Extreme Learning ANFIS for Control Applications

G. N. Pillai, Jagtap Pushpak, M. Germin Nisha
Department of Electrical Engineering
Indian Institute of Technology Roorkee
Roorkee, India

***Abstract*— This paper proposes a new neuro -fuzzy learning machine called extreme learning adaptive neuro-fuzzy inference system (ELANFIS) which can be applied to control of nonlinear systems. The new learning machine combines the learning capabilities of neural networks and the explicit knowledge of the fuzzy systems as in the case of conventional adaptive neuro-fuzzy inference system (ANFIS). The parameters of the fuzzy layer of ELANFIS are not tuned to achieve faster learning speed without sacrificing the generalization capability. The proposed learning machine is used for inverse control and model predictive control of nonlinear systems. Simulation results show improved performance with very less computation time which is much essential for real time control.**

Keywords—extreme learning machines; fuzzy neural systems; nonlinear model predictive control; inverse control

I. INTRODUCTION

Extreme learning machine (ELM) proposed by Huang [1] for single-hidden layer feed-forward neural networks (SLFNs) is a fast and accurate method for solving classification and regression problems. ELM reduces the computational burden, without reducing the generalization capability, by adjusting only the output layer parameters. Unlike the conventional techniques, hidden layer parameters of ELM are randomly chosen. Due to the fast implementation and generalization capability, ELM has been widely used in numerous classification and regression problems [2-3]. Since the introduction of classic ELM, various variants of ELM were proposed in literature [3].

In spite of the advantages, ELM suffers from the inherent randomness of the results. The randomness of hidden layer parameters causes an additional uncertainty problem, both in approximation and learning. Inappropriate activation function reduces the generalization and approximation capability of ELM and the choice of activation functions in ELM learning algorithm is problem dependent [4]. If average performance of ELM is to be assessed, any runs must be repeated several times which increases the computational cost [5].

Because of the absence of a capability to explain how the trained network arrives at a particular decision, neural network is considered as a black box. They don't have the capability of qualitative knowledge representation. Unlike neural networks, fuzzy logic systems are used to represent imprecise and qualitative knowledge. However, automated learning capability is absent in fuzzy logic systems. Neuro-fuzzy systems judiciously combine the merits of neural network and fuzzy logic to build more intelligent systems [6]. Adaptive network based fuzzy inference system (ANFIS) by Jang [7], is a hybrid intelligent system which integrates the fuzzy logic's qualitative approach and neural network's adaptive capabilities towards better performance. ANFIS is a universal approximator and hence applied to many function approximation problems in engineering applications. However, the hybrid learning algorithm used in ANFIS is computationally very expensive .

This paper proposes a simple learning algorithm called extreme learning ANFIS (ELANFIS) which can overcome the drawbacks of ELM and ANFIS networks. The preliminary idea of the proposed algorithm is given in [8]. The performances of ANFIS and ELANFIS in solving regression problems with small number of input features are compared in [9]. ELANFIS reduces the computational cost of conventional ANFIS by adopting the ELM strategy. Randomness of ELMs is avoided by adopting a Sugeno type fuzzy system. This paper shows that curse of dimensionality problem is less severe in the case of ELANFIS in solving problems where input dimensions are high. Because of the higher accuracy of ELANFIS models, this approach is applied to control of nonlinear systems. The proposed learning machine is successfully applied to model predictive control (MPC) and inverse control of nonlinear systems.

The remaining sections of the paper are organized as follows. Section 2 introduces the ELANFIS algorithm. Performance evaluation of the proposed algorithm on a regression problem is also done in section 2. In section 3, ELANFIS based inverse model control technique is used to control the level of conical tank which is highly nonlinear dynamic system. In section 4, the proposed method is applied for model predictive control of catalytic continuous stirred tank reactor (CSTR) process. Conclusions are given in section 5.

II. PROPOSED ELANFIS ALGORITHM

A. *Extreme Learning Machines (ELM)*

Extreme learning machines are single hidden layer feedforward networks (SLFN) with random parameters for hidden nodes [1]. Consider a training dataset with N samples (x_i, t_i) where $x_i = [x_{i1}, x_{i2}, x_{i3} ..., x_{in}]^T$ and $t_i = [t_{i1}, t_{i2}, ..., t_{im}]^T$. To solve this classification problem, consider a conventional SLFN with $\widetilde{N}$ hidden nodes and activation function g(x) . The output nodes are assumed to be linear. The output of the linear layer o_j can be obtained as

$$\sum_{i=1}^{\tilde{N}} \beta_i g_i(x_j) = \sum_{i=1}^{\tilde{N}} \beta_i g(w_i x_j + b_i) = o_j \;\; for \;\; j = 1,....N$$

where $w_i = [w_{i1}, w_{i2}, ..., w_{in}]^T$ between the input nodes and the j^th hidden node, $\beta_i = [\beta_{i1}, \beta_{i1}, ...\beta_{im}]$ is the weight vector for the linear output nodes and bi is the threshold of the *i*th hidden node. This network can approximate the given problem with *N* samples with zero error. That means there exist parameters β_i, w_i and b_i such that

$\sum_{i=1}^{\tilde{N}} \beta_i g(w_i x_j + b_i) = t_j$ for $j = 1,.......N$, where t is the target.

