaxIt} * currentIteration + PAR_{min} \quad (4)$$

$$FW = FW_{max} * \exp(coef * currentIteration) \quad (5)$$

$$coef = \frac{\log\left(\frac{FW_{min}}{FW_{max}}\right)}{\mathrm{NI}} \quad (6)$$

Most of the decision variables in the new harmony are selected from the other elements stored in harmony memory. In addition, the new harmony vector may have the opportunity to take a place in the memory after his fitness test. Then, this vector might influence the convergence speed of the HS to the global optimum. We can note also that the harmony memory is stable in most of the time and does not provide a large variety of values to the improvisation. Therefore, the HS has a low probability of generating a good-quality of the new harmony vector. For these reasons, we have to incorporate a mechanism to create a wide variety of values in memory while respecting their allowable ranges by a hybrid mechanism with PSO. This mechanism implicitly guides the global algorithm to converge to the optimal solution. The proposed algorithm (Figure 3) is called PSO-based update memory for Improved Harmony Search (PSOUM-IHS)[21]. In fact, the stochastic factors and the dynamic aspects of particle velocities can guide the system to the right areas of research in the workspace.

To ameliorate the performance of HS, we choose to apply the hybridization on the Improved Harmony Search algorithm instead of the basic version. Indeed, the memory vectors of IHS are considered as particles of the swarm and the new memory values for new improvisation as the new positions reached by these particles. The velocity parameter is calculated for each particle i with the position $x_i$according to the following equation:

$$v_i(t+1) = \Psi(t)v_i(t) + c_1\varphi_1(p_i(t) - x_i(t)) + c_2\varphi_2(p_g(t) - x_i(t)) \quad (7)$$

Where c_1, c_2 (acceleration) and Ψ(inertia) are positive constant and $\varphi_1$and φ_2 are randomlydistributed number in [0,1]. In addition, p_i corresponds to the best position of the current particles according to the best fitness and p_g is the best position among all the particles obtained so far in the population. Each particle changes its position as the following equation:

$$x_i(t+1) = x_i(t) + \left(1 - \Psi(t)\right)v_i(t+1) \quad (8)$$

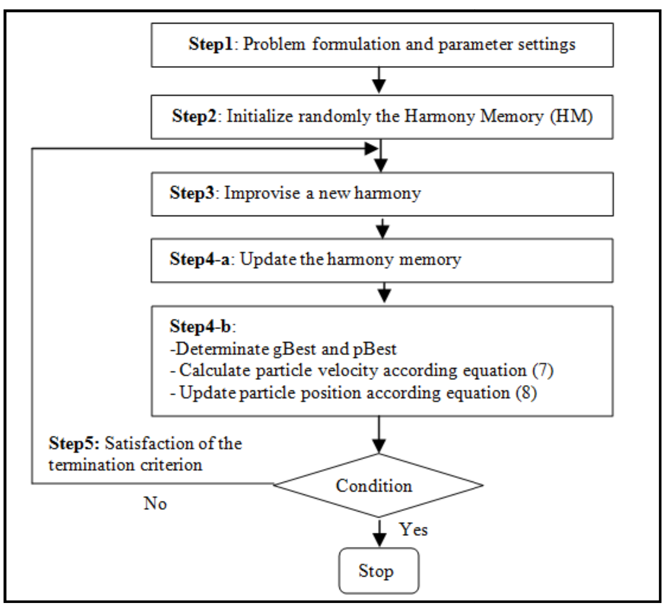

Fig. 3. Flowchart of PSOUM-IHS

In our case, an element of the memory is NParam × Sizematrix which corresponds to the optimal tree ofthe structure optimization. The PSOUM-IHS process is described as follows.

PSOUM-IHS Algorithm

```
input : HMCR, PARmin, PARmax, NI, NP, Fwmin and Fwmax.
- HMCR: rate of randomly selected values from the memory (0 ⩽ HMCR ⩽ 1)
- PAR: rate of altered values that was taken from the memory (0 ⩽ PAR ⩽ 1)
- NI: Improvisation number
- NP: Harmony Memory size
- FW: Width of the fret or bandwidth
output: p_g: the global best position.
begin
    Initialize randomly the Harmony Memory HM;
    Recognize x^worst ∈ {x^1, ..., x^NP};
    while (no terminal criterion specified by improvisation number :NI) do
        x^New := ∅ ;                                  /* new harmony matrix */
        for i := 1 to NParam do
            for j := 1 to Size do
                if (rand ≤ HMCR) then
                    x^New_{i,j} ∈ {x^1_i, ..., x^NP_i};   /* memory consideration */
                    if (rand ≤ PAR) then                   /* pitch adjustment */
                        x^New_{i,j} := x^New_{i,j} ± rand * FW;
                    end
                else
                    Random generation of x^New_{i,j} in parameter bounds;
                end
            end
        end
        Update the fitness function values and Update the p_g;
        if (Fit_par(x^New) ≤ Fit_par(x^worst)) then
            x^worst := x^New;
        end
        Choose the particle with the global best fitness value as the p_g;
        p_i correspond to the position of the particle which has the best fitness value;
        for i := 1 to NP do
            - Calculate particle velocity according equation (7);
            - Update particle position according equation (8);
        end
    end
end
```

