ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 6, June 2017

Non Linear Dynamic System Identification Using MRAN Algorithm

[1] Parimala Gandhi G, [2] Saritha C,, [3] Sangeetha C
[1][2][3] Department of Electronics and Communication Engineering, R R Institute of Technology, Chikkabanavara, Bengaluru-560090

***Abstract*— This paper presents a performance analysis of Minimal Resource Allocating Network (MRAN) algorithm for identification of nonlinearity in dynamic systems for nonlinear time invariant and time varying problems.Next, methods for improving the run time performance of MRAN for real time identificationof the nonlinear systems is also used. Extended Minimum Resource Allocating Network (EMRAN) which utilizes a winner neuron strategy is and its performance is also been discussed. The modification in MRAN reduces the computation to considerable reduction in the learning time with only a slight increase in the approximationerror. Using the same problem the results of EMRAN is also analyzed which shows that the later is well suited forfast on-line identification of nonlinear plants such as any type of non-linearity arises in aircraft control structure.**

***Index Terms*— ANN, RBF, MRAN, EMRAN**

INTRODUCTION

Neural networks can be used as nonlinear dynamic system controllers to tackle problems where conventional approaches fail and proved to be ineffective [1]. Areas like flight control [2][3] the online control learning scheme is sparse and requires a large computational load when neural network is used in practical use for online control scheme. Hence, the problem of designing fast on-line learning algorithm for practical implementation of neural control schemes remains an active research topic. For any practical application of a newly developed identification algorithms, it is important to study the real-time implementation of the algorithm and the same study is undertaken for the MRAN algorithm. Based on the analysis, an extension to MRAN called Extended MRAN (EMRAN) is proposed. The focus is to reduce the computation load of the MRAN and to realize a scheme for fast on-line identification.

I. NEURAL NETWORKS

The human brain is a remarkable parallel computer which is source of natural intelligence. The incomplete information is processed by the brain at a rapid rate. Nerve cells in the human body functions about times slower than electronic circuit gates, whereas human brains process auditory and visual information much faster than modern computers. Many researchers, Inspired by biological nervous systems, especially brain modelers, have been exploring artificial neural networks, a non-algorithmic approach to information processing.

A neural network[1], [2] is massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. The procedure used to perform learning process, known as Learning algorithm whose function is used to modify the weights to attain a desired and designed objective in an orderly fashion.

II. RADIAL BASIS FUNCTION

Radial basis function is a type of neural network. The basic form of radial basis function involves three layer. They are input layer, hidden layer and outpit layer. The input layer is the first layer. It connects the network to its environment. The second layer is the hidden layer. It applies the non-linear transformation from input layer to the hidden layer. The final layer is output layer.The hidden layer of an RBF network is nonlinear , whereas the output layer is linear.The activation function of each hidden unit in an rbf network computes the Euclidean distance between the input vector and center of the unit. In most applications the hidden space is of high dimensionality. The output layer is linear, supplying the response of the network to the activation pattern applied to the input layer.

Figure 1 shows the structure of the RBFNN. The RBFNN is three layered feed-forward neural network. The first layer is linear and only distributes the input signal, while the next layer is nonlinear and uses Gaussian functions. The third layer linearly combines the Gaussian outputs. Only the tap weights between the hidden layer and the output layer are modified during training.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 6, June 2017

RBFNN have 5 parameters for optimization:

- The weights between the hidden layer and the output layer.
- The activation function.
- The center of activation functions.
- The distribution of center of activation functions.
- The number of hidden neurons.

The weights between the hidden layer and the output layer are calculated by using Moore-Penrose generalized pseudo-inverse. This algorithm overcomes many issues in traditional gradient algorithms such as stopping criterion, learning rate, number of epochs and local minima. Due to its shorter training time and generalization ability, it is suitable for real-time applications.

