Prediction of Time Sequence Using a TSK-Type Fuzzy Cerebellar Model Articulation Controller Network

Chin-Ling Lee and Cheng-Jian Lin

***Abstract*—This paper shows fundamentals and applications of the novel TSK-type fuzzy cerebellar model articulation controller (TSK-FCMAC) network. It resembles a neural structure that derived from the Albus CMAC algorithm and Takagi-Sugeno-Kang parametric fuzzy inference systems. A self-constructing learning algorithm consists of the self-clustering method (SCM) and the backpropagation algorithm (BP) uses to tune the adjustable parameters are proposed. The SCM is a fast, one-pass algorithm for a dynamic estimation of the number of hypercube cells in an input data space. The clustering technique does not require prior knowledge of things such as the number of clusters present in a data set. The backpropagation algorithm is used to tune the adjustable parameters. Experimental results show the performance and applicability of the proposed model.**

***Index Terms*—TSK-type fuzzy model, cerebellar model articulation controller (CMAC), self-clustering, backpropagation, prediction.**

I. Introduction

In 1975, the cerebellar model articulation controller (CMAC) developed by Albus [1], is an artificial neural network inspired by the cerebellum. The advantages of the CMAC network are summarized as follows: a simple local neural network, fast learning speed, high convergence rate, and easier hardware implementation, etc. Because of the simple structure and fast learning speed of the CMAC network, it has been successfully applied in many fields, such as robot control, signal processing, and pattern recognition However, there are several limitations for the Albus' CMAC network. Using this technique, the input space is quantized into discrete states as well as larger size overlapped areas called *hypercubes*. In this paper, we propose a new network structure, mainly derived from the CMAC algorithm and Takagi-Sugeno-Kang (TSK) parametric fuzzy inference systems [2], to solve the problem. The so-called TSK-type fuzzy CMAC (TSK-FCMAC) network resembles the original CMAC proposed by Albus, in the sense that it is a local network, i.e., for a given input vector, only a few of the networks nodes (or hypercube cells) will be active and will effectively contribute to the corresponding network output.

Manuscript received March 9, 2014; revised May 30, 2014. This work was supported in part by the National Science Council of the Republic of China, Taiwan under grant NSC 102-2221-E-167-023.

C. L. Lee is with the International Trade Department, National Taichung University of Science and Technology, Taichung City, Taiwan 404, ROC (e-mail: merrylee@nutc.edu.tw).

C. J. Lin is with the Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung City, Taiwan 411, ROC (e-mail: cjlin@ncut.edu.tw).

II. The TSK-Type Fuzzy CMAC (TSK-FCMAC) Network

We propose a new TSK-type fuzzy CMAC (TSK-FCMAC) network (see Fig. 1). The architecture of the TSK-FCMAC network consists of the input space partition, association memory selection, and defuzzification. In the TSK-FCMAC network, we use Gaussian basis function as the receptive field functions and the linear parametric equation of the network input variance as the TSK-type output for learning. Some learned information is stored in the receptive field functions and TSK-type output vectors. A Gaussian basis function for N_D dimensions problem given as follows

$$\alpha_j = \prod_{i=1}^{N_D} e^{-((x_i - m_{ij})/\sigma_{ij})^2} \tag{1}$$

where the α_j represents the j_{th} element of the association vector, x_i represents the input value of the i_{th} dimension for a specific input state x, m_{ij} represents the center of the receptive field functions, σ_{ij} represents the variance of the receptive field functions, and N_D represents the number of the receptive field functions for each input state. Each element of the receptive field functions is inferred to produce a partial fuzzy output by applying the value of its corresponding association vector as input matching degree. The partial fuzzy output is defuzzified into a scalar output y by the centroid of area (COA) approach. Then the actual output y is derived as follows

$$y = \frac{\sum_{j=1}^{N_L} \alpha_j \left(a_{0j} + \sum_{i=1}^{N_D} a_{ij} x_i \right)}{\sum_{j=1}^{N_L} \alpha_j} \tag{2}$$

III. The Learning Algorithm for TSK-FCMAC Network

A learning algorithm, which consists of an input space partition scheme based on the self-clustering method (SCM) to appropriately determine the various distributions of the input training data and a parameter learning scheme based on the gradient descent learning algorithm to adjust the free parameters.