IV. THE HYBRID ALGORITHM FOR EVOLVING FBBFNT MODEL

Evolving FBBFNT includes two issues which are architecture optimization and parameter adjustment. In this study, finding an optimal or a near optimal Beta basis function neural tree architecture is achieved by using Extended Genetic Programming (EGP) algorithm [12, 13].

The EGP which is an extended version of the standard genetic programming is formed by three mainly operators:

- **Selection operator:** is used to select two parent individuals from the population in order to procreate a new child by crossover/mutation operator.
- **Crossover operator:** is implemented by randomly taking selected two sub-trees in the individuals, and then swapping them.
- **Mutation:** four different mutation operators were used in the EGP to generate offspring from the parents. These mutation operators are as follows: changing one terminal node; changing all the terminal nodes; growing; pruning.

The parameters implanted in a FBBFNT are optimized by the PSO-based update memory for Improved Harmony Search (PSOUM-IHS) algorithm as described in section III.

So, to find an optimal or near-optimal FBBFNT model, structure and parameter optimization are used alternately. Combining of the EGP and PSOUM-IHS algorithms, a hybrid algorithm for evolving FBBFNT model is described as follows:

a) Randomly create an initial population (FBBFNT trees and its corresponding parameters);

$G = 0$, where G is the generation number of the evolving algorithm;

GlobalIter = 0, where *GlobalIter* is the global iteration number of the learning algorithm;

b) Structure optimization is achieved by the Extended Genetic Programming (EGP) with structure fitness function Fit_{stru}.

$$Fit_{stru}(i) = \alpha f_1(i) + \delta f_2(i) \quad (9)$$

Where f_1 measures the RMSE between the target and output of the proposed model. The function $f_2$measures the complexity of the FBBFNT model. α, δ are user specified fitness coefficients that allow a trade-off between the objectives,$\alpha, \delta \in [0,1]$.

$$f_1(i) = \sqrt{\frac{1}{P}\sum_{j=1}^{P}(y_t^j - y_{out}^j)^2} \quad (10)$$

Where P is the number of samples, y_t^j and y_{out}^j are the desired output and the FBBFNT output of j^{th} sample.

$$f_2(i) = \frac{Size_i - SizeMin + 1}{SizeMax - Size_i + 1} \quad (11)$$

Where $Size_i$ is the node number of the i^{th} individual. The *SizeMax* and *SizeMin* are respectively the maximum and the minimum size of the tree.

c) If a better structure is found or a maximum number of EIP iterations is attained, then go to step (d),

GlobalIter = GlobalIter + EGP_Iter;

otherwise go to step (b);

d) Parameter optimization is achieved by the PSOUM-IHS algorithm. The architecture of FBBFNT model is fixed, and it is the best tree found by the structure search. The parameters (weights and flexible Beta function parameters) encoded in the best tree formulate a harmony matrix. The parameter fitness function is equal to the function f_1.

e) If the maximum number of PSOUM-IHS iterations is attained, or no better parameter vector is found for a fixed time then go to step (f);

GlobalIter = GlobalIter + PSOUM-IHS_Iter;

otherwise go to step (d);

f) If satisfactory solution is found or a maximum global iteration number is reached, then the algorithm is stopped; otherwise let ($G = G + 1$) and go to step (b).

V. Experimental Results

The FBBFNT model is applied to approximate the input/output map of nonlinear systems. Indeed, in order to prove the effectiveness of FBBFNT model, we compare its results with those provided by other learning methods in the literature. Before that the setting set of the FBBFNT is presented in the following Table 1.For all examples the illustrated results are obtained by averaging the results in 10 runs.

TABLE I. The FBBFNT's settings

Algorithm	*Parameter*	*Initial value*
EGP	Population size	50
	Crossover probability	0.3
	Mutation probability	0.6
	Maximum generation number	1000
	Structure fitness settings (α, δ)	(0.95, 0.05)
PSOUM-IHS	Population size	50
	Maximum iteration number	4000
	PARmin	0.00001
	PARmax	1.0
	HMCR	0.9
	c_1	0.2
	c_2	0.7
Hybrid evolving algorithm	Maximum global iteration number	40 000
	Connected weights	rand[0, 1]
	Beta center c	rand[*min*(*x*), *max*(*x*)]
	Beta spread σ	rand[0, |*max*(*x*)-*min*(*x*)|]
	Beta form parameters (p, q)	rand[0, 5]

