Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2018) 26: 2003 – 2015
© TÜBİTAK
doi:10.3906/elk-1712-391

An efficient recurrent fuzzy CMAC model based on a dynamic-group–based hybrid evolutionary algorithm for identification and prediction applications

Chin-Ling LEE[1], **Cheng-Jian LIN**[2,*]
[1]Department of International Business, National Taichung University of Science and Technology, Taichung, Taiwan
[2]Department of Computer Science & Information Engineering, National Chin-Yi University of Technology, Taichung, Taiwan

Received: 30.12.2017 • **Accepted/Published Online:** 03.04.2018 • **Final Version:** 27.07.2018

Abstract: This article presents an efficient TSK-type recurrent fuzzy cerebellar model articulation controller (T-RFCMAC) model based on a dynamic-group–based hybrid evolutionary algorithm (DGHEA) for solving identification and prediction problems. The proposed T-RFCMAC model is based on the traditional CMAC model and the Takagi–Sugeno–Kang (TSK) parametric fuzzy inference system. Otherwise, the recurrent network, which imports feedback links with a receptive field cell, is embedded in the T-RFCMAC model, and the feedback units are used as memory elements. The DGHEA, which is a hybrid of the dynamic-group quantum particle swarm optimization (QPSO) and the Nelder–Mead method, is proposed for adjusting the parameters of the T-RFCMAC model. In DGHEA, an entropy-based grouping technique is adopted to improve the searching capability and the convergent speed of quantum particles swarm optimization. Experimental results show that the proposed DGHEA-based T-RFCMAC model is more effective at identification and prediction than other models.

Key words: Fuzzy cerebellar model articulation controller, entropy, Nelder–Mead, particle swarm optimization, prediction, identification

1. Introduction

For nonlinear system processing, neural networks or neural fuzzy networks [1–3] are the most commonly used model. If a feedforward network is applied in the dynamic system, we first need to obtain the delay numbers of outputs and inputs [3]. In this situation, the accurate order of the dynamic system is usually not clear. To solve this problem, recurrent networks [4,5] are adopted for processing the dynamic system. In the present study, we present a novel recurrent network for solving different problems.

The cerebellar model articulation controller (CMAC) model was proposed by Albus [6,7]. It is a simple network architecture with a high convergence rate, high learning speed, good generalization capability, ease of hardware implementation, etc. The CMAC network has been employed in various areas successfully, such as robot control [8], pattern recognition [9], and signal processing [10]. However, Albus's CMAC network still has three major limitations. First, it is difficult to select the memory structure parameters. While the common CMAC network has a constant value allocated to each hypercube, the derivative information is not maintained and the data for a quantized state are constant. In order to solve this problem, inconstant differentiable basis functions are used, such as fuzzy membership functions by Jou [11] and spline functions by Lane et al.

[*]Correspondence: cjlin@ncut.edu.tw

[12]. Second, for solving high-dimensional problems, the CMAC model requires a great amount of memory space. Even though some approaches have been used to deal with this problem, such as hashing functions, the "collision" problem and the complexity problem in high dimensions still exist. The third limitation is that the CMAC model needs a more rigorous theory for function approximation. Several modified CMAC models were proposed to overcome this problem by using fuzzy concept or B-spline functions [13,14]. The basic function in the CMAC model is a binary case function and the input vectors produce the same output in the same hypercube from the network. That is why the output of the CMAC model is usually not as smooth as the target function. Many researchers combine the fuzzy concept into the CMAC network [14,15]. That is, they replace the basic functions with fuzzy membership functions. The model is called fuzzy CMAC (FCMAC). In this study, a novel TSK-type recurrent fuzzy cerebellar model articulation controller (T-FRCMAC) is proposed.

In general, a backpropagation (BP) algorithm [16] is used in parameter learning and has powerful training ability to employ with a forward structure on networks. Because the BP algorithm depends on the gradient descend method, it has some disadvantages, such as the lower convergent speed and being easily trapped into local minimum. Thus, many evolutionary algorithms have been proposed to conquer the above-mentioned disadvantages, such as genetic algorithm (GA) [17], differential evolution (DE) [18], and particle swarm optimization (PSO) [19]. Recently, PSO has been commonly used to perform parameter learning in various models. PSO was introduced by Eberhart and Kennedy and inspired by the social behavior of animals, such as bird flocking, fish schooling, and the swarm theory. Because PSO has some attractive characteristics, such as simple implementation, high speed convergence, and direct logic, it has been widely developed in many areas. Although traditional PSO has a fast convergence speed, it is also easily trapped into local minima. Recently, several hybrid models have been studied to improve the disadvantage of the traditional PSO. Sun et al. [20] proposed a quantum-behaved PSO algorithm that guaranteed theoretically finding an optimal solution in search space. Zhang et al. [21] proposed a hybrid algorithm by integrating GA and gravitational search algorithm (GSA) to avoid premature convergence and to improve the search ability. In [21], crossover and mutation operators are used for jumping out of the local optima. Mohan and Albert [22] proposed a hybrid algorithm that integrates GA and PSO algorithms. The GA will be the main optimizer and the PSO will be used to guide the GA to locate optimal solutions quickly and effectively. An improved hybrid method applying both PSO and GA was also developed in [23]. To avoid premature convergence of PSO, Idoumghar et al. [24] presented a hybrid evolutionary algorithm based on the idea that PSO ensures fast convergence, while simulated annealing brings the search out of local optima because of its strong local-search ability. Abadin and Rezaei [25] presented a combinational method including PSO and continuous ant colony optimization in order to improve the search process. Singh et al. [26] developed a combination of mean Gbest particle swarm optimization (MGBPSO) and GSA. The basic inspiration is to integrate the ability of exploitation in MGBPSO with the ability of exploration in GSA to synthesize the strength of both approaches. The presented approach has an automatic balance between local and global searching abilities.