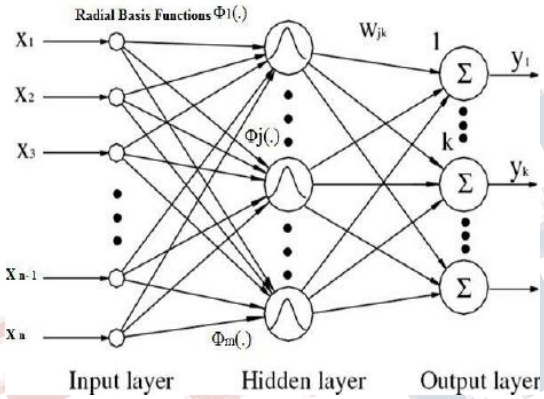

Fig. 1.

III. MRAN

The MRAN algorithm proposed by Lu Yingwei et al. combines the growth criteria of RAN with a pruning strategy to realize a minimal network structure. This algorithm is an improvement to the RAN of Platt and the RANEKF algorithm of Kadirkamanathan.

In this section, we present the MRAN algorithm for training RBF networks. Before explaining the algorithm in detail, the RBF network is briefly described.

Figure 1 shows a typical RBF network, with nx inputs x (x1, · · · ,xnx), and ny outputs ˆy(yˆ1 · · · yˆn_y). The hidden layer consists of N computing units (Φ1 to ΦN), they are connected to the output by N weight vectors $a_1 to a_N$). The outputs of the network which approximate the true output y are,

$$y'=f(x)=a_0 \sum_{n=1}^{N} a_n \phi_n(x) \qquad (1)$$

whereΦn(x) is the response of the nth hidden neuron to the input x, and an is the weightconnecting the nth hidden unit to the output unit. is the bias term and is a Gaussian function given by,

$$\phi_n(x) = \exp\left(-\frac{||x-\mu_n||^2}{\sigma_n^2}\right) \qquad (2)$$

Here, $\mu_n$is the center for the nth hidden neuron and σnis the width of the Gaussian function. ‖denotes the Euclidean norm.

In the MRAN algorithm, the RBF network begins with no hidden units, that is n = 0. As each input-output training data (x_i, y_i) (i is time index) is received, the network is built up based on certain growth criteria. The following steps describe the basic ideas of the MRAN algorithm.

Step 1.Calculate the three error defined

The first step of the algorithm is to check whether the criteria for recruiting a new hidden unit are met,

$$\|e_i = ||y_i - y'_i(x_i)|| > E1 \qquad (3)$$

$$e_{rmsi} = \sqrt{\sum_{j=i-(M-1)}^{i} \frac{e_j^2}{M}} > E2 \qquad (4)$$

$$d_i = ||x_i - \mu_{ir}|| > E3 \qquad (5)$$

where$\mu_{ir}$is the center of the hidden unit which is closest to current input x_i. E1, E2 andE3 are thresholds to be selected appropriately. Equation (3) decides if the existing nodes

are insufficient to obtain a network output that meets the error specification. Eqn. (4) checks whether the network met the required sum squared error specification for thepast M outputs of the network. Eqn (5) ensures that the new node to be added insufficiently far from all the existing nodes.

Only when all these criteria are met, a new hidden node is recruited. Then go to step 2to add a new RBF hidden unit, otherwise go to step 3 to update all the parameters of the network using EKF.

Step 2: Inclusion of a new RBF hidden unit

When all the criteria in Step 1 are satisfied, a new hidden unit is recruited. Each new hidden unit added to the network will have the following parameters associated,

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 6, June 2017

$a_{N+1} = e_i,\ \mu_{N+1} = x_i, \sigma_{N+1} = k||x_i - \mu_{ir}||$ (6)

Those parameters are set to remove the error caused. The overlap of the responses of the hidden units in the input space is determined by κ, the overlap factor. After adding the new hidden neuron, go to step 5 to perform a pruning strategy.

Step 3: Calculating the gradient matrix Bi

If the three criteria for adding new hidden unit cannot be satisfied, then an adaptation of the network parameters should be done. Bi = Δwf (xi) is the gradient matrix of thefunction f (x_i) with respect to the parameter vector w evaluated at wi−1, which will beused in the next step.