A. *Example 1: Mackey–Glass time series prediction*

A time-series prediction problem can be constructed based on the Mackey–Glass [22] differential equation:

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1+x^c(t-\tau)} - bx(t) \quad (12)$$

The settings of the experiment vary from one work to another. In our case, we take a = 0.2, b = 0.1, c = 10, and τ = 17. These values are the same onesused by the comparison systems [12-15, 23-28].As in the studies mentioned above, the task is to predict the value of the time series at point $x(t+6)$,with using the inputs variables $x(t)$, $x(t-6$, $xt-12$and$xt-18$.1000 sample points are used in our study. The first 500 data pairs of the series are used as training data, while the remaining 500 are used to validate the model identified.

The used Beta basis function sets to create an optimal FBBFNT model with EGP&PSOUM-IHS systemis$NS = \{\beta_2, \beta_3 /_3\} \cup \{x_1, x_2, x_3, x_4\}$, where x_i (i = 1, 2, 3, 4) denotes $x(t)$, $x(t-6)$, $x(t-12)$, and $x(t-18)$, respectively.

After 86 global generations (G = 86), 1,675,425 global number of function evaluations, and 23,206 global iterations of the hybrid learning algorithm, an optimal FBBFNT model was obtained with RMSE 4.8855e-13. The RMSE value for validation data set is 4.8876e-13. The evolved FBBFNT_EGP&PSOUM-IHS architecture is as shown in figure 4. The evolved FBBFNT_EGP&PSOUM-IHS output and the desired output are shown in figure 5.

In addition, the memory footprint of the source code is of the order of 10,720 *Kilo-bytes*, knowing that the memory footprint of the input data is 40 *Kilo-bytes*. For the execution time is of the order of 2924 *seconds*.

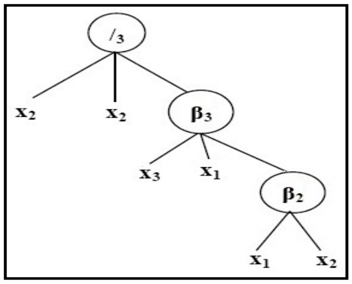

Fig. 4. Evolved FBBFNT_EGP&PSOUM-IHS architecture for Mackey-Glass time series prediction.

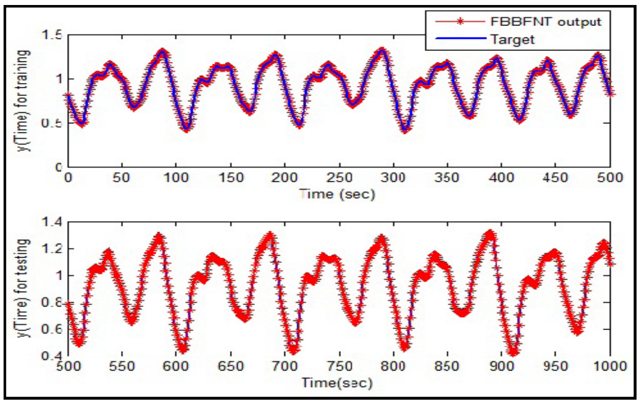

Fig. 5. Evolving FBBFNT_EGP&PSOUM-IHS output and the desired output for Mackey-Glass time-series prediction.

The FBBFNT_EIP&PSOUM-IHS model is essentially compared with the FBONT [12], FBBFNT_EGP&OPSO [13], FBBFNT_EIP&OPSO [14], and FBBFNT_EIP&HBFOA [15] with the same initial setting's values. The comparison is mainly based on the prediction error (RMSE) / Number of Function Evaluations (NFEs) compromise. In fact, for FBONT model RMSE is equal to 0.0076, NFEs is equal to 2,934,112 and for FBBFNT_EGP&OPSO, RMSE is equal to 0.0068, NFEs is equal to 1,966,825. Furthermore, for FBBFNT_EIP&OPSO model RMSE is equal to 0.0042, NFEs is equal to 1,616,648 and for FBBFNT_EIP&HBFOA, RMSE is equal to 1.8630e-09, NFEs = 6,004,148. It is clear that FBBFNT_EGP&PSOUM-IHS significantly reduces both the prediction error over the other four models and the number of function evaluations.

The FBBFNT_EIP&PSOUM-IHS network is also compared with Hierarchical multi-dimensional differential evolution for the design of Beta basis function neural network (HMDDE-BBFNN) [23] and the FNT model with Gaussian function as flexible neuron operator [24] and also with other systems. The HMDDE-BBFNN approach adopts for parameters: 50 to the population size, 10,000 to a total number of iterations, and 4 to the number of the hidden nodes. Moreover, the parameter settings of the FNT system are 30 to the population size, 135 as generation number, and 4 as hidden function unit number (with two hidden layers).