A. The Self-Clustering Method (SCM)

The SCM is proposed to implement scatter partitioning of the input space. It is a distance-based connectionist clustering algorithm. In any hypercube cell, the maximum distance between an example point and the hypercube cell center is less than a threshold value, which has been set as a clustering parameter and would affect the number of hypercube cells to be estimated. In the clustering process, the data examples

DOI: 10.7763/IJIEE.2014.V4.481

come from a data stream, and the process starts with an empty set of hypercube cells.

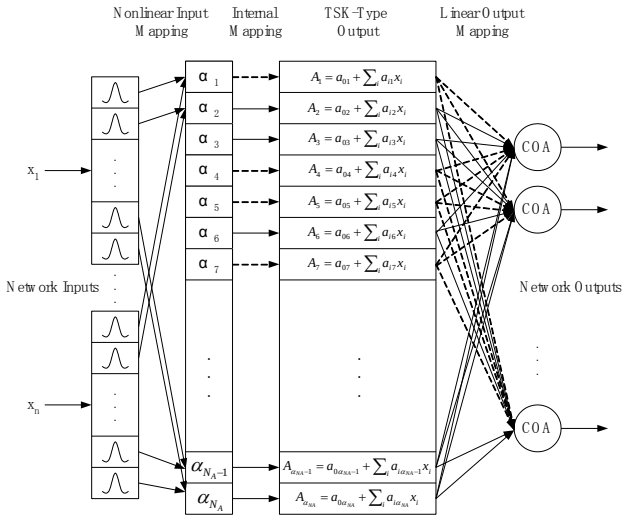


Fig. 1. The architecture of the TSK-FCMAC network.

The hypercube cell will be updated and be changed depends on the position of the current example in the input space. A hypercube cell will not be updated any more when its distance, *Dc*, reaches the value that is equal to the threshold value D_{thr}.

Step 1: Create the first hypercube cell from the input stream as the first hypercube cell center C_1, and setting its distance Dc_1 and width Wd_1_x, Wd_1_y to zero.

Step 2: If all samples of the data stream have been processed, the algorithm is finished. Otherwise, the current input sample, P_i, is taken and the distances between this sample and all *n* already created hypercube cell centers C_j, $Dist_{ij} = \|P_i - C_j\|$, *j*=1, 2, …, *n*, are calculated.

Step 3: If there is any distance value $Dist_{ij}$ equal to, or less than, at least one of the distance Dc_j, *j*=1, 2, …, *n*, it means that the current sample P_i belongs to a hypercube cell C_m with the minimum distance

$$Dist_{im} = \|P_i - C_m\| = \min\left(\|P_i - C_j\|\right),\ j = 1,2,...,n \tag{3}$$

In this case, neither a new hypercube cell is created, nor any existing it is updated, as in the cases of P_4 and P_6 shown in Fig. 3. The algorithm then returns to Step 2. Otherwise, the algorithm goes to the next step.

Step 4: Find a hypercube cell with center C_m and its distance Dc_m from all n existing hypercube cell centers by calculating the values $S_{ij} = Wd_{ij} + Dc_j$, *j*=1, 2, …, *n*, and then choosing the hypercube cell center C_m with the minimum value S_{im}:

$$S_{im} = Wd_{im} + Dc_m = \min(S_{ij}),\ j = 1,2,...,n \tag{4}$$

The special situation shows that the distances between a given point P_{11} and both hypercube cell means Dc_1 and Dc_2 are the same as shown in Fig. 2. The cluster C_2, which has small dimension distances D_2_x, will be selected to expand. To avoid the hypercube cell numbers increase quickly, we make a judgment, as follows:

If (the distance between P_{11} and Dc_1 is equal to the distance between P_{11} and Dc_2) and ($D_1\_x > D_2\_x$) then $D_{im} = Dc_1$

Step 5: If S_{im} is greater than D_{thr}, the example P_i does not belong to any existing hypercube cells. A new hypercube cells is created as described in *Step 1* and the algorithm returns to *Step2*.