The above N equations may be written as

$$H\beta = T \tag{1}$$

where

$$H = \begin{bmatrix} g(w_1.x_1 + b_1) & \cdots & g(w_{\tilde{N}}.x_1 + b_{\tilde{N}}) \\ \vdots & \ddots & \vdots \\ g(w_1.x_N + b_1) & \vdots & g(w_{\tilde{N}}.x_N + b_{\tilde{N}}) \end{bmatrix}_{N \times \tilde{N}}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times m} \quad \text{and} \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m}$$

Given a training set, the number of hidden nodes and hidden node activation functions, the algorithm for extreme learning machine is given as follows,

Step 1: The weights between hidden nodes and input nodes w_i and the threshold of the hidden nodes b_i are randomly assigned.

Step 2: Calculate the hidden layer output matrix H.

Step 3: Calculate the output linear layer weights using

$$\beta = H^{-1}T \tag{2}$$

where H^{-1} is the Moore–Penrose generalized inverse of matrix H. This gives the smallest norm least-squares solution of the above linear system and this solution is unique.

B. Extreme Learning ANFIS (ELANFIS)

The structure of extreme learning ANFIS is same as that of conventional ANFIS. The architecture which uses Sugeno type rules is shown in Fig. 1. It is assumed that two rules are used for the knowledge representation. A two rule Sugeno ANFIS network has rules of the form:

If x *is* A_1 *and* y *is* B_1 *THEN* $f_1 = p_1 x + q_1 y + r_1$

If x *is* A_2 *and* y *is* B_2 *THEN* $f_2 = p_2 x + q_2 y + r_2$

where *x* and *y* are crisp inputs, *A_i*, *B_i* are linguistic variables.

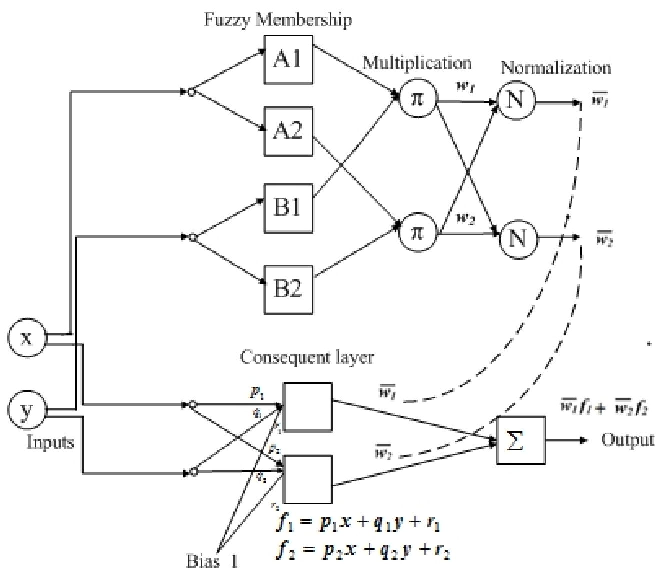

Fig. 1. Architecture of ELANFIS

The top part of Fig. 1 has three layers and represents the premise part of the fuzzy rules and the bottom part has two layers and represents the consequent part. In the top part, the nodes in the first layer represent the fuzzy membership functions. The output of each node is:

$$\mu_{A_i}(x) \qquad for\; i = 1,2$$

$$\mu_{B_{i-2}}(y) \qquad for\; i = 3,4$$

where $\mu(x)$ is the membership grade for input x and $\mu(y)$ is the membership grade for input y. Fixed nodes are used in the second layer. The product t-norm is used to '*and*' the membership grades of premise parameters, as its gives a smoothing effect. Firing strengths w_i of the fuzzy rules are calculated as

$$w_i = \mu_{A_i}(x)\mu_{B_i}(y), \quad i = 1,2 \tag{3}$$

Normalized firing strengths $\overline{w_i}$ are calculated in the third layer which also consists of fixed nodes.

$$\overline{w_i} = \frac{w_i}{w_1 + w_2} \tag{4}$$

The first layer in the bottom part of Fig. 1 represents a linear neural network with p_i, q_i and r_i as weight parameters. These weight parameters are are adaptive and are learned using least square estimation method. Assuming normalized firing strengths of fuzzy rules and weight parameters are known, the output of this is given as

$$\overline{w_i} f_i = \overline{w_i}(p_i x + q_i y + r_i) \tag{5}$$

The second layer in the bottom part computes the overall output as:

$$t = \sum_i \overline{w_i} f_i = \overline{w_1}(p_1 x + q_1 y + r_1) + \overline{w_2}(p_2 x + q_2 y + r_2) \tag{6}$$

Bell shaped membership functions are used in the premise part because smoothness in change in membership grade and flexibility in core of membership function. The mathematical representation of bell shape membership function is given as,

$$\mu_A(x) = \frac{1}{1+\left|\frac{x-c_i}{a_i}\right|^{2b_i}} \tag{7}$$

where the premise parameters a_j, b_j, and c_j are position and shape deciding parameters.

