International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:2, 2008

Modified Functional Link Artificial Neural Network

Ashok Kumar Goel, Suresh Chandra Saxena, and Surekha Bhanot

Abstract**—** In this work, a Modified Functional Link Artificial Neural Network (M-FLANN) is proposed which is simpler than a Multilayer Perceptron (MLP) and improves upon the universal approximation capability of Functional Link Artificial Neural Network (FLANN). MLP and its variants: Direct Linear Feed-through Artificial Neural Network (DLFANN), FLANN and M-FLANN have been implemented to model a simulated Water Bath System and a Continually Stirred Tank Heater (CSTH). Their convergence speed and generalization ability have been compared. The networks have been tested for their interpolation and extrapolation capability using noise-free and noisy data. The results show that M-FLANN which is computationally cheap, performs better and has greater generalization ability than other networks considered in the work.

Keywords**—**DLFANN, FLANN, M-FLANN, MLP

I. Introduction

MULTILAYER Perceptrons (MLPs) are the most common type of neural networks employed in process modeling. MLPs have been successfully applied for adaptive identification and control of a variety of nonlinear processes [1]-[6]. Direct Linear Feed-through (DLFANN) neural network combines conventional MLP neural network architecture with a set of linear terms to produce a network for modeling both linear and nonlinear systems simultaneously. Lee and Holt applied direct linear feed-through (DLF) network for modeling of spectroscopic process data [7]. DLF neural network offers many advantages over the conventional multilayer feed-forward networks for process modeling and control [8]-[9]. In Functional Link Artificial Neural Networks (FLANNs), the hidden layer is removed without giving up non-linearity by providing the input layer with expanded inputs that are constructed as the functions of original attributes [10]. Removal of hidden layer makes these networks extremely simple and computationally cheap. Identification of nonlinear processes using FLANNs has been reported by researchers [11]-[14]. FLANNs have an inherent limitation, of not guarantying universal approximation, which has deterred interest in them. Only a few applications using FLANNs are available in literature. Therefore, this has been a major motivating factor for modifying FLANN and improving upon its approximation ability. In this work, a Modified Functional Link ANN (M-FLANN) is proposed which is not only simpler than a MLP but also improves upon the universal approximation capability of FLANNs. MLP and its variants (DLFANN, FLANN, M-FLANN) have been implemented to model a simulated Water Bath System and a Continually Stirred Tank Heater (CSTH). The convergence speed, interpolation, and extrapolation ability of the four networks is verified.

Manuscript received February 27, 2006.

Ashok Kumar Goel is with GZS College of Engineering & Technology, Bathinda, INDIA 151 001. (phone: +91-175-2221697; fax: +91-164-2280164; e-mail: ashokkgoel@rediffmail.com).

Suresh Chandra Saxena is Director of Thapar Institute of Engineering & Technology, Patiala, INDIA 147 001 (phone:+91-175-2393201, e-mail: saxenasuresh@yahoo.co.in). Surekha Bhanot is with the Instrumentation Group, Birla Institute of Technology & Science, Pilani, Rajasthan, INDIA 133 031 (e-mail:

Surekha Bhanot is with the Instrumentation Group, Birla Institute of Technology & Science, Pilani, Rajasthan, INDIA 133 031 (e-mail: surekha_dv@yahoo.co.in).

II. Multilayer Perceptron and Its Variants

A. Multilayer Perceptron

In ANNs, neurons are arranged into groups called layers. The name Multilayer Perceptron itself indicates that there are multi i.e. more than one layers of neurons. There are usually layers of neurons called, hidden layers between the input layer and the output layer. .

B. Direct Linear Feed-through Artificial Neural Network

A process may exhibit both linear and/or nonlinear behavior over its operating range. Thus, it seems more appropriate to have a network structure that is capable of handling both linear and nonlinear systems. The DLFANN combines conventional neural network architecture with a set of linear terms to produce a network which can handle both linear and nonlinear behavior simultaneously.

The DLFANN network is shown in Fig. 1. In DLFANN, neurons in input layer are not only connected to the first hidden layer in the network, but are also connected directly to neurons in the output layer (shown by thick solid lines). These direct interconnections between input and output neurons are responsible for handling linear terms in the function to be approximated.

DLFANNs are better at approximating functions that contain both linear and nonlinear terms. But the computational complexity of network is increased as weights corresponding to direct interconnection between the input and output neurons are added. This results in the training to be slower than in MLP. It also increases algorithmic complexity. Another factor that cannot be ignored is that the added complexity may be useful only in case where the function to be approximated is considerably linear. In fact, for nonlinear systems, it may give only marginal improvement over MLPs.

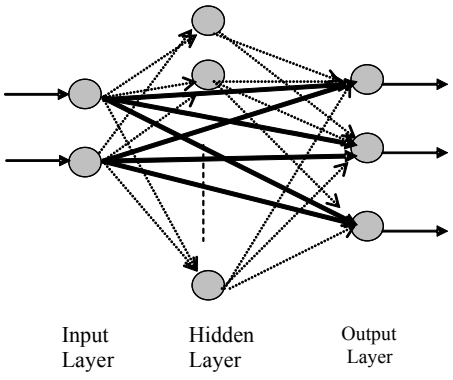

Fig. 1 Direct Linear Feed-through Artificial Neural Network (DLFANN)

C. Functional Link Artificial Neural Network

The function approximation capability of FLANNs can be understood as follows:

Take a MLP with d = 2 input units and h = 3 sigmoidal hidden units in the lone hidden layer as an example. The output function calculated for this neural network is

$$y_k = G_k\left(\sum_{j=1}^{h} W_{jk} * \Phi_j(x)\right) \quad (1)$$

where W_{jk} is the weight connecting hidden unit j with output unit k and G_k the activation function employed by the output layer neurons.