Because the above-mentioned algorithms do not change the characteristics of the original evolutionary algorithms, they also have problems with premature convergence and falling into local optima. To overcome the above-mentioned drawbacks, an efficient TSK-type recurrent fuzzy cerebellar model articulation controller (T-RFCMAC) model with a dynamic-group–based hybrid evolutionary algorithm (DGHEA) is proposed for solving prediction and identification problems in this study. The proposed DGHEA is a hybrid algorithm, integrating the dynamic-group quantum particle swarm optimization (QPSO) and the Nelder–Mead (NM) method. Unlike traditional QPSO, the formation of the dynamic group in the proposed improved QPSO is not only used for particle-position update, but also for diversity of particle reference. An entropy-based criterion is used to determine whether or not the particle belongs to the group.

2. The structure of the T-RFCMAC model

In this section, a TSK-type recurrent fuzzy cerebellar model articulation controller (T-RFCMAC) model is proposed. The architecture of the T-RFCMAC model is shown in Figure 1, which includes five layers: the input space partition, the recurrent unit, the association space, the TSK-type output, and the defuzzification.

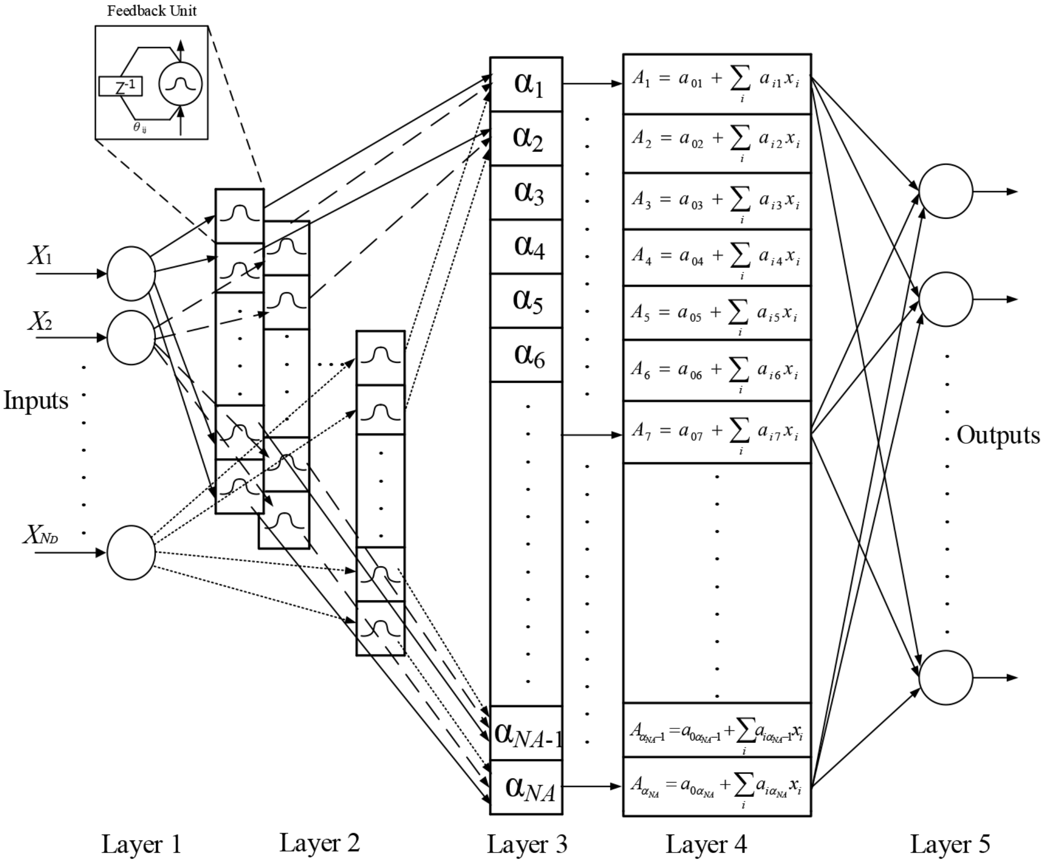
Feedback Unit
Z^{-1}
θ_{ij}
X_1
X_2
Inputs
X_{N_D}
α_1
α_2
α_3
α_4
α_5
α_6
α_{NA-1}
α_{NA}
$A_1 = a_{01} + \sum_i a_{i1} x_i$
$A_2 = a_{02} + \sum_i a_{i2} x_i$
$A_3 = a_{03} + \sum_i a_{i3} x_i$
$A_4 = a_{04} + \sum_i a_{i4} x_i$
$A_5 = a_{05} + \sum_i a_{i5} x_i$
$A_6 = a_{06} + \sum_i a_{i6} x_i$
$A_7 = a_{07} + \sum_i a_{i7} x_i$
$A_{\alpha_{NA}-1} = a_{0\alpha_{NA}-1} + \sum_i a_{i\alpha_{NA}-1} x_i$
$A_{\alpha_{NA}} = a_{0\alpha_{NA}} + \sum_i a_{i\alpha_{NA}} x_i$
Outputs
Layer 1
Layer 2
Layer 3
Layer 4
Layer 5

Figure 1. The architecture of the T-RFCMAC model.