If the sample size of the two dimensional (x and y) data for Fig. 1 is N, the targets $t_1, t_2 ... t_N$ can be found out using (6). The N linear equations can be expressed in matrix form as

$$\begin{bmatrix} t_1 \\ t_2 \\ . \\ . \\ . \\ t_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} \overline{w}_1 x & \overline{w}_1 y & \overline{w}_1 & \overline{w}_2 x & \overline{w}_2 y & \overline{w}_2 \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \end{bmatrix}_{N\times 6} \begin{bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \end{bmatrix}_{6\times 1}$$

The above matrix equation completely describes the two-rule ENANFIS structure shown in Fig. 1. The premise parameters a_j, b_j and c_j and the consequent parameters p_i , q_i and r_i are to be evaluated using the ELANFIS algorithm which is explained in the next sub section.

In the general case, if there are N training data pairs, the N linear equations can be represented in matrix form as

$$T_{N\times 1} = H_{N\times m^n (n+1)} \beta_{m^n (n+1)\times 1} \tag{8}$$

where m is the number of membership functions, n is the dimension of input data and m^n is the maximum number of rules used.

C. ELANFIS Algorithm

An In the conventional ANFIS, the premise parameters are determined using gradient descent methods like back propagation algorithm. Consequent parameters learned using linear network's training methods like least mean squares estimation [7]. In this type of hybrid learning algorithm (HLA), in the forward pass, the input patterns are applied assuming fixed premise parameters and the optimal consequent parameters are calculated by an iterative least mean square procedure. In the second pass called as backward pass the patterns are propagated again, and in this epoch, back propagation is used to change the premise parameters to reduce the training error, while the consequent parameters remain fixed. This procedure is continued till training error is minimized.

In ELANFIS, the ELM strategy is applied to tune the premise parameters of fuzzy rules. These parameters (a_j, b_j, and c_j) are randomly selected with some constraints on the ranges of these parameters. Compared to ELM, the randomness is reduced in ELANFIS due to the explicit knowledge embedded in the linguistic variables in the premise part of the fuzzy rules. Once the premise parameters for all the inputs are selected, the H matrix in (8) can be easily determined. Then the unknown linear network parameters can be determined as

$$\beta = H^{-1} T \tag{9}$$

Consider a training data, with n dimensional inputs

$$[X_1 \quad X_2 \quad . \quad . \quad X_n \,; T]$$

For input membership functions, the range of each input function is defined as

$$range_i = \max\{X_i\} - \min\{X_i\} \quad \text{for } i = 1,2,....n \tag{10}$$

If there are m uniformly distributed membership functions, then the default parameters (a_j^*, b_j^*, c_j^*) of the jth membership function is given by

$$a^* = \frac{range_i}{2m-2} \tag{11}$$

The default value of b_j^* is 2 and c_j^* is center of uniformly distributed membership function. With these details the ELANFIS algorithm can be summarized in the following steps:

Step1: Randomly assign the premise parameters (a_j, b_j, c_j) within the following ranges.

$$\frac{a_j^*}{2} \le a_j \le \frac{3a_j^*}{2}$$

$$\left(c_j^* - \frac{d_{cc}}{2}\right) < c_j < \left(c_j^* + \frac{d_{cc}}{2}\right)$$

where d_{cc} is distance between two consecutive centers b_j is selected from the range 1.9 to 2.1.

Step2: Calculate the premise layer output matrix H in (8).

Step3: Calculate the linear parameter matrix β using (9).

Step 4: Training runs are repeated for 50-70 times to select the best model.

The algorithm is simple compared to the computationally involving HLA in conventional ANFIS. Unlike in ELM networks, the qualitative knowledge embedded in the premise part of the fuzzy rules reduces the randomness of the first layer parameters.

D. Example 1- Modeling a three-input nonlinear function:

In this example [7], ELANFIS is used to model a three-input *nonlinear* function

$$output = (1 + x^{0.5} + y^{-1} + z - 1.5)^2 \tag{12}$$

where x, y and z are inputs with range [1, 6]. The training data consists of 216 instances. The ELANFIS used here contains 8 rules with 2 membership functions being assigned to each input variable. The testing data consists of 125 instances with input in the range [1.5, 5.5]. Fig. 2 shows the initial and final membership functions of inputs. Training time and testing time of both ANFIS and ELANFIS are listed in Table 1.