The hidden layer units calculate a projection of original input space into an intermediate one by means of hidden layer weights, W_{ij}.

$$(X_1,....X_i,...X_d) \rightarrow (\Phi_1(x),...\Phi_j(x),...,\Phi_h(x)) \quad (2)$$

$$\Phi_j(x) = G_j\left(\sum_{j=1}^{d} W_{ij} * x_i\right) \quad (3)$$

G_j is the activation function used at hidden layer neurons.

In this hidden space, linear discrimination, to be carried out by the output weights becomes easier than in the original input space. By contrast, linear networks used in FLANN take hidden units to the input layer and work with a single layer of weights. The output function described by (1) is modified as

$$y_k = G_k\left(\sum_{j=1}^{d'} W_{ik} * \Psi_i(x)\right) \quad (4)$$

where d' is the dimension of the new input space $((\psi_1(x),...\psi_i(x),...,\psi_{d'}(x))$.

The new units $\psi_i(x)$, instead of being learnable arbitrary functions of the original attributes as in (3), are now fixed polynomial and trigonometric terms constructed out of the original attributes. For example, a possible input layer for the two-dimensional problem could be $(x_1, x_2, x_1^2, x_1x_2, x_2^2)$. The FLANN network is shown in Fig. 2. The corresponding FLANN architecture is capable of capturing nonlinear

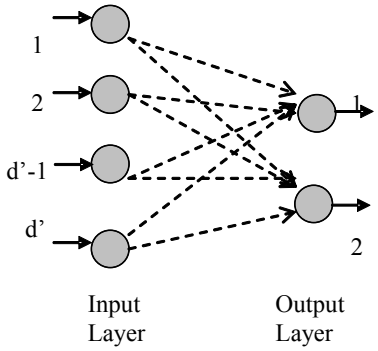

Fig. 2 Functional Link Artificial Neural Network (FLANN)

relationships between its original input and output patterns. Linear mapping in this polynomial space is, in fact, nonlinear in the original input space.

FLANNs do not have any hidden layer and the computational cost, in fact, moves from the hidden layer to selection of suitable expanded inputs for the input layer. The expanded inputs are chosen using an evolutionary technique, which makes use of Genetic Algorithms and gradually evolves inputs of the FLANN to achieve the desired model [15]. The order of the polynomial used to obtain expanded inputs can be gradually increased so that minimum number of inputs is used and the complexity of the neural network is minimized. The evolutionary algorithm is reproduced below in brief:

1. The evolutionary algorithm begins from original input attributes. This makes sense since some problems can be solved linearly, perhaps after rejecting some noisy or irrelevant attributes.
2. Each input vector is encoded by means of a binary chromosome of length equal to the number of available polynomial terms. A bit 1 specifies that corresponding polynomial term is fed into the network.
3. Instead of choosing initial random population, it starts from a pool of single feature networks. For example if the system to be modeled has five inputs, the following initial pool of chromosomes is taken: [(1 0 0 0 0), (0 1 0 0 0), (0 0 1 0 0), (0 0 0 1 0) and (0 0 0 0 1)].
4. Roulette-wheel selection and single-point crossover have been employed in the GA used. The crossover and mutation probability is fixed at 0.9 and 0.05 respectively.
5. The maximum number of generations has been fixed at 50.
6. If the error reached by the best individual in the population on the validation set is not satisfactory, then the order of the polynomial terms is raised by one. For example, system having two attributes x_1

and x_2 will yield five polynomials of degree two, i.e. $(x_1, x_2, x_1^2, x_1x_2, x_2^2)$.

7. The number of terms grows very quickly with the degree of product or trigonometric polynomial. For CSTH with 6 input attributes, the number of product polynomial terms increase from 27 to 83 as the degree is raised from 2 to 3. The search space dimension increases from 2^{27} to 2^{83}. Therefore, it is not recommended to increase the degree of product polynomial terms beyond 2. Instead the convergence is improved by including trigonometric polynomial basis terms given by
   $\langle \cos(\pi x), \sin(\pi x), \cos(2\pi x), \sin(2\pi x),$
   $....., \cos(m\pi x), \sin(m\pi x) \rangle$
   where m is the order of polynomial.
8. The process is repeated with increased number of polynomial terms. The best individual obtained in previous evolution run is also included in the new population. The algorithm is run till the error goal is achieved or the degree of polynomial becomes prohibitively high.

As FLANNs do not have any hidden layer; the architecture becomes simple and training does not involve full backpropagation. Thus, nonlinear modeling can be accomplished, by means of a linear learning rule, such as delta rule. The computational complexity is also reduced and the neural net becomes suitable for on-line applications. Further, it reaches its global minima very easily. As FLANNs involve linear mapping in polynomial space, they can easily map linear and nonlinear terms. Notwithstanding advantages accrued, unlike MLP, these networks lack universal approximation capability.

III. Modified Functional Link Artificial Neural Network

In this work, the FLANNs have been modified to improve upon their approximation ability while still maintaining advantages obtained by reduction in computational complexity.

Following are the modifications:

1. Self-feedback weighted interconnections are added at the output layer.
2. Lateral feedback interconnections are added at the output layer.
3. Output neurons first process expanded inputs by applying appropriate activation function as was being done in the FLANN and then combine the self and lateral feedback outputs only for calculating the outputs.