The proposed T-RFCMAC model involves the Gaussian basic function as the receptive field functions and the TSK-type output as the linear parametric function of the model output for learning. The one-dimensional Gaussian basic function is given as follows:

$$\mu\left(x\right) = e^{-\left(\frac{x-m}{\sigma}\right)^2} \tag{1}$$

where x represents a specific input state and m and σ represent the corresponding center and variance. A N_D-dimension Gaussian basic function is given as follows:

$$\alpha_j = \prod\nolimits_{i=1}^{N_D} e^{-\left(\frac{x_i-m_{ij}}{\sigma_{ij}}\right)^2} \tag{2}$$

where Π denotes the product operation, α_j represents the jth component of the association vector, x_i is the input value of the ith dimension for a specific input state x, and m_{ij} and σ_{ij} represent the center and the variance of the receptive field functions. Furthermore, the input of each membership function for discrete time $t$is given as follows:

$$r_{ij}\left(t\right) = x_i\left(t\right) + r_{ij}(t-1)\theta_{ij}, \tag{3}$$

where θ_{ij} represents the recurrent weight of the feedback unit, $r_{ij}(t)$ denotes the input of each membership function, x_i denotes the input value of the ith dimension for a specific input state x, and $r_{ij}(t-1)$ denotes the storage blocks that store the last time information of the model. Each hypercube element of the receptive field functions is deduced to generate a partial fuzzy output by applying its corresponding association vector value as matching degree of input; the formula is given as follows:

$$a_{0j} + \sum\nolimits_{i=1}^{N_D} a_{ij}x_i \tag{4}$$

where a_{0j} and a_{ij} represent the scalar values, N_D is the number of the input dimensions, and x_i is the ith input dimension. After employing Eq. (4), the single fuzzy output is defuzzified by the centroid of area approach into a scalar output y and the real output y is calculated as follows:

$$y = \frac{\sum_{j=1}^{N_c} \alpha_j \left(a_{0j} + \sum_{i=1}^{N_D} a_{ij}x_i\right)}{\sum_{j=1}^{N_c} \alpha_j}, \tag{5}$$

where N_c represents the number of hypercube cells.

3. The proposed hybrid evolutionary algorithm

In this section, a hybrid evolutionary algorithm is proposed. The proposed algorithm consists of the QPSO and the NM algorithms.

3.1. Review of the NM method

The NM method, originally proposed by Nelder and Mead [27], is a simplex method for finding a local minimum in an optimal problem. A "simplex" is a convex hull that is structured by n+ 1 pointers $(X_0X_1X_2,\ldots,X_{n-1}X_n)$ in n dimensions. Because the NM method has few parameters and is easy to comprehend and implement, it is an effective method in various fields, such as engineering, biology, chemistry, and physics. The NM method includes four parameters, namely *reflection* (α), *expansion* (β), *contraction* (γ), and *shrinkage* (δ).

In each iteration, we start the process with n+ 1 points and the objective function is optimized at each point of the simplex. After each transformation, a better point will replace the current worst point. The NM process is given as follows:

Step 1: Set the basic parameters: *reflection* (α), *expansion* (β), *contraction* (γ), and *shrinkage* (δ).

Step 2: Order the objective values at the points as $f\left(X_0\right) \leq f\left(X_1\right) \leq f\left(X_2\right) \cdots \leq f\left(X_n\right)$

Step 3: *Center Point M*: The centroid $M$of the points excluding $X_{n+1}$is calculated and the formula is given as follows:

$$M{=}\frac{1}{n}\sum\nolimits_{i=0}^{n-1} X_i \tag{6}$$

where n denotes the number of points and X_i is the ith point.

Step 4: *Reflection Point* X_r: Generate the reflection point X_r.

$$X_r = M + \alpha(M - X_n) \tag{7}$$

Step 5: According to Eq. (7), three cases are described as follows:

Case 1: If $f(X_0) \leq f(X_r) \leq f(X_n)$, we set $X_n = X_r$.

Case 2: If $f(X_r) < f(X_0)$, we use Eq. (8) to calculate the expansion point X_e.

$$X_e{=}X_r + \gamma(X_r - M) \tag{8}$$

If $f(X_e) < f(X_0)$, we set $X_n = X_e$. Otherwise, we set $X_n = X_r$.

Case 3: If $f\,(X_r) > f(X_n)$, we use Eq. (9) to calculate the contraction point X_c.

$$X_c{=}M + \beta(X_r - M) \tag{9}$$

If $f\,(X_c) < f(X_n)$, we set $X_n = X_c$ Otherwise, we use Eq. (10) to replace all the points X_i and generate a new simplex.

$$X_i = X_0 + \delta\,(X_i - X_0)\ \ for\, i{\in}\,\{1, 2, 3, \cdots, n\} \tag{10}$$

Step 6: If the terminal criterion is satisfied, stop the process. Otherwise, move to Step 2. According to the above process, the NM method is a speedy algorithm for searching a local minimum without complex formulas.

3.2. The proposed DGHEA

In this subsection, a new DGHEA is introduced. The proposed DGHEA contains a swarm of particles. Each particle represents a parameter solution vector of the T-RFCAMC model. In traditional QPSO, the initial P_p particles are randomly generated to form a swarm. In the proposed DGHEA, the formation of the dynamic group is not only used for particle-position update, but also for diversity of particle references. The number of groups in the proposed DGHEA is not fixed, and it can confirm that the particles are not classified in unreasonable group. For group formation, the fitness values of all particles are ordered from the greatest to least. That is, the first particle has the best (the greatest) fitness value in the sorted swarm and becomes the group leader (L_g) in the first group. An entropy-based criterion is adopted as a measure to evaluate the group membership of the particles in sequence. Entropy is defined as follows:

$$EM = -\sum_{i=1}^{P-1} D_i log_2 D_i \tag{11}$$

$$D_i = L_1 - P_i \tag{12}$$

where p represents the number of particles, P_i denotes the particles in sequence, and D_i is the distance between the group leader and other particles. In order to facilitate calculation, D_i is normalized to $D_i \in [0, 1]$. By using Eq. (11), the threshold EM is used to determine whether a new group is generated or not. If D_i is larger than the threshold EM, set the ith particle as a new leader and form a new group. The schematic of dynamic groups is shown in Figure 2.