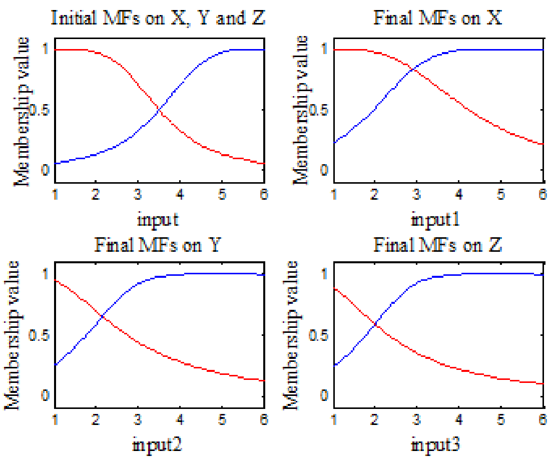

Fig. 2. Initial and final membership functions of *x*, *y*, *z*

It is clearly seen that with the increase in the number of membership functions, the accuracy increases. But in the case of conventional ANFIS algorithm, as the number of membership functions increases, training time also increases. The ELANFIS algorithm is faster with comparable training and better testing performance.

E. Example 2- The residuary resistance of yatchs problem:

Prediction of residuary resistance of a sailing yacht at the initial design stage is of a great value for evaluating the yacht's performance and determining the required propulsive power. The data set contains 308 instances with six input features which include the basic hull dimensions and the boat velocity [10].The data is divided in the ratio 50: 50 for training and testing. Four membership functions are assigned to each of these six variables. Fig. 3 shows the linear regression analysis performed between the output predicted by the ELANFIS and the target. The correlation coefficient R^2 is with ELANFIS is 0.9916 which is better than given in [11] where the problem is solved using a 6:9:1 multilayer perceptron. When the conventional ANFIS algorithm is used to solve this example, the computer which uses an Intel Core i3-2350M CPU @ 2.30 GHz run out of memory because of the large number of rules used. ELANFIS can also face this shortcoming, when number of input features is increased.

TABLE I. PERFORMANCE ANALYSIS OF EXAMPLE 1

	Number of membership functions	Training Time (s)	Training error (RMSE)	Testing error (RMSE)
HLA	2	0.733	0.0254	0.3128
ELANFIS		1.023	0.0942	0.3057
HLA	4	23.607	3.5e-4	2.1263
ELANFIS		3.072	0.0023	0.4825
HLA	5	97.064	7.9e-5	4.5413
ELANFIS		5.112	1.78e-4	1.4510

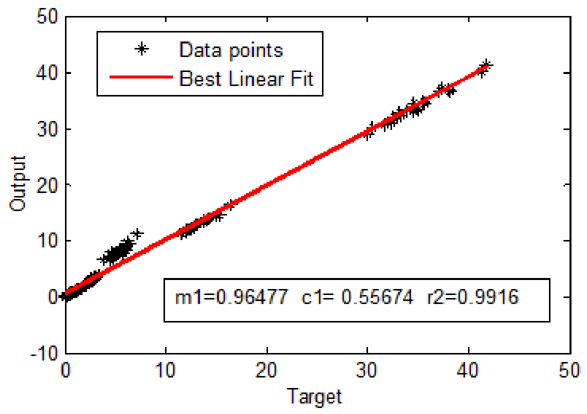

Fig. 3. Linear regression analysis plot of example 2

III. INVERSE MODEL CONTROL USING ELANFIS

In inverse model control [12], the system with inverse dynamics of original system is connected in series with the original system. The method assumes the existence of unique and stable inverse of system. Neuro-fuzzy approaches do not lead to an analytical control formulation and stability analysis is difficult because the black box nature of of neuro-fuzzy models. Lypanov based stability analysis can be used to study the closed stability of neuro-fuzzy controllers [13-14]. Here, ELANFIS is used to get inverse dynamic model of a conical tank system [15]. The system model is shown in Fig. 4.

The inlet flow Q_{in}, is the control input and water level h is the controlled output. The limit given in (13) for inlet flow, is the most important constraint on controlling input, which limits rate of change of reference.

$$0 \le Q_{in} \le 5\times10^{-4}\, m^3/s \qquad (13)$$

Q_{out} is outlet flow, which depends on the hydrostatic pressure and cross-section area of outlet a_{out}, and is given as

$$Q_{out} = a_{out}\sqrt{2gh} = k_{out}\sqrt{h} \qquad (14)$$

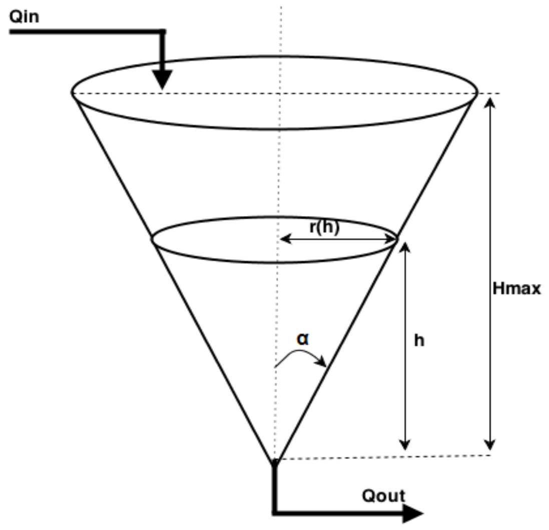

Fig. 4. Conical tank system

k_{out} is a constant and its value is given as 6×10^{-4}. The maximum height of tank is 0.4 m and the angle α=20°. The volume of water in tank is