Step 6: If S_{im} is not greater than D_{thr}, the hypercube cells C_m is updated by moving its center, C_m, and increasing the value of its distance, Dc_m, and width Wd_m_x, Wd_m_y. The parameters are updated by the following equation:

$$Wd_m\_x^{new} = \left(\left\|C_m\_x - P_i\_x\right\| + Wd_m\_x\right)/2 \tag{5}$$

$$Wd_m\_y^{new} = \left(\left\|C_m\_y - P_i\_y\right\| + Wd_m\_y\right)/2 \tag{6}$$

$$C_m\_x^{new} = P_i\_x - D_m\_x^{new} \tag{7}$$

$$C_m\_y^{new} = P_i\_y - D_m\_y^{new} \tag{8}$$

$$Dc_m^{new} = S_{im}/2 \tag{9}$$

where C_m_x is the value of the *x* dimension for C_m, C_m_y is the value of the *y* dimension for C_m, P_i_x is the value of the *x* dimension for P_i, and P_i_y is the value of the *y* dimension for P_i .The algorithm returns to *Step 2*.

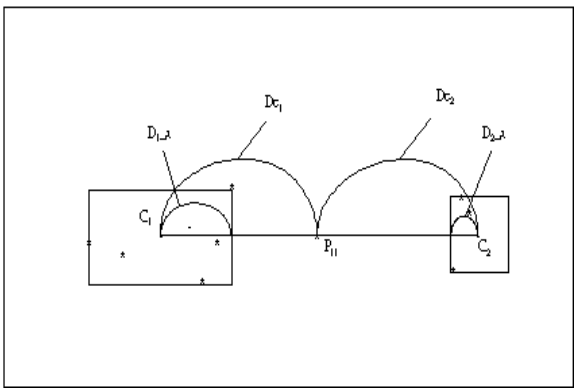


Fig. 2. The special case of SCM (where Pi: pattern, Cj: cluster center, Dcj: hypercube cell distance, Wdj_x: x-dimensions hypercube cell width, Wdj_y: y-dimensions hypercube cell width).

B. The Parameter Learning Scheme

There are four parameters need to be tuned, i.e. m_{ij}, σ_{ij}, a_{0j}, and a_{ij}, and used the supervised gradient descent method to modify these parameters. Our goal is to minimize the cost function *E*, defined as follows

$$E(t) = \frac{1}{2}[y^d(t) - y(t)]^2 \tag{10}$$

where $y^d(t)$ is the desired output and $y(t)$ is the actual output at time *t*. Then the parameter learning algorithm, based on backpropagation, is described as follows

$$a_{0j}(t+1) = a_{0j}(t) + \Delta a_{0j} = a_{0j}(t) + \eta \cdot e \cdot \left(\frac{\partial y}{\partial a_{0j}}\right) \tag{11}$$

$$a_{ij}(t+1) = a_{ij}(t) + \Delta a_{ij} = a_{ij}(t) + \eta \cdot e \cdot \left(\frac{\partial y}{\partial a_{ij}}\right) \tag{12}$$

where a_{0j} and a_{ij} is the proper scalar and coefficient of the i_{th} input dimension, and *j* is the j_{th} element of the TSK-type

output vector for j=1, 2, …, N_L (N_L denote the number of hypercube cells) , η is the learning rate between 0 and 1. The elements of the TSK-type output vectors are updated by the amount