A comparison result of different methods for forecasting Mackey-Glass data is shown in Table 2. As observed, the FBBFNT_EIP&PSOUM-IHS achieves the lowest testing and training error.

TABLE II. COMPARISON OF DIFFERENT METHODS FOR THE PREDICTION OF MACKEY-GLASS TIME-SERIES.

Method	**Training error (RMSE)**	**Testing error (RMSE)**
HCMSPSO [26]	0.0095	0.0208
Fuzzy&MRB [27]	0.000990	0.000884
LNF[28]	0.0199	0.0322
FNT [24]	0.0069	0.0071
HMDDE–BBFNN [23]	0.0094	0.0170
GA-BBFNN [25]	-	0.013
FBONT [12]	0.0074	0.0076
FBBFNT_EGP&OPSO [13]	0.0061	0.0068
FBBFNT_EIP&OPSO [14]	0.004194	0.004299
FBBFNT_EIP&HBFOA[15]	5.3430e-10	1.8630e-09
FBBFNT_EGP&PSOUM-IHS	**4.8855e-13**	**4.8876e-13**

B. Example 2 : Box and Jenkins' Gas Furnace Problem

The gas furnace data of Box and Jenkins [29] was saved from a combustion process of a methane-air mixture. It is used as a benchmark example for testing prediction methods. The data set forms of 296 pairs of input-output measurements. The input $u(t)$is the gas flow into the furnace and the output $y(t)$is the CO_2 concentration in outlet gas. The inputs for constructing FBBFNT model are $y(t-1), u(t-4)$, and the output is $y(t)$. In this study, 200 data samples are used for training and the remaining data samples are used for testing the performance of the proposed model. The used instruction set is$NS = \{\beta_2/_3\} \cup \{x_1, x_2\}$, where x_i (i = 1, 2) denotes $y(t-1), u(t-4)$, respectively.

After 17 global generations (G = 27), 199,229 global number of function evaluations, and 4,564global iterations of the hybrid learning algorithm, an optimal FBBFNT model was obtained with RMSE 0.008135. The RMSE value for validation data set is 0.008773. In addition, the memory footprint of the source code is of the order of 2508 KB, knowing that the memory footprint of the input data is 7.008 KB. For the execution time is of the order of 1021 seconds.

The evolved FBBFNT_EGP&PSOUM-IHS architecture is as shown in Figure 6. Our method uses only three hidden function neurons for two hidden layers. The evolved FBBFNT_EGP&PSOUM-IHS output and the desired output are shown in Figure 7.

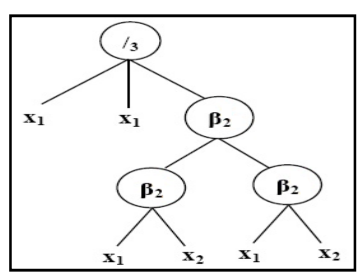

Fig. 6. The evolved FBBFNT_EGP&PSOUM-IHS for prediction of the Jenkins–Box time-series $(y(t-1), u(t-4))$.

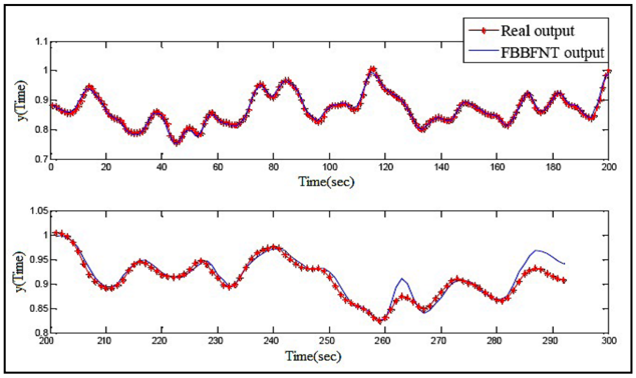

Fig. 7. The actual time series data and the output of the FBBFNT model for training and test samples to forecast the Jenkins–Box time-series $(y(t-1), u(t-4))$.

A comparison result of different methods for forecasting Jenkins-Box data is shown in Table 3.

TABLE III. COMPARISON OF TESTING ERRORS OF BOX AND JENKINS.

Method	**Prediction error (RMSE)**
ANFIS model [30]	0.0845
FNN_AP&PSO [31]	0.0260
FuNN model [32]	0.0714
FNT [24]	0.0256
HMDDE -BBFNN[23]	0.2411
FBBFNT_EGP& OPSO [13]	0.011618
FBBFNT_EIP& OPSO [14]	0.00981
FBBFNT_EIP& HBFOA [15]	0.009121
FBBFNT_EGP&PSOUM-IHS	**0.008773**

It is clear from the results of Table 3, that the FBBFNT_EGP&PSOUM-IHS model gives the best prediction rate for the Jenkins-Box time-series.