QPSO was first inspired by analysis of the convergence of the traditional PSO and quantum systems. Sun et al. [20] proposed quantum-behaved PSO. The updated position of each particle is defined as follows:

$$P_i = \varphi \times Pbest_i + (1 - \varphi)Gbest \tag{13}$$

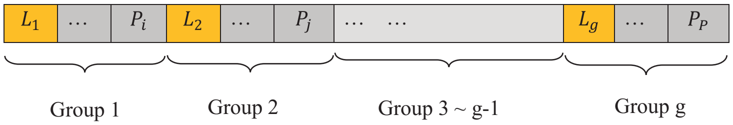

Figure 2. The schematic of dynamic groups.

$$Mbest = \frac{1}{P}\sum_{i=1}^{P} Pbest_i \tag{14}$$

$$x_i\,(t+1) = P_i \pm \beta\,|Mbest - x_i(t)| \times \ln\left(\frac{1}{u}\right),\ i = 1, 2, \cdots, P \tag{15}$$

where P_i represents local attractor, $Mbest$ is the mean best position defined as the mean of all the best positions of a swarm, φ and u are random numbers with uniform distribution on $[0, 1]$ respectively, and P is the number of particles. β is called the contraction–expansion coefficient and can be tuned to control the convergence speed of the algorithm. The symbol $\pm$ is used to determine the minus sign and the plus sign. If the random value is larger than 0.5, the minus sign $(-)$ is selected; otherwise, the plus sign $(+)$ is selected. In the DGHEA, particles do not refer to *Pbest* position. In contrast, the group leader is used to substitute the position of *Pbest* and all the particles have suitable reference points. The new formula is given as follows:

$$P_i = \varphi \times L_g + (1-\varphi)Gbest \tag{16}$$

The flowchart of the DGHEA algorithm is shown in Figure 3. The proposed DGHEA is described by the following steps:

Step 1: Initialization

$N+$ *1* particles are initialized with random positions in the N-dimensional problem and their fitness value evaluated. Each particle represents all control parameters of T-RFCMAC, which is shown in Figure 4. That is, this figure shows the adjustable parameters in the T-RFCMAC model that are coded into a particle. The jth fuzzy hypercube cell contains the center m_{ij} and width σ_{ij} of the Gaussian membership function, the constants a_{0j} and a_{ij} of the corresponding memory of linear combination of input variables, and the recurrent weight of the feedback unit θ_{ij}.

Step 2: NM method

The fitness values of all particles are ordered from greatest to least. That is, the particles are given fitness values from the best (the greatest fitness value) to the worst (the smallest fitness value). Thus, the NM method is used to update the particle position.

Step 3: Group formation

According to the fitness values of all particles from the greatest to least, the distance values between the particle with the best fitness value and the others particles are calculated. Based on those distance values, an entropy-based criterion is used to determine whether or not the particle belongs to the group. The group leader is used to substitute the position of *Pbest*. That is, all of the particles in the swarm have a new reference point to search.

Step 4: Updating position of particle

The group leaders and the *Gbest* are used as suitable reference points for all the particles. The new formula is given by Eq. (16). The group leader can lead the particles to find the actual solution efficiently.

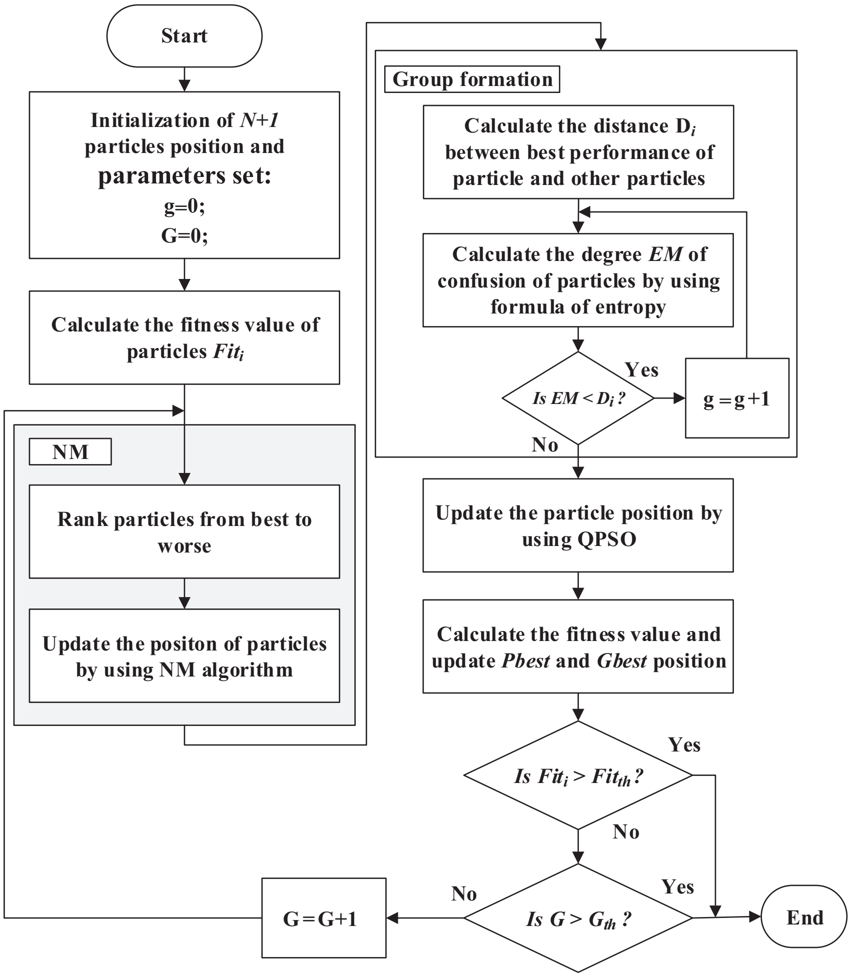
Start
Initialization of N+1 particles position and parameters set:
g=0;
G=0;
Calculate the fitness value of particles Fit_i
NM
Rank particles from best to worse
Update the positon of particles by using NM algorithm
Group formation
Calculate the distance D_i between best performance of particle and other particles
Calculate the degree EM of confusion of particles by using formula of entropy
Is $EM < D_i$?
Yes
g =g+1
No
Update the particle position by using QPSO
Calculate the fitness value and update Pbest and Gbest position
Is $Fit_i > Fit_{th}$?
Yes
No
Is $G > G_{th}$?
Yes
No
G=G+1
End