$$\Delta a_{0j} = \eta \cdot e \cdot \frac{\partial y}{\partial a_{0j}} = \eta \cdot e \cdot \frac{\alpha_j}{\sum_{j=1}^{N_L} \alpha_j} \tag{13}$$

$$\Delta a_{ij} = \eta \cdot e \cdot \frac{\partial y}{\partial a_{ij}} = \eta \cdot e \cdot \frac{x_i \alpha_j}{\sum_{j=1}^{N_L} \alpha_j} \tag{14}$$

where η is the learning rate, between 0 and 1, and e is the error between the desired output and the actual output, $e = y^d - y$. The receptive field functions are updated as follows

$$m_{ij}(t+1) = m_{ij}(t) + \Delta m_{ij} = m_{ij}(t) + \eta \cdot e \cdot \left(\frac{\partial y}{\partial \alpha_j} \cdot \frac{\partial \alpha_j}{\partial m_{ij}} \right) \tag{15}$$

$$\sigma_{ij}(t+1) = \sigma_{ij}(t) + \Delta \sigma_{ij} = \sigma_{ij}(t) + \eta \cdot e \cdot \left(\frac{\partial y}{\partial \alpha_j} \cdot \frac{\partial \alpha_j}{\partial \sigma_{ij}} \right) \tag{16}$$

where i denotes the i_{th} input dimension for i=1, 2, …, n, m_{ij} denotes the mean of the receptive field functions, and σ_{ij} denotes the variance of the receptive field functions. The parameters of the receptive field functions are updated by the amount

$$\begin{aligned} \Delta m_{ij} &= \eta \cdot e \cdot \frac{\partial y}{\partial \alpha_j} \cdot \frac{\partial \alpha_j}{\partial m_{ij}} \\ &= \eta \cdot e \cdot \frac{\left(a_{0j} + \sum_{i=1}^{N_D} a_{ij} x_i\right) \sum_{j=1}^{N_L} \alpha_j - \sum_{j=1}^{N_L} \alpha_j \left(a_{0j} + \sum_{i=1}^{N_D} a_{ij} x_i\right)}{\left(\sum_{j=1}^{N_L} \alpha_j\right)^2} \cdot \alpha_j \cdot \frac{2\left(x_i - m_{ij}\right)}{\sigma_{ij}^{\ 2}} \end{aligned} \tag{17}$$

$$\begin{aligned} \Delta \sigma_{ij} &= \eta \cdot e \cdot \frac{\partial y}{\partial \alpha_j} \cdot \frac{\partial \alpha_j}{\partial \sigma_{ij}} \\ &= \eta \cdot e \cdot \frac{\left(a_{0j} + \sum_{i=1}^{N_D} a_{ij} x_i\right) \sum_{j=1}^{N_L} \alpha_j - \sum_{j=1}^{N_L} \alpha_j \left(a_{0j} + \sum_{i=1}^{N_D} a_{ij} x_i\right)}{\left(\sum_{j=1}^{N_L} \alpha_j\right)^2} \cdot \alpha_j \cdot \frac{2\left(x_i - m_{ij}\right)^2}{\sigma_{ij}^{\ 3}} \end{aligned} \tag{18}$$

where η is the learning rate of the mean and the variance for the receptive field functions.

IV. Prediction of the Chaotic Time Series

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

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{19}$$

Crowder [3] extracted 1000 input-output data pairs $\{x, y^d\}$ which consist of four past values of $x(t)$, i.e.

$$[x(t-18), x(t-12), x(t-6), x(t); x(t+6)] \tag{20}$$

where τ=17 and $x(0)$=1.2. There are four inputs to the TSK-FCMAC network, corresponding to these values of $x(t)$, and one output representing the value $x(t+\Delta t)$, where Δt is a time prediction into the future.