The structure of the M-FLANN network is shown in Fig. 3. The solid lines in the figure indicate the modifications added. It can be seen that the two output neurons have self-feedback and lateral connections. For single output systems, a dummy output needs to be added for successful implementation of M-FLANN. Further, though the network still has no hidden layer, it is no longer a true feedforward neural network as feedback connections are present.

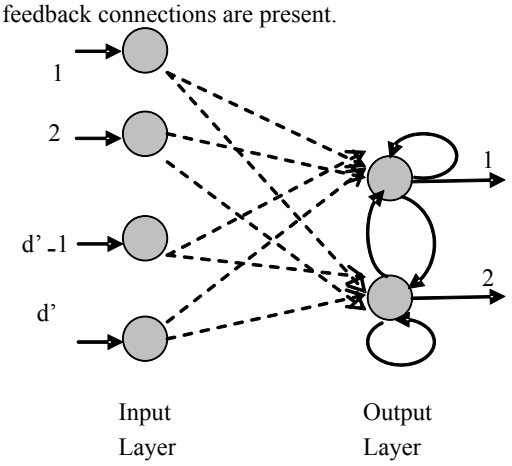

Fig. 3 M-FLANN Network

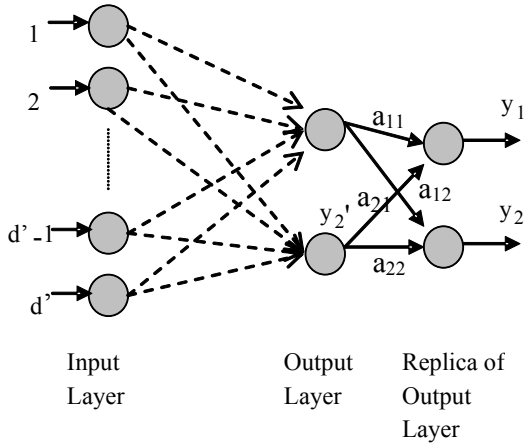

Fig. 4 M-FLANN, Equivalent Feedforward Neural Net

The output neurons of M-FLANN in Fig. 3 contain self and lateral feedback connections. A single hidden layer MLP neural net in Fig. 4 is equivalent to the M-FLANN network of Fig. 3. The hidden layer of equivalent feedforward network simulates the first stage of output computation of M-FLANN whereby it computes the output y'(y_1',y_2') from the original FLANN inputs $\psi_i(x)$, using the equation

$$y'_k = G'_k\left(\sum_{j=1}^{d'} W_{ik}.\psi_i(x)\right) \quad (5)$$

The output layer neurons have linear activation function, as their purpose is only to represent the combination of outputs from the self and lateral feedback connections in the output layer of M-FLANN, given by equations below.

$$y_1 = (a_{11}.y'_1 + a_{21}.y'_2) \quad (6)$$

$$y_2 = (a_{12}.y'_1 + a_{22}.y'_2)$$

$$y = (y_1, y_2)$$

where a_{11}, a_{12}, a_{21} and a_{22} are lateral and self feedback interconnection weights as shown in Fig. 4.

Leaving out inputs at this stage saves significant computational overhead as well as the unnecessary repeated computation step. The equivalent network is akin to a Multilayer Perceptron, and therefore, can make use of any gradient descent based learning algorithm for training. The inputs of M-FLANN can be very conveniently determined using an evolutionary algorithm that makes use of GAs to evolve expanded set of inputs, which are product and trigonometric polynomial terms of original inputs. Details of the algorithm have been described in the previous section.

The novel proposed neural network is superior to other neural architectures in many respects. M-FLANN has a systematically laid down procedure for selecting its architecture details. It has better convergence speed than MLP, DLFANN and FLANN. It has no hidden layer, is computationally cheap and has small number of interconnection weights and biases, which makes it suitable for on-line applications. Also, M-FLANN makes use of linear terms, just as used in FLANNs and is able to approximate linear as well as nonlinear functions. M-FLANN improves upon the approximation ability of FLANN. Therefore, it is expected that M-FLANN is likely to re-ignite the interest of research community in FLANNs.

IV. Water Bath System Model Simulation Details

Water Bath System is a single-input single-output system. However, as different assumptions taken do not hold in practice, system components do not behave perfect linearly. A mathematical model has been developed for a Water Bath System with the following specifications: tank capacity- 12 litres, inlet temperature- 25˚C, base heater wattage- 2500 watts, inlet/outlet flow rate- 1 litre/min, stirrer speed- 120 rpm, sampling period- 30 seconds and system time delay- one sample. The control objective is to regulate the temperature of water in the tank.. The four ANN architectures discussed in sections II and III have been used to model the process.

Water Bath System is represented by the equation:

$$dT/dt = F/V.\left(T_i - T\right) + Q/V.C_p.\rho \quad (7)$$

where T is the tank temperature, F the flow rate, T_i the inlet temperature, V the volume of the tank, Q the heat input, C_p the specific heat and ρ the density of water.

A. Network Architecture

1) Multilayer Perceptron Based Model

A single hidden layer MLP is employed to model a Water Bath System. Four neurons are taken in the input layer. Two neurons account for inputs, the tank flow rate F and the tank inlet temperature T_i and the other two for the heat input to tank water, Q and the delayed output tank temperature, T_p respectively. The single neuron in the output layer is used for the output tank temperature, T. Twelve neurons are chosen for the lone hidden layer and architecture of the MLP network is 4-12-1. The total number of weights and biases is 73.