Bi = [I, Φ1(x_i) I, Φ1(x_i) ($2a_1/\sigma_1^2$) $(x_i - \mu_1)^T$, Φ1(x_i))($2a_1/\sigma_1^3$)$||x_i)\ - \mu 1||^2$, ... ,ΦN(x_i) I, ΦN(x_i)($2a_N$/σ2N)$(x_i - \mu_N)^T(x_i - \mu_1)$T, ΦN($x_i$)($2a_N$/σ3N)$||x_i - \mu 1||^2$]T (7)

After this preparation, the vector w can be updated, therefore go to Step 4.

Step 4: Updating the parameters using EKF

In this step, the network parameters $w=[a_0^T, a_1^T, \mu_1^T, \sigma_1, \ldots . a_N^T, \mu_N^T, \sigma_N^T]^T$ are adapted using the EKF as follows,

$$w_i = w_{i-1} + k_i e_i \ , \quad (8)$$

wherek_i is the Kalman gain matrix given by,

$$k_i = P_{i-1}B_i[\ R_i + B_i^T P_{i-1} B_i]^{-1} \quad (9)$$

Ri is the variance of the measurement noise. Pi is the error covariance matrix which is updated by,

$$P_{i-1}=[\text{I-}k_i B_i^T]\ P_{i-1}+I_{z*z}\text{q}. \quad (10)$$

Here, q is a scalar that determines the allowed random step in the direction of the gradient matrix. If the number of parameters to be adjusted is z, then Pi is a z × z positive definite symmetric matrix. When a new hidden neuron is allocated, the dimensionality of Pi increases to,

$$p_i = \begin{pmatrix} p_{i-1} & 0 \\ 0 & p_0 I_{z1*z1} \end{pmatrix} \quad (11)$$

Step 5: Pruning strateg

The last step of the algorithm is to prune those hidden neurons that contribute little to the network's output for Nwconsecutive observations. Let matrix O denotes the outputs of the hidden layer and A denotes the weight matrix A = (a1, · · · ,aN), consider the output

O_{nj} (j = 1· · · ny) of the nth hidden neuron,

$$O_{nj} = A_{nj}\exp(-\frac{||x-\mu_n||^2}{\sigma_n^2})),\ \text{n=1……N,j=1…….}n_y \quad (12)$$

If A_{nj}or $\sigma_n$in the above equation is small,O_{nj}might become small. Also, if ||x – μn|| is large, the output will be small. This would mean that the input is sufficiently far away fromthe center of this hidden neuron. To reduce inconsistency caused by using the absolutevalue of the output, this value is normalized to that of the highest output,

$$r_{nj} = \frac{O_{nj}}{\max\{O_{1j}, O_{2j}, O_{Nj}, \ldots\ldots, O_{nj}\}},\ \text{n=1……N,j=1…….}n_y \quad (13)$$

The normalized output of each neuronr_{nj} is then observed for Nw consecutive inputs. Aneuron is pruned, if its output rnj(j = 1· · · ny) falls below a threshold value(δ) for the New consecutive inputs. Then the dimensionality of all the related matrixes will be adjusted to suit the reduced network.

The nonlinear SISO time-invariant system (BM-1) to be identified is described by the following first order difference equation,

$$\text{y(i)} = \frac{29}{40}\sin\left(\frac{16u(i-1)+8y(i-1)}{3+4u(i-1)^2+4y(i-1)^2}\right) + \frac{2}{10}u(i-1) + \frac{2}{10}(i-1) \quad (14)$$

A random signal uniformly distributed in the interval [-1,1] is used for u(i) in the system. For comparison purposes, the same error criteria defined in [16] vizId(i) is used,

$$I_d\text{(i)} = \frac{1}{n_y * N_w}\sum_{p=0}^{N_w-1}\sum_{j=1}^{n_y}[y_j(i-p) - y'_j(i-p)] \quad (15)$$

In this case, the MRAN's 3 criteria gates are selected as: E1 = 0.01, E2 = 0.1,E3= max{εmax× γi, εmin}, εmax= 1.2, εmin= 0.6, γ = 0.997, we use δ =

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 4, Issue 6, June 2017

0.0001 as the pruning threshold, and thesize of the two sliding windows (M,Nw) as 48.