The first 500 pairs, from $x(1)$ to $x(500)$, are the training data set, while the remaining 500 pairs, from $x(501)$ to $x(1000)$, are the testing data set. The initial threshold value in the SCM is 0.7, and the learning rate is η=0.01. After the SCM clustering process, there are four hypercube cells generated.

There are two parameter learning scheme are used during the training process. First scheme (i.e., scheme 1) represents that only the TSK-type consequent parameters are tuned by gradient descent method while the receptive field functions are fixed. Second scheme (i.e., scheme 2) represents that both the TSK-type consequent parameters and the receptive field functions are tuned by gradient descent method. The learning curves of the scheme 1 and scheme 2 methods are shown in Fig. 3. Table I shows the comparison results of the prediction performance among various predictors.

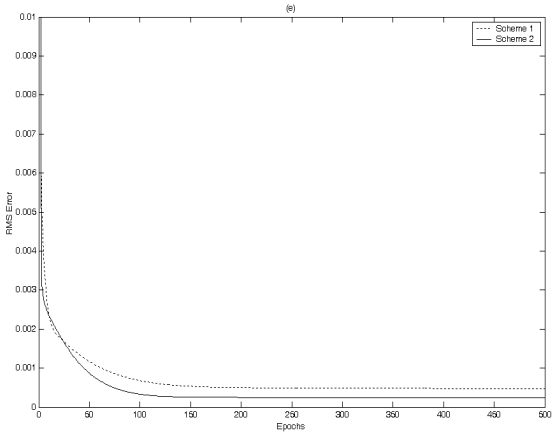
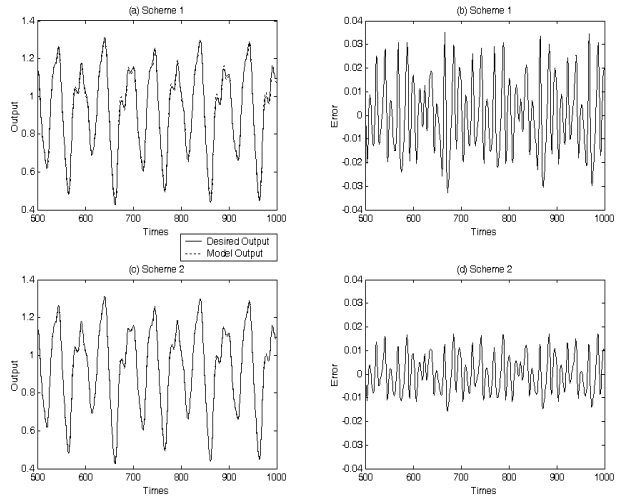


Fig. 3. Learning curves of the scheme 1 and scheme 2 parameter learning methods.

The prediction outputs of the chaotic time series for the scheme 1 and scheme 2 methods are shown in Fig. 4 (a) and Fig. 4 (c). The solid line represents the output of the time series, and the dotted line represents the output of the TSK-FCMAC network. Fig. 4 (b) and Fig. 4 (d) show the prediction errors and the TSK-FCMAC network output using the scheme 1 and scheme 2 methods. Table I shows the comparison results of the prediction performance among various predictors. The previous results were taken from [4]-[6]. The performance of the very compact fuzzy system obtained by the TSK-FCMAC network is better than all previous works. The performance of the very compact fuzzy system obtained by the TSK-FCMAC network is better than all previous works.

TABLE I: Comparison Results of Various Existing Models on Chaotic Time Series Prediction

Methods	RMSE
TSK-FCMAC	0.0094 (Scheme 1)
	0.0048 (Scheme 2)
GEFREX [4]	0.0061
ANFIS [5]	0.007
Kim & Kim [6]	0.026
6th-order Polynomial	0.04
Cascade Correlation NN	0.06
Min operator	0.09
Wang (product operator)	0.091



Figs. 4. Results from $x(501)$-$x(1000)$ of (a) The time series for the scheme 1 method. (b) Prediction errors between the desired output and the TSK-FCMAC network output for scheme 1 method. (c) The time series for scheme 2 method. (d) Prediction errors between the desired output and the TSK-FCMAC network output for scheme 2 method.