$$V(h)=\tfrac{1}{3}\pi r^2(h)h=\left[\tfrac{1}{3}\pi\tan^2\alpha\right]h^3 \tag{15}$$

where $r(h)$ is a radius of tank at corresponding water level h; thus, $r(h)$ = h $tan(\alpha)$. By using law of conservation of mass, the rate of change of volume is given as

$$\frac{dV(h)}{dt}=Q_{in}-Q_{out}=Q_{in}-k_{out}\sqrt{h} \tag{16}$$

Taking derivative of (15)

$$\frac{dV(h)}{dt}=\left[\tfrac{1}{3}\pi\tan^2\alpha\right]\frac{dh^3}{dt}=\left[\pi\tan^2\alpha\right]h^2\frac{dh}{dt} \tag{17}$$

Equations (16) and (17) give system dynamics as

$$\dot{h}=\frac{dh}{dt}=f(h,Q_{in})=\frac{1}{\pi\tan^2\alpha}h^{-2}\left[Qi_n-k_{out}\sqrt{h}\right] \tag{18}$$

Discrete form of system equation is needed for designing of controller, the discretized *equation* is given as

$$h(k+1)=\frac{T}{\pi\tan^2\alpha}\left[h(k)^{-2}Q_{in}(k)-k_{out}h(k)^{-\frac{3}{2}}\right]+h(k) \tag{19}$$

The discretized plant model finds state h (k+1) using current state h (k) and randomly varying inlet flow Qin (k). The training data samples are obtained as

$$\left[h(k)^T, h(k+n)^T; Q_{in}(k)^T\right] \tag{20}$$

After obtaining the inverse model, it is connected in series with the plant as controller as in Fig. 5

Input-output data for training the inverse model is shown in Fig. 6. 500 pairs of training data are used for training. ANFIS structure is generated for different number of standard bell shaped membership functions with two inputs and linear consequent part which generates m^n rules with m×n×3=30 premise parameters and 3×m^n consequent parameters. These parameters are adjusted to tune membership function with the help of extreme learning algorithm in learning phase.

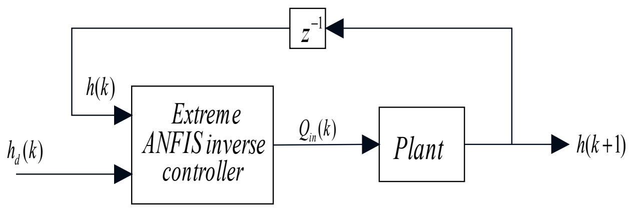

Fig. 5. Block diagram of inverse model control

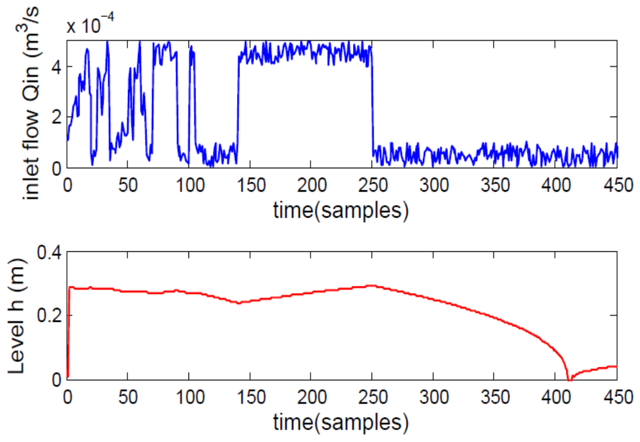

Fig. 6. Training data pair

After designing the inverse model controller the system is subjected to tracking references. The tracking signal used as reference is

$$h_d(k)=0.21+0.05\sin\left(\frac{0.2\pi k}{250}\right)+0.025\sin\left(\frac{0.2\pi k}{50}\right)$$

Error graphs for tracking error of ELANFIS and ANFIS are shown in Fig. 7 for 3 membership functions which shows that ELANFIS has better generalization ability than conventional ANFIS.

The overall performance analysis for different number of membership functions is listed in Table 2. It is seen from observed results, as the number of membership functions increases hybrid learning algorithm (HLA) gives reduced performance error. But in the proposed algorithm generalization is not much affected while improving the accuracy in terms of training error. Learning times and root mean square errors (RSME) listed in the table are obtained by taking average of 15 trials.