2) Direct Linear Feed-through ANN Based Model

The DLFANN used to model a Water Bath System makes use of additional linear weights. Rest of the architecture is akin to that of MLP. Thus, the architecture of the DLFANN model of Water Bath System is 4-12-1 and the number of weights and biases is increased from 73 to 77.

3) Functional Link ANN Based Model

The expanded set of inputs for FLANN is evolved using both the product and trigonometric polynomial terms obtained from original inputs. The evolutionary algorithm described in Section IIC is implemented to determine the expanded inputs terms. The evolved FLANN network architecture is 16-1 and uses a total of 17 weights and biases.

4) Modified Functional Link ANN Based Model

M-FLANN model makes use of identical expanded input terms as that of FLANN. But, M-FLANN is designed for multi-input multi-output systems having interacting equations. Therefore, a dummy output T_d, with a constant value taken as 0.5 is added. Thus, the architecture of M-FLANN controller 16-2 and the total number of weights & biases, including the self and lateral feedback connections is 40.

B. Generation of Training and Test Data

1) Generation of Noise-free Training Data

The ANN models have been trained using training samples taken from open loop response of the process. Equation (7) describing the process is solved using ODE solver in MATLAB. The heat input, Q, to the tank is varied from 250 to 2250 watts at regular steps of 250 watts. For each value of Q, five random combinations of plant inputs are generated. For each combination, the response of the system is observed for 15 minutes. Seven samples at regular time intervals of 150 seconds are taken from each response. Thus, a total of 316 input-output training samples are generated.

2) Generation of Test Data for Interpolation Test

For generation of interpolation test data, 10 combinations of four input parameters, Q, F, T_i & T_p and initial output states, T and T_p, are taken at random in ranges for which Water Bath System model is trained. The response of the processant for each set of input variables is observed for 15 minutes and 7 samples at regular intervals of 150 seconds are taken. Thereby, a total of 70 test samples for interpolation are generated.

3) Generation of Test Data for Extrapolation Test

For generation of extrapolation test data, 10 combinations of the four input parameters, Q, F, T_i & T_p and the initial output states are taken at random. Q is chosen randomly in the range varying from 0 to 250 watts and from 2250 to 2500 watts, i.e. outside the range of training data. The response of the plant for each set of input variables is observed for 15 minutes and 7 input-output samples at regular intervals of 150 seconds are taken from each response. Thereby, a total of 70 test samples for extrapolation are generated.

The ranges of input-output variables for generation of training, interpolation and extrapolation data are shown in Table 1.

4) Generation of Noisy Data for Training and Testing

In the mathematical model of Water Bath System, it is assumed that there are no losses due to heat radiation to and from the surroundings and temperature in tank is kept uniform with the help of a stirrer. In practice, this may not be exactly true. Therefore, to simulate noisy practical situations, random

TABLE 1
RANGES OF INPUT-OUTPUT VARIABLES FOR GENERATION OF TRAINING, INTERPOLATION TEST AND EXTRAPOLATION TEST DATA FOR ANN MODELS OF WATER BATH SYSTEM.

Variables*	Range					
	Training		Interpolation		Extrapolation	
	Min	Max	Min	Max	Min	Max
F	0	2	0	2	0	2
Q	250	2250	250	2250	0 2250	250 2500
T_i	23	27	23	27	20	30
T_p	35	45	35	45	30	50
T	27	85	27	85	27	85
T_d	0.5	0.5	0.5	0.5	0.5	0.5

noise (maximum ±0.5°C), is added to the tank output temperature in the noise-free input-output data sets using the random function in MATLAB.

Twenty-five sets of randomly generated weights for all the four ANNs are taken. The DLFANN weights are taken same as that of MLP with the addition of direct input to output layer weights, so that it could be verified if the addition of these linear weights leads to any improvement in results.

C. Training of ANN Based Models

MLP, DLFANN and FLANN based controller have been trained using SORRPROP training algorithm [16]. For training DLFANN, FLANN and M-FLANN, the learning algorithm is modified suitably. Although, FLANN can be trained using simple delta rule, to have fair comparison of different MLP variants, same learning algorithm has been employed for all the networks. It is all the more essential, as SORRPROP algorithm is considerably faster compared to both the delta learning rule and conventional backpropagation.

V. RESULTS AND DISCUSSIONS FOR WATER BATH SYSTEM MODEL

The first set of simulation tests is conducted using the training data to compare their convergence times. Two values of sum-squared error (SSE) goal, equal to 0.01 and 0.001, are taken to judge the short and long term convergence rates of the four ANNs. Maximum training time of 100 and 200 seconds is chosen for the short-term and long-term convergence tests respectively. The network that does not converge within the maximum prescribed time-limit is assumed to be non-converging. All four ANNs are trained using SORRPROP training algorithm. The results obtained have been tabulated in terms of minimum, maximum, and mean of convergence times taken to converge to the error goal by 25 sets of runs for each ANN model. The number of times networks do not converge is also reported. For the first set of test results given in Table 2 it is seen that FLANN gives the least mean convergence time of 1.1494 seconds. M-FLANN gives comparable performance and takes 1.7542 seconds. The MLP and DLFANN give large convergence times.