Figure 3. The DGHEA flowchart.

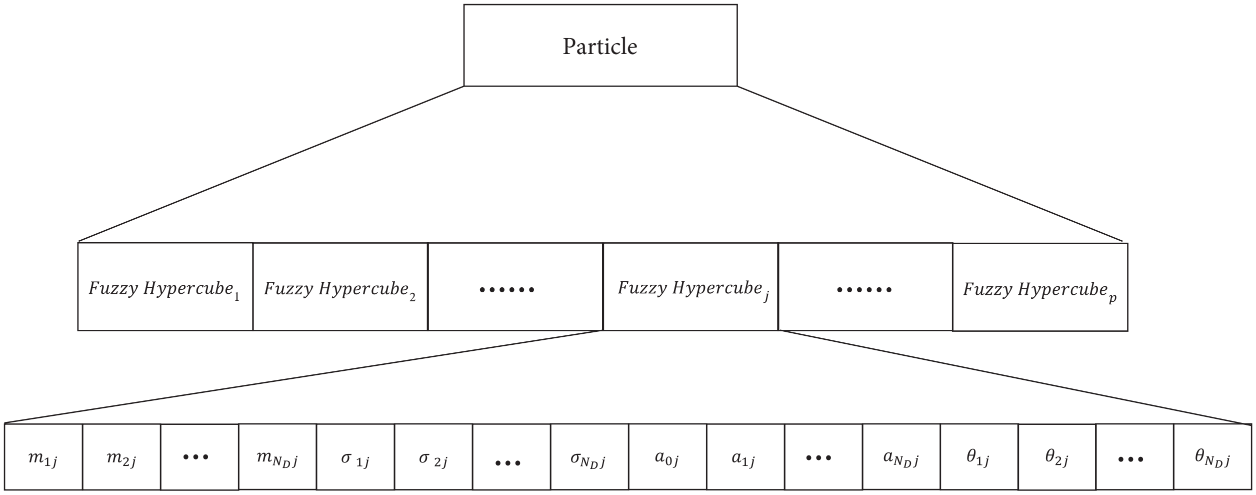

Figure 4. Coding the parameters of T-RFCMAC model into a particle.

Step 5: Updating *Pbest* (personal best) and *Gbest* (global best)

After updating the position of each particle, the *Pbest* and *Gbest* will be updated if necessary. First, the fitness values of each particle are compared with the personal best fitness value (*Pbest*). If the current fitness value is larger than *Pbest*, the *Pbest* will be replaced by the current fitness value and also the position of the *Pbest* in N-dimensional space will be updated. The updating approach of the global best fitness value (*Gbest*) is similar to the updating approach of *Pbest*.

Step 6: If the number of generations is satisfied, the algorithm will return to Step 2; otherwise, the algorithm will be terminated.

4. Simulation results

In this section, two examples are simulated. These simulation studies include the Mackey–Glass chaotic series prediction problem and the dynamic system identification problem. The chaotic time series and the dynamic system identification problems are difficult to predict and identify. Recently, several researchers [28–39] tried to improve the prediction and identification performance. Therefore, in the present study we compare the performance of the DGHEA-based T-RFCMAC model with those of other methods [28–39]. The root mean squared error (RMSE) is used for performance evaluation in this study. Two simulation comparisons are considered. One is the performance comparison of various evolutionary learning algorithms and the other is that of various models. All the programs were developed by using Visual Studio 2013 C++ software on an Intel i7-870 2.93 GHz personal computer.