C. *Example 3: Lorenz chaotic time series prediction*

The Lorenz system is a model of fluid motion between a hot surface and a cool surface [33]. It is described by the following nonlinear ordinary differential equations:

$$\begin{cases} \dot{x} = \sigma(y-x) \\ \dot{y} = -y - xz + rx \\ \dot{z} = xy - bz \end{cases} \quad (10)$$

In this example, the x-component in the Lorenz equations is used as the time series. The data that describe the Lorenz attractor, were generated by solving the system of differential equations, with σ= 10, r=50 and b= 8/3. The data were used as inputs to the neural networks. The prediction is based on four past values$(x(t-4), x(t-3), x(t-2, xt-1)$ and thus the output pattern is $xt=f(xt-4, xt-3, x(t-2), x(t-1))$.

From 1000 generated observations, the first 500 data pairs of the series are used as the training set and the last 500 are employed as test series. The used Beta basis function sets to create an optimal FBBFNT model with EGP&PSOUM-IHS system is $NS = \{\beta_2/_3\} \cup \{x_1, x_2, x_3, x_4\}$, where x_i (i = 1, 2, 3, 4) denotes $x(t-4), x(t-3), x(t-2), x(t-1)$, respectively.After 14 global generations (G = 14), 227,904 global number of function evaluations, and 5115global iterations of the hybrid learning algorithm, an optimal FBBFNT model was obtained with RMSE 2.5000e-11. The RMSE value for validation data set is 2.6092e-11. In addition, the memory footprint of the source code is of the order of 2480 KB, knowing that the memory footprint of the input data is 40 KB. For the execution time is of the order of 979 seconds. The evolved FBBFNT_EGP&PSOUM-IHS architecture is as shown in Figure 8. Our method uses only two hidden function neurons for one hidden layers. The evolved FBBFNT_EGP&PSOUM-IHS output and the desired output are shown in Figure 9.

Table 4 illustrates the comparison of the proposed algorithm with other models according to the training and testing errors. This table shows the performance of our approach for Lorenz chaotic time series comparing with the other model.

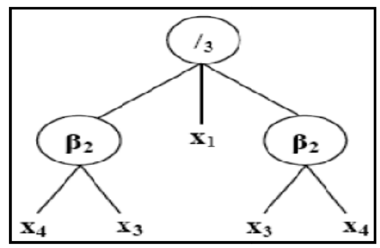

Fig. 8. Evolved FBBFNT_EGP&PSOUM-IHS architecture for Lorenz time-series prediction.

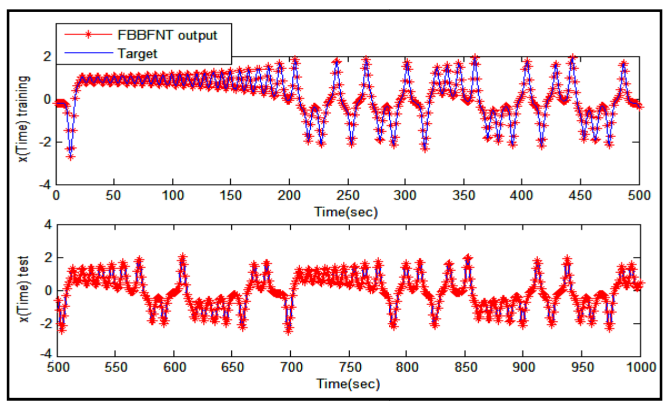

Fig. 9. Evolving FBBFNT_EGP&PSOUM-IHS output and the desired output for Lorenztime-series prediction.

TABLE IV. COMPARISON OF DIFFERENT MODELS OF LORENZ TIME SERIES PREDICTION.

Model	**RMSE Training**	**RMSE Testing**
UD-SVM [34]	-	0.290
MLP-EKF [36]	0.0145	0.0408
MLP-BLM [36]	0.0174	0.0314
RNN-BPTT [36]	0.0227	0.0436
RNN-RTRL [36]	0.0229	0.0420
RNN-EKF [36]	0.0182	0.0352
RBLM-RNN [36]	0.0182	0.0304
LNF [28]	0.0039	0.0081
ABC_BBFNN [35]	-	0.076
FBBFNT_EGP&PSOUM-IHS	**2.5000e-11**	**2.6092e-11**

D. Example 4: Nonlinear system identification

In this example, the nonlinear system [29] to be identified is expressed by:

$$y_p(t+1) = \frac{y_p(t)[y_p(t-1)+2][y_p(t)+2.5]}{8.5+[y_p(t)]^2+[y_p(t-1)]^2} + u(t) \quad (11)$$

Where $y_p(t)$ is the output of the system at the t^{th} time step and $u(t)$ is the plant input which is uniformly bounded in the region [-2, 2]. The identification model is as follows:

$$y_p(t+1) = f(y_p(t), y_p(t-1)) + u(t) \quad (12)$$

Where $f(y_p(t);y_p(t\text{-}1))$ is the nonlinear function of $y_p(t)$ and $y_p(t\ \text{-}1)$ that will be input of our model and related works; and $y_p(t+1)$ will be the output from the neural models.