The results of simulation runs for testing the long-time convergence ability of ANN models are given in Table 3. It is observed that FLANN that gave least convergence time for SSE goal of 0.01 totally fails to converge for SSE goal of 0.001. This confirms that FLANNs although converge

TABLE 2
COMPARISON RESULTS FOR SHORT-TERM CONVERGENCE TEST FOR ANN MODELS OF A SIMULATED WATER BATH SYSTEM

ANN Type	Convergence Time in seconds				Not Conv-erged
	Mean	Min.	Max.	Std. dev.	
MLP	41.6725	7.750	100.000	28.3275	02
DLFANN	48.7444	11.656	100.000	28.4415	03
FLANN	1.1494	0.765	2.093	0.2685	00
M-FLANN	1.7542	0.500	7.984	1.5598	00

TABLE 3
COMPARISON RESULTS FOR SHORT-TERM CONVERGENCE TEST FOR ANN MODELS OF A SIMULATED WATER BATH SYSTEM

ANN Type	Convergence Time in seconds				Not Conv-erged
	Mean	Min.	Max.	Std. dev.	
MLP	145.2675	39.688	200.000	70.435	14
DLFANN	172.8769	59.453	200.000	50.971	19
FLANN	200.0000	200.000	200.000	0.0	25
M-FLANN	6.3631	0.921	28.375	7.410	00

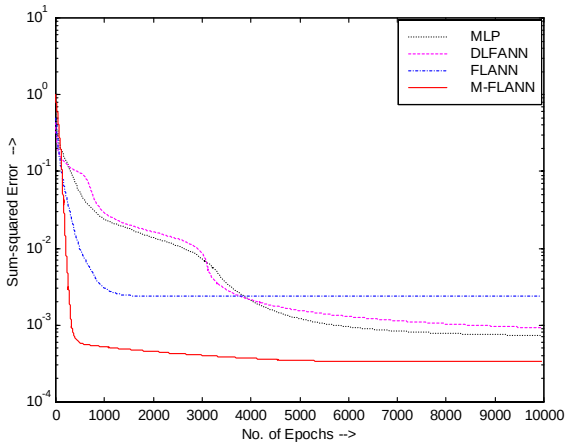

Fig. 5 Comparison of Convergence Times of ANN Models of a Simulated Water Bath System

rapidly to their global minima, lack universal approximation capability. M-FLANN gives mean convergence time of 6.3631 seconds and converges at every run. MLP and DLFANN networks not only give large convergence times but also fail to converge for most of the runs.

The configuration showing the fastest convergence speed out of the 25 runs for each ANN model is trained further and the results are plotted in Fig. 5. From the figure, it is observed that FLANN shows fastest convergence speed but reaches its global minima extremely fast and there is no further improvement in SSE thereafter. Therefore, it fails to converge for small values of the error goal. DLFANN also does not show improvement in convergence time over that of MLP.

M-FLANN network model has the smallest convergence time and outperforms all other ANN networks. M-FLANN is trained further using adaptive backpropagation algorithm to find out if it also reaches its global minima like FLANN. It is seen that unlike FLANN, the SSE value for M-FLANN continues to decrease further although at reduced pace to 0.0002314 in 10,000 epochs.

TABLE 4
COMPARISON RESULTS FOR INTERPOLATION AND EXTRAPOLATION TESTS OF ANN MODELS OF A SIMULATED WATER BATH SYSTEM TRAINED WITH NOISE-FREE TRAINING DATA

ANN Type	Interpolation		Extrapolation	
	SSE for Noise-free Test Data	SSE for Noisy Test Data	SSE for Noise-free Test Data	SSE for Noisy Test Data
MLP	0.000118	0.0338	0.0091	0.0805
DLFANN	0.000157	0.0351	0.0217	0.0816
FLANN	0.000285	0.0326	0.0136	0.0828
M-FLANN	0.000055	0.0330	0.0055	0.0969

TABLE 5
COMPARISON RESULTS FOR INTERPOLATION AND EXTRAPOLATION TESTS OF ANN MODELS OF A SIMULATED WATER BATH SYSTEM TRAINED WITH NOISE-FREE TRAINING DATA

ANN Type	Interpolation		Extrapolation	
	SSE for Noise-free Test Data	SSE for Noisy Test Data	SSE for Noise-free Test Data	SSE for Noisy Test Data
MLP	0.0137	0.0047	0.0235	0.0308
DLFANN	0.0135	0.0057	0.0321	0.0427
FLANN	0.0161	0.0039	0.0281	0.0355
M-FLANN	0.0145	0.0041	0.0147	0.0312

The next two sets of simulation tests are conducted to evaluate the interpolation and extrapolation ability of four ANN models. The configurations showing the fastest convergence speed out of the 25 runs for each type of ANN models are chosen. In the first set of these simulation tests, all ANNs except M-FLANN are trained for a fixed time of 100 seconds with noise-free training data. As M-FLANN has very small convergence times, to avoid overfitting, it is trained for 25 seconds only. The trained networks are then tested for their interpolation and extrapolation ability using noise-free and noisy interpolation and extrapolation test data sets.

The SSE obtained for each ANN model in interpolation and extrapolation tests is tabulated in Table 4. From the table, it is observed that M-FLANN network gives the minimum SSE of 0.000055 for noise-free interpolation test data compared to 0.000118 of MLP, 0.000157 of DLFANN and 0.000285 of FLANN. Similar result is observed for noise-free extrapolation test data. M-FLANN gives SSE equal to 0.0055 compared to 0.0091 of MLP, 0.0136 for FLANN and 0.0217 for DLFANN. The four ANN models perform almost similarly for tests conducted using noisy data set.