V. Conclusion

In this paper, a new TSK-type fuzzy CMAC (TSK-FCMAC) network was proposed for prediction. The advantages of the proposed TSK-FCMAC network are summarized as follows: (1) it implements scatter partitioning of the input space dynamically; (2) it can keep a smaller rms error; and (3) it has much lower memory requirement than conventional CMAC network.

Acknowledgment

This work was supported by National Science Council, R.O.C., under grant NSC 102-2221-E-167 -023.

References

[1] J. S. Albus, "A new approach to manipulator control: The cerebellar model articulation controller (CMAC)," *Trans. ASME J. Dyn. Syst., Meas., Contr.*, vol. 97, pp. 220-227, 1975.

[2] T. Takagi and M. Segeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, Jan./Feb. 1985, pp. 116-132.

[3] R. Saey, "Cowder III: predicting the mackey-glass time series with cascade- correlation learning," in D. Touretzky, G. Hinton, and T. Sejnowski ed., pp. 117-123, 1990.

[4] M. Russo, "Genetic fuzzy learning," *IEEE Trans. on Evol. Comput.*, vol. 4, no. 3, pp. 259–273, 2000.

[5] J. S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. on Syst., Man, and Cybern.*, vol. 23, pp. 665-685, 1993.

[6] D. Kim and C. Kim, "Forecasting time series with genetic fuzzy predictor ensemble," *IEEE Trans. Fuzzy Syst.*, vol. 5, no. 4, pp. 523–535, 1997.

Chin-Ling Lee received the B. A. degree in English from Tamkang University, Taiwan, R. O. C., in 1986, the M. A. degree in English from University of Central Missourri, U. S. A., in 1990, and the Ph. D. degree in industrial education from National Taiwan Normal University, Taiwan, R. O. C. Currently, she is an associate professor of International Trade Department, National Taichung University of Science and Technology, Taichung City, Taiwan, R. O. C. Her current research interests are e-learning education, English for specific purposes, and English teaching and learning.

Cheng-Jian Lin received the Ph.D. degrees in electrical and control engineering from the National Chiao-Tung University, Taiwan, R.O.C., in 1996. Currently, he is a distinguished professor of Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung County, Taiwan, R.O.C. His current research interests are soft computing, pattern recognition, intelligent control, image processing, bioinformatics, and Android/iPhone program design.

Dr. Lin is an editorial board of Applied Computational Intelligence and Soft Computing from 2011, an editorial Board of computational Intelligence and Neuroscience from 2011, an editorial board of Int'l journal of Control Engineering and Technology from 2011, and an editorial board member of the Open Cybernetics and Systemics Journal from 2010. Dr. Lin is a member of the Phi Tau Phi, the Chinese Fuzzy Systems Association (CFSA), the Chinese Automation Association, the Taiwanese Association for Artificial Intelligence (TAAI), the IEEE Systems, Man, and Cybernetics Society, and the IEEE Computational Intelligence Society. He is an executive committee member of the Taiwanese Association for Artificial Intelligence (TAAI) from 2003 to 2010. He is an executive committee member of the Chinese Fuzzy Systems Association (CFSA) from 2007.

Dr. Lin has received several honors and awards, including the 2006 Outstanding Paper Award of the 11th Conference on Artificial Intelligence and Applications, the 2007 Outstanding Paper Award of the 12th Conference on Artificial Intelligence and Applications, the 2006 Best Paper Award of International Trans. on Computer Science and Engineering (Vol. 32, No. 1), the First Place of the 14th College Contest of IT Services Innovation 2009, the Silver Prize of International Trade Fair Ideas-Invention-New Productions Nuremberg 2009 (iENA 2009), and the Silver Prize of Seoul International Invention Fair 2010 (SIIF 2010).