In this example, 500 data samples are used for training and 500 data samples are used for testing the performance of the evolved model. After the training is over, the identifier's prediction ability has been tested for the input calculated as following:

$$u(t) = \begin{cases} 2\cos(2\pi t/100)\ if t \leq 200 \\ 1.2\sin(2\pi t/20)\ if\ 200 < t \leq 500 \end{cases} \quad (13)$$

The used Beta basis function sets to create an optimal FBBFNT model with EGP&PSOUM-IHS system is $NS = \{\beta_2/_3\} \cup \{x_1, x_2\}$, where x_i (i = 1, 2) denotes, $y_p(t)$ and $y_p(t\text{-}1)$, respectively.

After 10 global generations (G = 10), 245,126 global number of function evaluations, and 5,469 global iterations of the hybrid learning algorithm, an optimal FBBFNT model was obtained with RMSE 1.0464e-12. The RMSE value for identification data set is 1.5441e-12. In addition, the memory footprint of the source code is of the order of 9092 KB, knowing that the memory footprint of the input data is 24 KB. For the execution time is of the order of 2533 seconds.

The evolved FBBFNT_EGP&PSOUM-IHS architecture is as shown in Figure 10. Our method uses only two hidden function neurons for one hidden layers. The evolved FBBFNT_EGP&PSOUM-IHS output and the desired output are shown in Figure 11. A comparison result of different methods for forecasting Lorenz data is shown in Table 5.Results show that applying the FBBFNT_EGP&PSOUM-IHS for the nonlinear plant identification improves the generalization error.

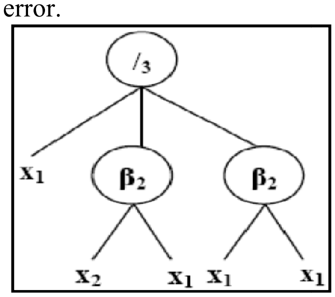

Fig. 10. Evolved FBBFNT_EGP&PSOUM-IHS architecture for nonlinear plant identification.

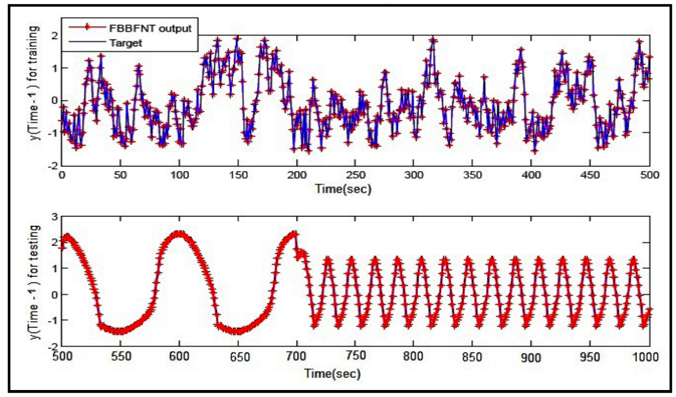

Fig. 11. Evolving FBBFNT_EGP&PSOUM-IHS output and the desired output for nonlinear plantidentification.

TABLE V. COMPARISON FBBFNT_EGP&PSOUM-IHS WITH OTHER METHODS FOR NONLINEAR PLANTIDENTIFICATION.

Model	RMSE Training	RMSE Testing
ODE [37]	0.019	0.1137
HMDDE [23]	0.019	0.110
FBBFNT_EGP&PSOUM-IHS	**1.0464e-12**	**1.5441e-12**

VI. CONCLUSIONS

In this paper, a new PSO-based update memory for Improved Harmony Search (PSOUM-IHS) algorithm is introduced to evolve the parameters of the Flexible Beta Basis Function Neural Tree (FBBFNT) model. These parameters include the parameters of Beta basis function nodes and connected weights. The PSOUM-IHS algorithm is combined with the Extended Genetic Programming for FBBFNT's structure optimization. This combination can successfully optimize simultaneously the structure and the parameters of the FBBFNT. The results show that the FBBFNT_EGP&PSOUM-IHS method can effectively predict nonlinear systems such as Mackey-Glass chaotic time series, Jenkins–Box time series, Lorenz chaotic time series, and nonlinear plant identification.

ACKNOWLEDGMENTS

The authors would like to acknowledge the financial support of this work by grants from General Direction of Scientific Research (DGRST), Tunisia, under the ARUB program. This work was also supported in the framework of the IT4 Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070 by operational programme 'Research and Development for Innovations' funded by the Structural Funds of the European Union and state budget of the Czech Republic, EU.