For second set of simulation tests the ANN models are trained afresh with noisy training data for 100 seconds each. M-FLANN model is trained for 25 seconds only. The SSE obtained for noise-free and noisy test data sets is tabulated in Table 5. For noise-free test data set the SSE reported by ANN models increases manifolds compared to results obtained when trained with noise-free data. From the results, it is observed that M-FLANN gives consistent performance while other ANNs give larger SSE for extrapolation test. For the noisy test data set, the performance of ANN models is almost similar. It is inferred that on the whole, performance of M-FLANN model is better than others particularly for noise-free interpolation and extrapolation test data sets.

From the results discussed above, it is evident that M-FLANN has better convergence speed and generalization ability than other three ANNs considered.

VI. MODELING OF A CONTINUALLY STIRRED TANK HEATER

The CSTH process is a MIMO system, described by a set of two interacting equations. CSTH is one of the most commonly used processes in chemical industry. In CSTH, the objective is to raise the temperature of the inlet stream to a desired value. A heat transfer fluid is circulated through a jacket to heat the fluid in tank.

The equations used to model the system are as given below:

$$dT/dt = F/V.(T_i - T) + UA.(T_j - T)/(V.\rho.C_p) \quad (8)$$

$$dT_j/dt = F_j.V_j.(T_{ji} - T_j) - UA.(T_j - T)/(V_j.\rho_j.C_{pj}) \quad (9)$$

where, T and T_j are the tank and jacket temperatures, F and F_j the tank and jacket flow rates, T_i and T_{ji} the tank and jacket inlet temperatures, U the overall heat transfer coefficient, A the area for heat transfer, C_p and C_{pj} the tank and jacket heat capacities, ρ and ρ_j the tank and jacket fluid density and V and V_j the tank and jacket volumes.

A. Network Architecture

1) Multilayer Perceptron Based Model

A single hidden layer MLP is employed to model the CSTH process. Six neurons are taken in the input layer. Four neurons account for four inputs, tank flow rate F, jacket flow rate Fj, tank inlet temperature Ti and jacket inlet temperature Tji and the other two for two delayed output states i.e. the tank and jacket temperatures, T and T_j respectively. The two neurons in the output layer are used for two output variables T and T_j. Twelve neurons are chosen for a lone hidden layer. Thus, the architecture of MLP is 6-12-2. The total number of weights and biases are 110.

2) Direct Linear Feed-through ANN Based Model

The DLFANN used to model the CSTH process makes use of additional weights, which directly connect the input layer to the output layer. Rest of the architecture is akin to that of MLP i.e. 6-12-2. It results in higher computational complexity as the number of weights is increased to 122. But the DLFANN is able to model the linearity in CSTH behavior wherever applicable.

3) Functional Link ANN Based Model

Twenty-six expanded input terms are evolved using both the second order product and third order trigonometric polynomial terms obtained from original inputs. Thus, architecture of FLANN network is 26-2 and a total of 54 weights are used.

4) Modified Functional Link ANN Based Model

M-FLANN used to model CSTH has the same number of expanded inputs as used in the FLANN. Thus, the architecture of M-FLANN network is also 26-2. Further, it adds self and lateral feedback connections in the output layer and a total of 60 weights are used.

TABLE 6
RANGES OF INPUT-OUTPUT VARIABLES FOR GENERATION OF TRAINING, INTERPOLATION AND EXTRAPOLATION TEST DATA FOR ANN MODELS OF A CONTINUALLY STIRRED TANK HEATER.

Variables*	Range					
	Training		Interpolation		Extrapolation	
	Min	Max	Min	Max	Min	Max
F	20	30	20	30	15	35
F_j	16	46	16	46	0	16
					46	56
T_i	23	27	23	27	20	30
T_{ji}	85	95	85	95	80	95
T	35	65	35	65	30	70
T_j	55	85	55	85	50	90

B. Generation of Training and Test Data

1) Generation of Noise-free Training Data

The CSTH process is modeled using all four ANNs. For generating input-output training samples, jacket flow rate, F_j is varied from 16 to 46 litres/minute, in steps of 5 litres/minute. For each value of F_j, five random combinations of three input variables, F, T_i & T_{ji} and the initial output states, T and T_j are taken. The sampling time period of the process is taken as 6 seconds. The equation (8) and (9) describing CSTH are solved using ODE solvers in MATLAB. The response for each set of input variables is observed for 5 minutes and 11 training samples at regular intervals of 30 seconds are taken. One sample corresponding to steady state conditions is also added. Thereby, a total of 386 training samples are chosen.

2) Generation of Test Data for Interpolation Test

For generation of interpolation test data set, ten combinations of four input parameters, F_j, F, T_i & T_{ji} and initial output states, T and T_j, are taken at random but limited to the ranges for which the CSTH model is trained. The response of the plant for each set of input variables is observed for 5 minutes and 11 samples at regular intervals of 30 seconds are taken from each response. Thereby, a total of 110 test samples for interpolation are generated.

3) Generation of Test Data for Extrapolation Test

For generation of extrapolation test data set, F_j is chosen randomly in the range varying from 0 to 16 litres/minute and from 46 to 56 litres/minute, i.e. outside the range of training data. Ten combinations of the four input parameters, F_j, F, T_i & T_{ji} and the initial output states, T and T_j, are taken at random. The response of the plant for each set of input variables is observed for 5 minutes and 11 input-output samples at regular intervals of 30 seconds are taken from each response. Thereby, a total of 110 extrapolation test data samples are generated.