4.1. Prediction of Mackey–Glass chaotic time series

The Mackey–Glass chaotic time series $x(t)$ in consideration here is generated from the following delay differential equation:

$$\frac{dx(t)}{dt} = \frac{0.2x(t - \tau)}{1 + x^{10}(t - \tau)} - 0.1x\left(t\right), \tag{17}$$

where the initial value is given as $\tau = 17$ and $x\left(0\right) = 1.2$. Crowder extracted 1000 input–output data pairs $\left\{x, y^d\right\}$ using four past values of $x(t)$, i.e.

$$\left[x\left(t-18\right), x\left(t-12\right), x\left(t-6\right), x\left(t\right); x\left(t+6\right)\right] \tag{18}$$

Four inputs of the T-RFCMAC model correspond to these value of $x(\Delta t)$, and one output represents the value $x(t+\Delta t)$, where Δt is a time interval into the future. The first 500 pairs are used for training data, whereas the remaining 500 pairs are used for testing data. In this experiment, the parameters are set and shown in Table 1. Figure 5 shows the prediction results of the actual output and the output of T-RFCMAC model with DGHEA. Figure 6 describes the learning curves of the T-RFCMAC model with various learning algorithms, including the proposed DGHEA, QPSO [20], and NM [27]. Table 2 shows the performance comparison of the T-RFCMAC model with various learning algorithms. Table 3 lists the performance comparison of various models, including the proposed T-RFCMAC model with DGHEA, P-FCMAC [28], GEFREX [29], ANFIS [30], Kim and Kim [31], DENFIS [32], SEFC [33], and NFIS-SEELA [34]. Experimental results show that the T-RFCMAC model with DGHEA performs better than other methods.

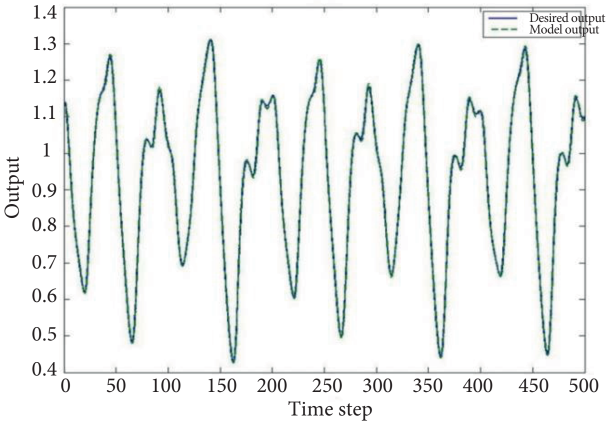

Figure 5. Prediction results of the actual output and the output of T-RFCMAC model with DGHEA.

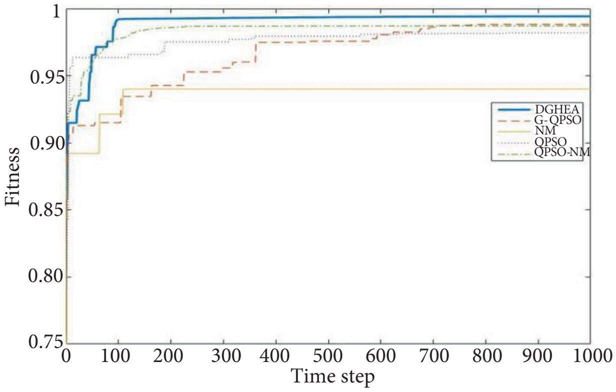

Figure 6. Learning curves of the T-RFCMAC model with various evolutionary learning algorithms.

Table 1. The parameters of the proposed method for chaotic time series prediction.

Parameter	Generation	Number of particles	Number of fuzzy hypercubes	NM parameter $\alpha, \beta, \gamma, \delta$
Value	1000	52	3	1, 0.5, 2, 0.5

Table 2. Comparison results of the T-RFCMAC model with various evolutionary learning algorithms for chaotic time series prediction.

Method RMSE	Proposed method	QPSO[20]	Group-based QPSO	QPSO-NM	NM[27]
Best	0.0030	0.0057	0.0089	0.0065	0.0631
Worst	0.0087	0.0182	0.0149	0.0176	0.0827
Average	0.0057	0.0137	0.0125	0.0121	0.0727
Standard error	0.0046	0.0030	0.0022	0.0027	0.0078

Table 3. Comparison results of various models for chaotic time series prediction.

Models	Proposed method	P-FCMAC [28]	GEFREX [29]	ANFIS [30]
RMSE	0.0030	0.0094	0.0061	0.007
Models	Kim and Kim [31]	DENFIS [32]	SEFC [33]	NFIS-SEELA [34]
RMSE	0.026	0.033	0.032	0.00675

4.2. Identification of dynamic system

In this example, the identification of the dynamic system is expressed as follows:

$$y_p\left(t+1\right) = 0.72 y_p\left(t\right) + 0.025 y_p\left(t-1\right) u\left(t-1\right) + 0.01 u\left(t-2\right) + 0.2 u\left(t-3\right) \tag{19}$$

The output of dynamic system depends on three last inputs and two last outputs. Two current variables, $y_p(t)$ and $u(t)$, are fed as inputs into the T-RFCMAC model. The input signal $u(t)$ is used to determine the results:

$$u(t) = \begin{cases} \sin\left(\frac{\pi t}{25}\right), & t < 250 \\ 1.0, & 250 \le t < 500 \\ -1.0, & 500 \le t < 750 \\ 0.3\sin\left(\frac{\pi t}{25}\right) + 0.1\sin\left(\frac{\pi t}{32}\right) + 0.6\sin\left(\frac{\pi t}{10}\right), & 750 \le t < 1000 \end{cases} \tag{20}$$

In this experiment, the adjustable parameters are set and shown in Table 4. Figure 7 displays the identification results of the actual output and the output of the T-RFCMAC model with DGHEA. Figure 8 shows the learning curves of the T-RFCMAC model with various learning algorithms, including the proposed DGHEA, QPSO [20], and NM [27]. Table 5 shows the performance comparison of the T-RFCMAC model with various learning algorithms. Table 6 lists the performance comparison of various models, including the proposed T-RFCMAC model with DGHEA, ERNN [35], RFNN [36], WRFNN [37], SRFNN [38], Feedforward T2-FNN, and RIFNN [39]. Experimental results show that the T-RFCMAC model with DGHEA also performs better than other models.