IV. RESULTS AND DISCUSSIONS

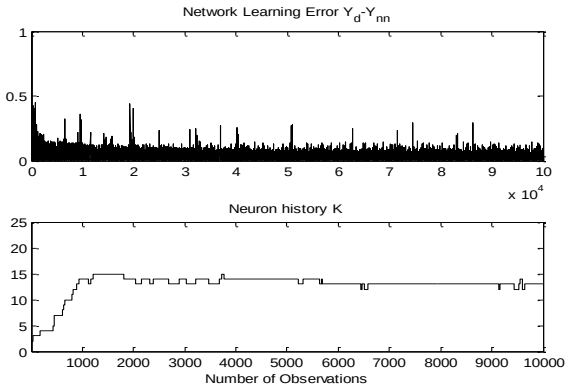

Fig. 2.

The identification results using MRAN and are given in figures 2and 3. Fig. 2 presents the hidden neuron evolution history along with the time histories of the three error functions.
From Fig.2 one can see clearly how the hidden neurons are added and pruned according to the three criteria. In this case, MRAN takes 8 hidden units to identify the system,

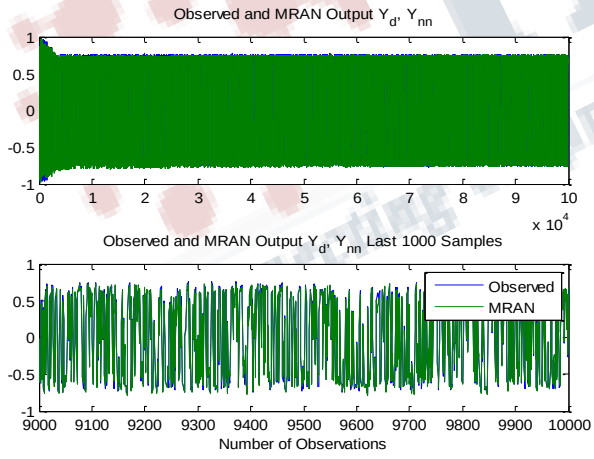

Fig. 3.

Fig. 3 shows the time history of the error index Id. It can bee seen from Fig. 3 that MRAN's error is coming down drastically and eventually it is low.

CONCLUSION

The results indicate that MRAN realizes networks using less hidden neurons with better approximation accuracy. EMRAN utilizes a winner neuron strategy . This modification reduces the computation load for MRAN and leads to considerable reduction in the identification time with only a minimal increase in the approximation error. This also indicates the minimum sampling time one can select using EMRAN for identification problems. EMRAN can 'adaptively track' the dynamics of the nonlinear system quickly without loss of accuracy and is ideal for fast on-line identification of nonlinear plants. The same algorithm can be used to solve the non-linearity arises in aircraft dynamic system.

REFERENCEE

[1] Agarwal, M., "A systematic classification of neural-network-based control", Journal of guidance, control, and dynamics, pp. 75-93, Apr.1997

[2] Sadhukhan, D. and Feteih,S., "F8 neurocontroller based on dynamic inversion", IEEE control systems magazines, 19, No.1, pp. 150-156, 1996

[3] Narendra, K. and Parthasarathy, K., "Identification and control of dynamic systems using neural networks", IEEE transactions on neural networks, Vol.1, No.1, pp. 4-26, Mar.1990

[4] Analysis of Minimal Radial Basis Function Network Algorithm for Real-Time Identification of Nonlinear Dynamic Systems by Li Yan, N.Sundararajan and P.Saratchandran

[5] Simon J Haykin, Neural Networks,a comprehensive foundation, second editionpearson education prentice hall

[6] Bharath Y.K., Veena S, Nagalakshmi K.V., ManjunathDarshan, RohiniNagapadma, "Development of Robust VAD Schemes for Voice Operated Switch Application in Aircrafts",IEEE-ICATccT, pp189-193,July 2016.

[7] NagarajRamrao, T.V. Ramamurthy ,"Fault Tolerant Flight Controller With Neural Network Augmentation For A High Performance Fighter Aircraft During Auto-Landing "ACSE Journal, Volume (6), Issue (4), Dec., 2006