4) Generation of Noisy Data for Training and Testing

For modeling CSTH, it is been assumed that there are no losses due to heat radiation to and from the surroundings and the temperature in tank and jacket is kept uniform with the help of a stirrer. In practice, this may not be exactly true. Therefore, noise (maximum of ±0.5°C), is added to the outputs using the random function in MATLAB, to simulate noisy practical situations. The noise-free data sets generated earlier are modified by adding noise to output tank and jacket temperatures. The ranges of input-output variables taken for generation of training and test data sets are shown in Table 6.

TABLE 7
COMPARISON RESULTS FOR SHORT-TERM CONVERGENCE TIME FOR ANN MODELS OF A CONTINUALLY STIRRED TANK HEATER

ANN Type	Convergence Time in seconds				Not Conver -ged
	Mean	Min.	Max.	Std. dev.	
MLP	125.71	24.19	240.00	79.41	06
DLFANN	39.28	5.38	181.86	38.13	00
FLANN	0.52	0.41	0.67	0.06	00
M-FLANN	1.31	1.06	1.72	0.19	00

TABLE 8
COMPARISON RESULTS FOR LONG-TERM CONVERGENCE TIME FOR ANN MODELS OF A CONTINUALLY STIRRED TANK HEATER

ANN Type	Convergence Time in seconds				Not Conver -ged
	Mean	Min.	Max.	Std. dev.	
MLP	224.28	110.19	240.00	37.36	20
DLFANN	161.59	24.55	240.00	80.26	11
FLANN	240.00	240.00	240.00	0.00	25
M-FLANN	2.21	1.65	2.67	0.22	00

VII. RESULTS AND DISCUSSION ON ANN MODELS OF A SIMULATED CONTINUALLY STIRRED TANK HEATER

Twenty-five sets of weights for all the four ANNs are generated at random. This has been done so that the results obtained on these 25 sets of weights for each network could be justifiably treated as generalized.

The first set of simulation tests is conducted using the training data to compare their convergence times. Two values of SSE goal, equal to 0.15 and 0.04, are taken to judge the short and long term convergence rates of the four networks. Maximum training time of 240 seconds is chosen. The network that does not converge within 240 seconds, is assumed to be non-converging. ANNs are trained using SORRPROP training algorithm to achieve the error goal and time taken by each to converge to set error goal is noted.

From the results for short-term convergence rates given in Table 7, it is observed that FLANN model of CSTH takes the minimum mean time (0.52 seconds) to converge to the SSE goal of 0.15. The small value of standard deviation (0.06) for FLANN model also indicates that FLANN gives very consistent results for every run. M-FLANN model also converges fast (1.31 seconds). DLFANN model is about 30 times slower. The MLP model gives the poorest performance and fails to converge for 6 of the 25 runs.

Table 8 provides results for long-term convergence test, SSE=0.04. It is observed that FLANN that showed least convergence time in first test set, fails to converge even once. This confirms that FLANN reaches its global minima really fast and does not improve thereafter. MLP does not converge for 20 out of the 25 runs. DLFANN fails half the number of times. M-FLANN gives best convergence time (2.21 seconds) and converges at every run. Its low value of standard

deviation (0.22) also indicates its consistent performance.

The training results for configurations showing fastest convergence speed out of 25 runs for each type of ANN models are plotted in Fig. 6. From the results, it is observed that M-FLANN network has the smallest convergence times. FLANN shows fastest convergence speed but reaches its global minima extremely fast and does not improve thereafter. DLFANNs perform better than MLPs, but looses in the long run. The performance of M-FLANNs is the best and when trained further, its SSE decreases to even lower than 0.0009.

The next two sets of simulation tests are conducted to evaluate the interpolation and extrapolation ability of the four ANNs. Each ANN configuration that generated minimum SSE for modeling CSTH using training data for the first set of simulation tests is chosen. In the first set of these simulation tests all ANNs are trained for a fixed time of 200 seconds with noise-free training data set. The trained networks are tested for their interpolation and extrapolation ability using noise-free and noisy interpolation and extrapolation test data sets.

The SSE obtained for each ANN model in interpolation and extrapolation tests is tabulated in Table 9. It is observed that M-FLANN gives minimum SSE (0.00009) for noise-free interpolation test data set compared to 0.0727 of MLP, 0.0727 of DLFANN and 0.0444 of FLANN. Similar result is observed for noise-free extrapolation test data. M-FLANN gives SSE equal to 0.0059 compared to 0.1173 of MLP, 0.1718 for DLFANN and 0.1834 for FLANN. The SSE reported by four ANNs for interpolation and extrapolation tests conducted using noisy data set is higher but M-FLANN gives least error and performs best. The performance of other three ANN models is almost identical.

The second set of simulation runs is conducted to test the interpolation and extrapolation ability of ANNs in presence of noise. The ANNs are trained afresh using noisy training data set for 200 seconds each. The SSE obtained for noise-free and noisy test data sets is given in Table 10. M-FLANN again scores over other three networks and gives pest performance. Interpolation results for noise-free test data set indicate SSE of 0.0310 for M-FLANN, 0.0483 for FLANN, 0.32 for DLFANN and 1.0669 for MLP. Similar results are observed for noise-free extrapolation test data set. The results for the noisy interpolation and extrapolation data sets also confirm superior performance of the M-FLANN model of CSTH.

It is therefore, concluded that M-FLANN has good convergence rate, better generalization ability and gives consistent and better performance than all other ANN models considered in this work.