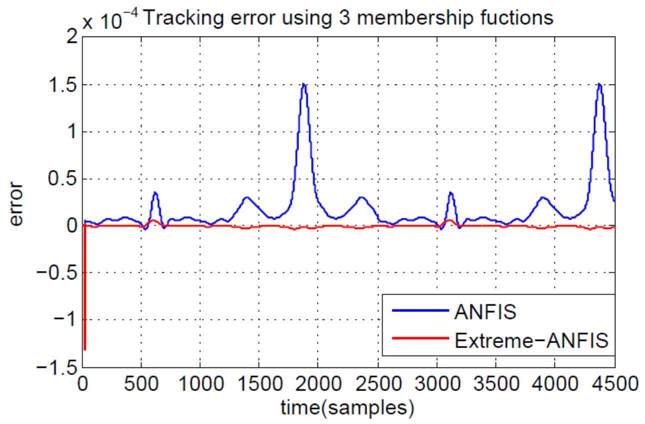

Fig. 7. Tracking error comparison with 3 membership functions

TABLE II. PERFORMANCE ANALYSIS OF INVERSE CONTROL MODEL

	Number of membership functions	Training Time (Sec)	Training error (RMSE)	Performance error (RMSE)
HLA	3	1.424	4×10^{-6}	3.3474×10^{-5}
Proposed		1.513	7.093×10^{-8}	6.4653×10^{-6}
HLA	5	6.210	1.8×10^{-6}	4.4115×10^{-5}
Proposed		2.615	4.5709×10^{-9}	1.7023×10^{-5}
HLA	7	20.753	1.6×10^{-6}	3.868×10^{-5}
Proposed		4.363	2.1813×10^{-10}	9.3314×10^{-6}

IV. MODEL PREDICTIVE CONTROL USING ELANFIS

The basic structure of ELANFIS based nonlinear model predictive controller (MPC) is shown in Fig. 8. It includes three important blocks, first is the actual plant to be controlled with output $y(k)=[y_1(k),\ldots y_M(k)]^T$. Then the extreme learning ANFIS model of the actual plant to be controlled with predicted output $\hat{y}(k)= [\hat{y}(k+1)\ ,\ldots\ \hat{y}(k+Np)]$ here, N_p is the prediction horizon of MPC which dictates how far we wish the future to be predicted for. Next is the optimization block which provides the optimized control signal $u(k)=[u_1(k),\ldots u_{Nu}(k)]$ where N_u is the control horizon of MPC which dictates the number of control moves used to attain the future control trajectory, subjected to the specified constraints which is required for the plant to achieve the desired trajectory $ref(k+1)=[ref_1(k+1)\ \ldots.ref_{Np}(k+1)]$. Here k stands for the current sampling instant.

Thus at each sampling instant a sequence of manipulated variable $u(k)$ is calculated in order to minimize the formulated performance index in (23) i.e. the difference between the predicted output of the model and the desired reference trajectory over the specified prediction horizon N.

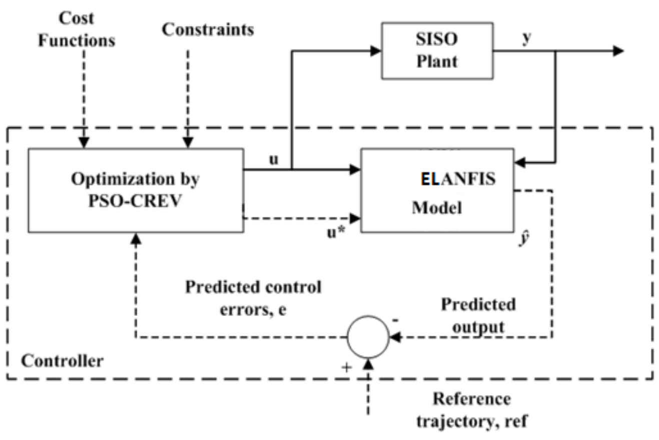

Fig. 8. Structure of ELANFIS based model predictive control

The number of manipulated variable in the sequence is decided by the control horizon value Nu and only the first manipulated variable in the control sequence is applied to the actual plant. This course is repeated at each sampling instant.

For a single input single output (SISO) nonlinear process the predicted output of ELANFIS model is a function of past process outputs, $Y(k)=[y(k)\ldots..y(k-n_y+1)]$ and past process inputs, $U(k-1)=[u(k-1)\ldots u(k-n_u+1)]$. The number of past controlled outputs and past manipulated inputs depends on the corresponding process orders n_u and n_y respectively.

Thus a single step ahead prediction of a SISO nonlinear process can be illustrated by the following discrete time model,

$$\hat{y}(k+1)=f[y(k),.\,.y(k-n_y+1),u(k),..\,.u(k-n_u+1)] \quad (20)$$

where k is the discrete time index

The above equation can be rewritten as

$$\hat{y}(k+1)=f[Y(k),u(k),U(k-1)] \quad (21)$$

Here, $Y(k)$ and $U(k-1)$ are the vectors holding the past controlled outputs and past manipulated inputs respectively. Thus after system identification using the regression data set the one step ahead predicted output of ELANFIS model could be formulated as

$$\hat{y}=\sum_{k=1}^{m_n}\overline{W}_k\left(\sum_{i=1}^{n}R_{ki\,I_i}+Q_k\right) \quad (22)$$

Accordingly, the performance index to be minimized to achieve the optimal control sequence $u(k)$ can be obtained as shown below,

$$J[u(k)]=\sum_{j=N_1}^{N_2}\left[ref(k+j)-\hat{y}(k+j)\right]^2+\sum_{j=1}^{N_u}\lambda\left[\Delta u(n+j)\right]^2 \quad (23)$$

In the performance index formulated in (23) $\hat{y}$ is a function of membership function and rules related to the manipulated variable u, which is optimized and applied to the actual plant in order to minimize the deviation between the reference value and controlled variable.