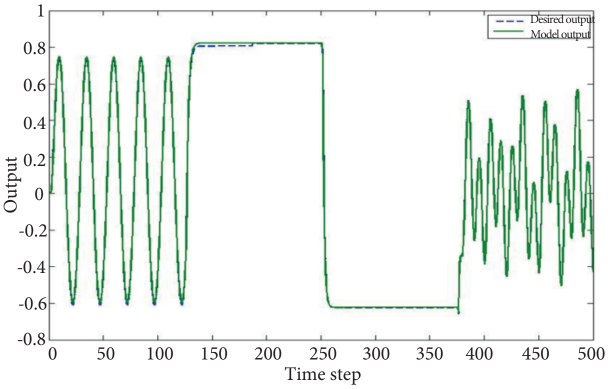

Figure 7. Identification results of the actual output and the output of the T-RFCMAC model with DGHEA.

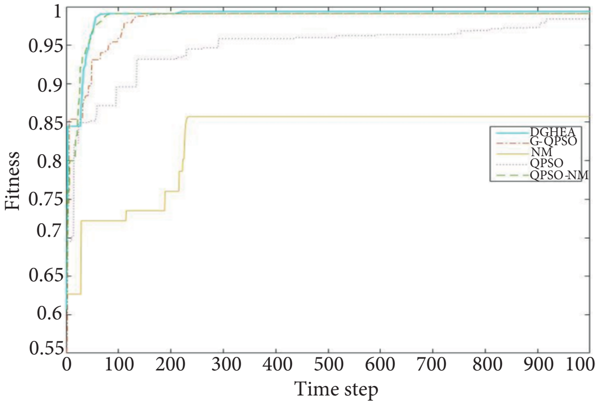

Figure 8. Learning curves of the T-RFCMAC model with various evolutionary learning algorithms.

Table 4. The parameters of the proposed method for dynamic system identification.

Parameters	Generation	Number of particles	Number of fuzzy hypercubes	NM parameter $\alpha, \beta, \gamma, \delta$
Value	1000	52	3	1, 0.5, 2, 0.5

Conclusion

This study proposes an efficient T-RFCMAC model with DGHEA for solving identification and prediction problems. The proposed DGHEA learning method is a hybrid of the modified QPSO and NM algorithms. The traditional QPSO is modified and added to the group concept to promote the reference information of particles and find the global optimal quickly. Two simulation studies included the dynamic system identification problem

Table 5. Comparison results of the T-RFCMAC model with various evolutionary learning algorithms for dynamic system identification.

Method RMSE	Proposed method	QPSO [20]	G-QPSO	QPSO+NM	NM [27]
Best	0.0063	0.0090	0.0090	0.0087	0.1663
Worst	0.0090	0.0305	0.0091	0.0090	0.6710
Average	0.0086	0.0119	0.009003	0.0090	0.4196
Standard deviation (STD)	0.00060267	0.0046	0.0000038352	0.000104	0.1383

Table 6. Comparison results of various models for dynamic system identification.

Models	Proposed method	ERNN [35]	RFNN [36]	WRFNN [37]
RMSE	0.0063	0.036	0.072	0.0574
Models	SRFNN [38]	Feedforward T2-FNN	RIFNN [39]	-
RMSE	0.02	0.0155	0.0125	-

and the Mackey–Glass chaotic series prediction problem. Simulation results show that the proposed T-FRCMAC model with DGHEA has a better RMSE performance and faster convergent speed than other methods.

Acknowledgment

The authors would like to thank the Ministry of Science and Technology of the Republic of China, Taiwan, for financially supporting this research under Contract No. MOST 106-2221-E-167-016.

References

[1] Lee CH, Chang FY, Lin CM. An efficient interval type-2 fuzzy CMAC for chaos time-series prediction and synchronization. IEEE T Cybernetics 2010; 44: 329-341.

[2] Lin CJ, Chen CH, Lin CT. Efficient self-evolving evolutionary learning for neuro-fuzzy inference systems. IEEE Trans Fuzzy Syst 2008; 16: 1476-1490.

[3] Lin CM, Li HY. TSK fuzzy CMAC-based robust adaptive backstepping control for uncertain nonlinear systems. IEEE T Fuzzy Syst 2012; 1: 1147-1145.

[4] Lin CJ, Wu CF, Lin HY, Yu CY. An interactively recurrent functional neural fuzzy network with fuzzy differential evolution and its applications. Sains Malaysiana 2015; 44: 1721-1728.

[5] Juang CF, Chen JS. Water bath temperature control by a recurrent fuzzy controller and its FPGA implementation. IEEE T Ind Electron 2006; 53: 22-22.

[6] Albus JS. A new approach to manipulator control: the cerebellar model articulation controller (CMAC). J Dyn Sys Meas Control 1975; 97: 220-227.

[7] Albus JS. Data storage in the cerebellar model articulation controller (CMAC). J Dyn Sys Meas Control 1975; 97: 228-233.

[8] Dierks T, Jagannathan S. Neural network output feedback control of robot formations. IEEE T Syst Man Cybern B Cybern 2010; 40: 383-399.

[9] Yumtao D, Liu L, Zhao X. Modeling of nonlinear parameters on ship with fuzzy CMAC neural networks. In: IEEE International Conference on Information and Automation (ICIA); 20–23 June 2010; pp. 2070-2075.