REFERENCES

[1] Z.W. Geem, J.H. Kim and G.V. Loganathan,"A new heuristic optimization algorithm: Harmony search", Simulation, vol. 76, pp. 60-68, 2001.

[2] K.S. Lee, Z.W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice", Comput. Methods Appl. Mech. Engrg., vol. 194, pp. 3902–3933, 2005.

[3] J.H. Holland, “Adaptation in Natural and Artificial Systems”, University of Michigan Press, Ann Harbor, 1992.

[4] P. Tangpattanakul and P. Artrit, "Minimum-time trajectory of robot manipulator using Harmony Search algorithm", 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2009), Pattaya, Thailand, 2009.

[5] R. Forsati, A.T. Haghighat, and M. Mahdavi, "Harmony search based algorithms for bandwidth-delay-constrained least-cost multicast routing", Computer Communications, vol. 31, pp. 2505-2519, 2008.

[6] H. Ceylan, H. Ceylan, S. Haldenbilen, and O. Baskan, "Transport energy modeling with metaheuristic harmony search algorithm, an application to Turkey", Energy Policy, vol. 36, pp. 2527-2535, 2008.

[7] X. Yao, “Evolution of connectionist networks”, Int. Symp. AI, Reasoning and Creativity, T. Dartnall, Ed. Queensland, Australia: Griffith Univ., pp. 49–52, 1991.

[8] X. Z. Gao, J. Wang, J. M. A. Tanskanen, R. Bie, and P. Guo, “BP Neural Networks with Harmony Search Method-based Training for Epileptic EEG Signal Classification”, Eighth International Conference on Computational Intelligence and Security, pp. 252 – 257, Guangzhou ,2012.

[9] A. Kattan, R. Abdullah, and R.A. Salam, “Harmony Search Based Supervised Training of Artificial Neural Networks”, International Conference on Intelligent Systems Modelling and Simulation, pp. 105 - 110, Liverpool, 2010.

[10] A. Kattan and R. Abdullah, “Training Feed-forward Artificial Neural Networks for Pattern-Classification Using The Harmony Search Algorithm”, The second International Conference on Digital Enterprise and Information Systems, pp. 84-97, Malaysia, 2013.

[11] J. Di, and N. Wang, “Harmony Search Algorithm with Chaos for Training RBFNN”, Journal of Software, vol. 8, No. 9, pp. 2231-2237, 2013.

[12] S.Bouaziz, H.Dhahri, Adel M. Alimi, “Evolving Flexible Beta Operator Neural Trees (FBONT) for Time Series Forecasting”, T. Hung et al. (Eds.) : 19th International Conference in neural information Processing(ICONIP’12), Proceedings, Part III, Series: Lecture Notes in Computer Science, Doha-Qatar,vol. 7665, pp. 17-24,2012.

[13] S.Bouaziz, H.Dhahri, Adel M. Alimi, A. Abraham, “A hybrid learning algorithm for evolving Flexible Beta Basis Function Neural Tree Model”, Neurocomputing, vol. 117, pp. 107–117, 2013.

[14] S. Bouaziz, Adel M. Alimi and A. Abraham,“Extended Immune Programmingand Opposite-based PSO for Evolving Flexible Beta Basis Function Neural Tree”, IEEE International Conference on Cybernetics, pp. 13 –18, Lausanne Switzerland,13-15 June 2013.

[15] S. Bouaziz, Adel M. Alimi and A. Abraham, “Evolving Flexible Beta BasisFunction Neural Tree for nonlinear systems”, International Joint Conference on NeuralNetworks, Dallas Texas, 4-9 August 2013.

[16] Adel M. Alimi, “The Beta Fuzzy System: Approximation of Standard Membership Functions”, in Proc. 17eme Journees Tunisiennes d'Electrotechnique et d'Automatique: JTEA'97,Nabeul, Tunisia, vol. 1, pp. 108-112, 1997.

[17] Adel M. Alimi, “The Beta System: Toward a Change in Our Use of Neuro-Fuzzy Systems”, International Journal of Management, Invited Paper, June, pp. 15-19, 2000.

[18] C. Aouiti, Adel M. Alimi, K. Karray, A. Maalej, “The design of bate basis function neural network and beta fuzzy systems by a hierarchical genetic algorithm”, fuzzy Sets and Systems, vol. 154, pp. 251-274, 2005.

[19] J. Yu and P. Guo,“Improved PSO algorithm with harmony search for complicatedfunction optimization problems”, In Proceedings of the 9thinternational conferenceon Advances in Neural Networks - Volume Part I, ISNN’12, pp. 624–632,Shenyang, China, 2012.

[20] M. Mahdavi, M. Fesanghary and E. Damangir,“An improved harmony search algorithm for solving optimization problems”, Applied Mathematics and Computation, vol. 188, no. 2, pp. 1567–1579, May 2007.