N_1	-	minimum prediction horizon
N_2	-	maximum prediction horizon
N_u	-	control horizon
$ref(.)$	-	reference trajectory
$\hat{y}(.)$	-	predicted output of Extreme ANFIS model
Δu	-	control input change defined as $u(k+j)-u(k+j-1)$
k	-	current sampling instant
λ	-	control input weighting factor

ELANFIS algorithm is applied for model predictive control of a catalytic continuous stirred tank reactor (CSTR) process. In a catalytic reactor, the rate of catalytic reaction is proportional to the amount of catalyst the reagents contact.

The dynamic model of the Catalytic CSTR process is,

$$\frac{dh(t)}{dt} = q_1(t) + q_2(t) - 0.2\sqrt{h(t)} \quad (24)$$

$$\frac{dC_b(t)}{dt} = (C_{b1} - C_b(t))\frac{q_1(t)}{h(t)} + (C_{b2} - C_b(t))\frac{q_2(t)}{h(t)} - \frac{k_1 C_b(t)}{(1+k_2 C_b(t))^2} \quad (25)$$

where

h(t)	- liquid level in the reactor,
$C_b(t)$	- product concentration at the output of the process
$q_1(t)$	- flow rate of the concentrated feed C_{b1},
$q_2(t)$	- flow rate of the diluted feed C_{b2},
q_0	-product flow rate at the output of the process.

The nominal conditions for the feed concentrations are set to C_{b2} = 24.9 mol L^{-1} and C_{b1} = 0.1 mol L^{-1}. The rate of consumption of both the feeds are associated with the constants k_1=1 and k_2=1. The objective of this catalytic reactor is to obtain the desired concentration C_b of the product by adjusting the feed flow rate q_1 and q_2.The illustration is made simple by setting $q_2(t)$ = 0. 1 L min^{-1}.

Randomly generated 300 training data pairs are used to train the ELANFIS network offline. Three bell shaped membership functions with 81 rules are used for adapting parameters of ANFIS. The training error of the model is 0.0303 and its corresponding testing error for unseen data is 0.0605 which shows its good generalization ability. The optimization of performance index is done using a modified form of particle swarm optimization (PSO) known as PSO-CREV [16-17]. Faster convergence is the reason for modifying the basic PSO. See more details of PSO-CREV method in [17].

The offline trained and tested ELANFIS model is then used as the nonlinear model for nonlinear MPC. The minimum and maximum value of prediction horizon N1, N2 and control horizon Nu of ELANFIS based MPC are set to 1, 1 and 1 correspondingly. The predictive controller presented in this paper is a single step ahead predictive controller, which does one step ahead prediction at each sampling instant. Since the process under simulation is a low order process and as the ELANFIS model is accurate a single step ahead prediction is sufficient to provide satisfactory tracking performance. The control input weighting factor λ is set to 0.5. which decides the smoothness of control signal (manipulated variable) which in turn smoothens the controlled variable. With the selected MPC parameters, satisfactory tracking performance is achieved with minimum computation burden.

Fig. 9 illustrates the random set point tracking performances of ELANFIS model based MPC with PSO-CREV. For comparison, tracking performance of a neural network based MPC is also simulated. It is clear from Fig. 9 that ELANFIS based MPC settles without overshoot. In NN based MPC, single hidden layer neural network is used to train the data. 20 hidden nodes with *tansig* activation functions are used in the neural network. Levenberg–Marquardt algorithm is used for training the neural network. PSO-CREV is used in both MPCs for optimization. A population method like PSO which requires numerous evaluations of the cost function .may not be suitable when considering an online solution to the optimization problem. To address this problem, faster convergence is obtained by deleting the additional stochastic term in the velocity update equation of PSO-CREV and by selecting the parameters as suggested in [17].

Table 3 shows the performance indices of ELANFIS and NN based model predictive controllers. The Integral absolute error (IAE) value and MPC computational time related to each controller for the simulation results are shown in the table. ELANFIS based MPC gives reduced error with less computational cost. Sample size is also less in ELANFIS. Table 4 gives the model accuracies of both controllers. Training and testing errors are small in the case of ELANFIS even after using reduced number of training data.