[10] Azimi-Sadjadi MR, Gao W, Vonder Haar TH, Reinke D. Temporal updating scheme for probabilistic neural network with application to satellite cloud classification-further results. IEEE T Neural Netw 2001; 12:1196-203.

[11] Jou CC. A fuzzy cerebellar model articulation controller. In: 1992 IEEE International Conference on Fuzzy Systems; 8–12 March 1992; San Diego, CA, USA: IEEE. pp. 1171-1178.

[12] Lane SH, Handelman DA, Gelfand JJ. Theory and development of higher-order CMAC neural networks. IEEE Control Systems 1992; 12: 23-30.

[13] Lee HM, Chen CM, Chen JM, Jou YL. An efficient fuzzy classifier with feature selection based on fuzzy entropy. IEEE T Syst Man Cybern B Cybern 2001; 31: 426-432.

[14] Guo C, Ye Z, Sun Z, Sarkar P, Jamshidi M. A hybrid fuzzy cerebellar model articulation controller based autonomous controller. Comput Electric Eng 2002; 28: 1-16.

[15] Chen JY, A VSS-type FCMAC controller. In: The 10th IEEE International Conference on Fuzzy Systems; 2–5 December 2001; pp. 872-875.

[16] Chen CH, Lin CJ, Lin CT. A functional-link-based neuro-fuzzy network for nonlinear system control. IEEE T Fuzzy Syst 2008; 16: 1362-1378.

[17] Tang A, Quek C, Ng G. GA-TSKfnn: parameters tuning of fuzzy neural network using genetic algorithms. Expert Syst Appl 2005; 29: 769-781.

[18] Ilonen J, Kamarainen JK, Lampinen J. Differential evolution training algorithm for feed-forward neural networks. J Neural Process Lett 2003; 17: 93-105.

[19] Hassanhosseini S, Taban MR, Abouei J. Indoor localization of wireless emitter using direct position determination and particle swarm optimization. Turk J Elec Eng & Comp Sci 2018; 26: 655-665.

[20] Sun J, Feng B, Xu WB. Particle swarm optimization with particles having quantum behavior. In: IEEE Proceedings of Congress on Evolutionary Computation; 19–23 June 2004; pp. 325-331.

[21] Zhang A, Sun G, Wang Z, Yao Y. A hybrid genetic algorithm and gravitational search algorithm for global optimization. Neural Netw World 2015; 25: 53-73.

[22] Mohan VJ, Albert TAD. Particle swarm optimization guided genetic algorithm: A novel hybrid optimization algorithm. International Journal of Engineering and Technology 2017; 9: 628-634.

[23] Bertram AM, Zhang Q, Kong SC. A novel particle swarm and genetic algorithm hybrid method for diesel engine performance optimization. Int J Engine Res 2016; 17: 732-747.

[24] Idoumghar L, Melkemi M, Schott R, Aouad MI. Hybrid PSO-SA type algorithms for multimodal function optimization and reducing energy consumption in embedded systems. Appl Comput Intell Soft Comput 2011: DOI: http://dx.doi.org/10.1155/2011/138078.

[25] Abadin MFH, Rezaei H. A hybrid model of particle swarm optimization and continuous ant colony optimization for multimodal functions optimization. Journal of Mathematics and Computer Science 2015; 15: 108-119.

[26] Singh N, Singh S, Singh SB. A new hybrid MGBPSO-GSA variant for improving function optimization solution in search space. Evol Bioinform 2017; 13: DOI: http: 10.1177/1176934317699855.

[27] Nelder JA, Mead R. A simplex method for function minimization. J Comput 1965; 7: 308-313.

[28] Lin CJ, Lee CY. A novel parametric fuzzy CMAC network and its applications. J Appl Soft Comput 2009; 9: 775-785.

[29] Russo M. Genetic fuzzy learning. IEEE T Evol Comput 2000; 4: 259-273.

[30] Jang JSR. ANFIS: adaptive-network-based fuzzy inference system. IEEE T Syst Man Cybern 1993; 23: 665-685.

[31] Kim D, Kim C. Forecasting time series with genetic fuzzy predictor ensemble. IEEE T Fuzzy Syst 1997; 5: 523-535.

[32] Kasabov NK, Song Q. DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. IEEE T Fuzzy Syst 2002; 10: 144-154.

[33] Juang CF, Lin JY, Lin CT. Genetic reinforcement learning through symbiotic evolution for fuzzy controller design. IEEE T Syst Man Cybern B Cybern 2000; 30: 290-302.

[34] Lin CJ, Chen CH, Lin CT. Efficient self-evolving evolutionary learning for neurofuzzy inference systems. IEEE T Fuzzy Syst 2008; 16: 1476-1490.

[35] Elman JL. Finding structure in time. Cognit Sci 1990; 14: 179-211.

[36] Lee CH, Teng CC. Identification and control of dynamic systems using recurrent fuzzy neural networks. IEEE T Fuzzy Syst 2000; 8: 349-366.

[37] Lin CJ, Chin CC. Prediction and identification using wavelet-based recurrent fuzzy neural networks. IEEE T Systems Man Cybern B Cybern 2004; 34: 2144-2154.

[38] Juang CF, Chiou CT, Lai CL. Hierarchical singleton-type recurrent neural fuzzy networks for noisy speech recognition. IEEE T Neural Netw 2007; 18: 833-843.

[39] Juang CF, Lin YY, Huang RB. Dynamic system modeling using a recurrent interval-valued fuzzy neural network and its hardware implementation. Fuzzy Sets Syst 2011; 179: 83-99.