[21] M. Ammar, S. Bouaziz, Adel M. Alimi and A. Abraham,“Hybrid Harmony Search algorithm for Global Optimization”, Fifth World Congress on Nature and BiologicallyInspired Computing, pp. 69–75, Fargo-USA, 12-14 August 2013.

[22] L. Glass and M. C. Mackey,“Pathological physiological conditions resulting from instabilities in physiological control systems”, Annals of the New YorkAcademy of Sciences, vol. 316, pp. 214–235, 1979.

[23] H. Dhahri, Adel M. Alimi, A. Abraham,“Hierarchical multi-dimensional differential evolution for the design of beta basis function neural network”, Neurocomputing, vol. 79, pp.131-140, 2012.

[24] Y. Chen, B. Yang, J. Dong, A. Abraham, “Time-series forecasting using flexible neural tree model”, Information Sciences, vol. 174, pp. 219–235, 2005.

[25] C.Aouiti, , Adel M. Alimi, A.Maalej, “A Genetic Designed Beta Basis Function Neural Net-works for approximating of multi-variables functions”, in Proc. Int. Conf. Artificial Neural Nets and Genetic Algorithms Springer Computer Science, Prague, Czech Republic, pp. 383-386, 2001.

[26] C.F.Juang,C.M. Hsiao, C.H. Hsu,“Hierarchical Cluster-Based Multispecies Particle-Swarm optimization for Fuzzy-System Optimization”, IEEE Transactions on Fuzzy Systems, vol. 18(1),pp. 14-26, 2010.

[27] C.G. Coy and D. Kaur, “Improving evolutionary training for Sugeno Fuzzy Inference Systems using a Mutable Rule Base”, 2010 Annual Meeting of the North American, Fuzzy Information Processing Society (NAFIPS), pp. 1–6, 12-14 July, 2010.

[28] A. Miranian and M. Abdollahzade,“Developing a LocalLeast-Squares Support Vector Machines-Based Neuro-Fuzzy Model for Nonlinear andChaotic Time Series Prediction”, IEEE Transactions on Neural Networks and LearningSystems, vol. 24, no. 2, pp. 207–218, 2013.

[29] G.E.P.Box, G.M. Jenkins, “Time SeriesAnalysis--Forecasting and Control”, Holden Day, SanFrancisco, CA, 1976.

[30] J.Nie, “Constructing fuzzy model by self-organising counter propagation network”, IEEE Transactions on Systems Man and Cybernetics 25, pp. 963–970, 1995.

[31] Y. Chen, B. Yang, J. Dong, “Evolving Flexible Neural Networks Using Ant Programming and PSO Algorithm”, International Symposium on Neural Networks (ISNN’04), Lecture Notes on Computer Science 3173, pp.211-216, 2004.

[32] J.-S.R. Jang, C.-T.Sun, E.Mizutani, “Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence”, Prentice-Hall, Upper Saddle River, NJ, 1997.

[33] E.N. Lorenz, “Deterministic non-periodic flows”, J. Atm. Sci., vol. 20 pp. 130–141, 1963.

[34] C. Xiang, W. Zhou, Z. Yuan, Y. Chen and X. Xiong,“A new parameters joint optimization method of chaotic time series prediction”, International Journal of thePhysical Sciences, vol. 6, no. 10, pp. 2565–2571, May 2011.

[35] H. Dhahri, Adel M. Alimi,“Designing Beta Basis Function Neural Network for Optimization Using Artificial Bee Colony (ABC)”, International Joint Conference on Neural Networks, Brisbane, pp. 2161-4393, 2012.

[36] D.T. Mirikitani and N.I. Nikolaev,“Recursive Bayesian recurrentneural networks for time-series modeling”, IEEE Transactions on Neural Networks,vol. 21, no. 2, pages 262–274, 2010.

[37] B. Subudhi and D. Jena,“A differential evolution based neural networkapproach to nonlinear system identification”, Applied Soft Computing, vol. 11, no. 1,pp. 861–871, January 2011.

[38] J. H. Holland,“Adaptation in natural and artificial systems”, University ofMichigan Press, Ann Arbor, MI, USA, 1975.

[39] Y. Jarraya, S. Bouaziz, Adel M. Alimi and A. Abraham,“The AdaptiveChemotactic Foraging with Differential Evolution algorithm”, Fifth World Congress onNature and Biologically Inspired Computing, Fargo-USA, pp. 63–68, 2013.

[40] Y. Jarraya, S. Bouaziz, Adel M. Alimi and A. Abraham,“A Hybrid ComputationalChemotaxis in Bacterial Foraging Optimization Algorithm for Global NumericalOptimization”, IEEE International Conference on Cybernetics, Lausanne Switzerland,pp. 213 –218, 2013.