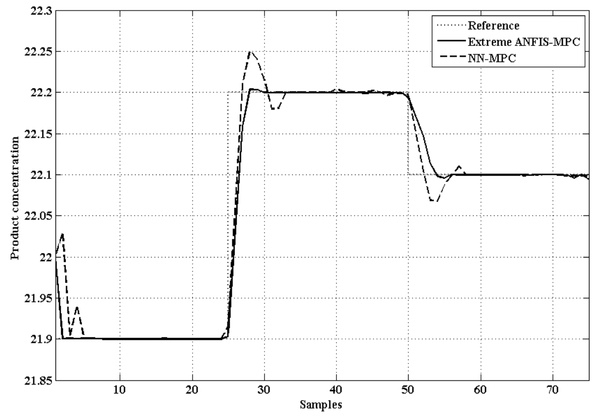

Fig. 9. Tracking performance of product concentration for CSTR process

TABLE III. PERFORMANCE INDICES OF NN AND ELANFIS MPCS

Control tactics	**Number of Training Samples**	**IAE**	**Computational time (Seconds)**	**Relative computational time**
ELANFIS-PSO-CREV	300	0.7701	5.21	1
NN-PSO-CREV	1000	0.982	21.23	4.07

TABLE IV. MODEL ACCURACIES OF NN AND ELANFIS MPCS

Model	**Number of Training data**	**RMSE$_{training}$**	**RMSE$_{testing}$**
NN	1000	0.0312	0.0989
ELANFIS	300	0.0303	0.0605

V. CONCLUSIONS

Extreme learning ANFIS is a simple learning technique which avoids the learning complexity of conventional ANFIS and back-propagation neural networks. ELANFIS also avoids the randomness of the results inherent in ELM networks by incorporating explicit knowledge representation using fuzzy membership functions. In this paper, it is shown that ELANFIS can be successfully used in control applications. Because of better generalization ability, compared to conventional neural network architectures, ELANFIS is a better choice for modeling systems. ELANFIS model accuracy increases with the increase in number of membership functions of input variables. In control applications where high model accuracy is needed ELANFIS can give better results with less computation time. However, if the number of inputs is high, number of rules will be large and like conventional ANFIS, ELANFIS can also suffer from curse of dimensionality. This problem can be alleviated by adopting suitable strategy for partitioning the input space.

REFERENCES

[1] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, ‘Extreme learning machine: theory and applications’, Neurocomputing 70 (2006) 489–501.

[2] G. B. Huang, H. M. Zhou, X. J. Ding and R. Zhang. Extreme learning machine for regerssion and multiclass classification. IEEE Trans. On Syst., Man, and Cyber., Part B: Cyber., 42(2): 513-529, 2012.

[3] G. B. Huang, D. H.Wang and Y. Lan. Extreme learning machines: a survey. Int. J. Mach. Learn. Cyber. 2(2): 107-122, 2011.

Li, Y. Li, X. Rong, ‘The extreme learning machine learning algorithm with tunable activation function’ Neural Computing and Applications, (2013) 22:531–539

[4] E. Parviainen, J. Riihimäki, Y. Miche, A. Lendasse ‘Interpreting Extreme Learning Machine as an Approximation to an Infinite Neural Network’, Proceedings of the International Conference on Knowledge Discovery and Information Retrieval, Valencia, Spain, October 25-28, 2010

[5] S. Mitra and Y. Hayashi, “Neuro-fuzzy rule generation: Survey in soft computing framework,” IEEE Trans. Neural Netw., vol. 11, no. 3, pp. 748–767, May 2000.

[6] J.-S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference systems," *IEEE Trans. Svst., Man, Cybern.,* vol. 23, no. 3, pp. 665-685, 1993.

[7] G. N. Pillai, ‘Extreme ANFIS: A New Learning Machine for Faster Learning’ Proceedings of IEEE International Conference on knowledge collaboration in engineering, Coiimbatore, January 2014.

[8] Jagtap Pushpak, G. N. Pilai, “Comparison of extreme-ANFIS and ANFIS networks for regression problems” IEEE International Conference on Advance Computing (IACC), pp. 1190 – 1194, 21-22 Feb. 2014, Gurgaon.

[9] R. Lopez. Flood: An open source neural networks C++ library. www.cimne.com/flood , 2008.

[10] I Ortigosa, R. Lopez, and J. Garcia, “A neural networks approach to residuary resistance of sailing yachts prediction”. Proceedings of the International Conference on Marine Engineering, MARINE 2007, Barcelona, June 2007.

[11] Cabrera, J. B. D. & Narendra, K. S., “Issues in the application of neural networks for tracking based on inverse control”, *IEEE Trans. Automat .Contr., ,* 1999, pp. 2007-2027.

[12] Li-Xin Wang “Stable Adaptive Fuzzy Control of Nonlinear Systems” IEEE Transactions On Fuzzy Systems, Vol. 1, No. 2, May 1993

[13] J. T. Spooner and K. M. Passino “Stable Adaptive Control Using Fuzzy Svstems and Neural Networks” IEEE Transactions On Fuzzy Systems, Vol. 4, No. 3, August 1996

[14] Bhuvaneswari, N. S., Uma, G. & Rangaswamy, T. R., “Adaptive and optimal control of a non-linear process using intelligent controllers”, Applied Soft Computing, pp., 2009, pp. 182-190.

[15] Germin Nisha, G. N. Pillai, “Nonlinear model predictive control with relevance vector regression and particle swarm optimization”, Journal of Control Theory and Applications November 2013, Volume 11, Issue 4, pp 563-569.

[16] Xin Chen, Yangmin Li. A Modified PSO Structure Resulting in High Exploration Ability With Convergence Guaranteed. IEEE Transactions on systems, man and cybernetics, 2007, 37(5) 1271-1289