VIII. Conclusions

From the results obtained in the work, it is concluded that the M-FLANN is computationally cheap and has small convergence times. The simulation tests on modeling of Water Bath System and CSTH suggest that the M-FLANN network has better interpolation and extrapolation ability in comparison to MLP, DLFANN and FLANN neural networks and is better suited for process modeling.

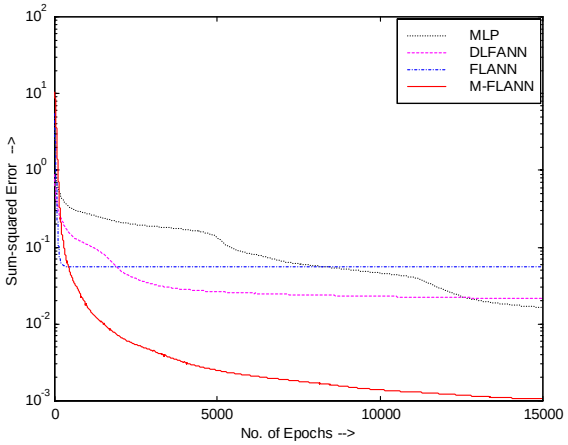

Fig. 6 Comparison of Convergence Speed of ANN Models for a Simulated Continually Stirred Tank Heater

Table 9
Comparison Results for Interpolation and Extrapolation Tests of ANN Models of a Simulated Continually Stirred Tank Heater Trained With Noise-free Data

ANN Type	Interpolation		Extrapolation	
	SSE for Noise-free Test Data	SSE for Noisy Test Data	SSE for Noise-free Test Data	SSE for Noisy Test Data
MLP	0.0727	0.2469	0.1173	0.1692
DLFANN	0.0726	0.2590	0.1718	0.2029
FLANN	0.0444	0.2137	0.1834	0.2375
M-FLANN	0.00009	0.2004	0.0059	0.0420

Table 10
Comparison Results for Interpolation and Extrapolation Tests of ANN Models of a Simulated Continually Stirred Tank Heater Trained With Noisy Data

ANN Type	Interpolation		Extrapolation	
	SSE for Noise-free Test Data	SSE for Noisy Test Data	SSE for Noise-free Test Data	SSE for Noisy Test Data
MLP	1.0669	1.2964	3.8253	3.9625
DLFANN	0.3200	0.2972	1.3524	1.3195
FLANN	0.0483	0.3225	0.2013	0.2696
M-FLANN	0.0310	0.3246	0.0409	0.0986

References

[1] Guez Allon, James L. Ellbert and Moshe Kam, "Neural networks architecture for control," *IEEE Control System magazine,* pp. 22-25, 1998.

[2] Rivals Isabelle and Léon Personnaz, "Nonlinear internal model control using neural networks: Application to processes with delay and design issues," *IEEE Trans. on Neural Networks*, vol. 11, no.1, pp. 80-90, 2000.

[3] Yu Wen and Li Xiaoou, "Some new results on system identification with dynamic neural networks," *IEEE Transactions on Neural Networks*, vol.12, no. 2, pp. 412-417, 2001.

[4] Naira Hovakimyan, Flavio Nardi, Anthony Calise, and Nakwan Kim, "Adaptive output feedback control of uncertain nonlinear systems using single-hidden-layer neural networks," *IEEE Trans. on Neural Networks*, vol. 13, no. 6, pp. 1420-1431, 2002.

[5] Sunan N Huang., K. K. Tan and T. H. Lee, "Further results on adaptive control for a class of nonlinear systems using neural networks," *IEEE Trans. on Neural Networks*, vol. 14, no. 3, pp. 719-722, 2003.

[6] Sam Ge Shuzhi and Cong Wang, "Adaptive neural control of uncertain MIMO nonlinear systems," *IEEE Trans. on Neural Networks*, vol. 15, no. 3, pp. 674-692, 2004.
[7] S. E Lee and B. R. Holt, "Regression analysis of spectroscopic process data using a combined architecture of linear and nonlinear artificial neural networks," *Neural Networks*, vol. 4, pp. 549-554, 1992.
[8] Ying-Kai Zhao, Ning Cai, "DLF NEural network applied to process modelling and control," *Proc. Pacific-Asian Conf. on Expert Systems PACES'95: Huangshan, PR China* (1995)
[9] T.X Brown., "A high performance two-stage packet switch architecture," *IEEE Transactions on Communications*, vol. 47, no. 8, pp. 1792-1795, 1999.
[10] Y. -H. Pao, "Adaptive pattern recognition and neural networks," (Addison-Wesley (1989)
[11] S. Chen and S. A. Billings, "Neural networks for non-linear dynamic system modeling and identification," *Int. J. of Control*, vol. 56, no. 2, pp. 319-346, 1992.
[12] J. C Patra., R. N. Pal and B. N.Chatterji, "Identification of non-linear dynamic systems using functional link artificial neural networks," *IEEE Trans. on Neural Networks*, vol. 29, no. 2, pp. 254-262, 1999.
[13] A Ugena., F.de Arriaga and M. El Alami, "Speaker-independent speech recognition by means of functional-link neural networks," *Int. Conf. on Pattern Recognition (ICPR'00)-Vol. 2, Barcelona, Spain* (2000)
[14] L.H.P Harada.; A.C Da Costa.; R.M Filho., "Hybrid neural modeling of bioprocesses using functional link networks," *Applied Biochemistry and Biotechnology*, vol. 98, no. 1-3, pp. 1009-1024, 2002.
[15] A. Sierra, J. A. Macias and F. Corbacho, "Evolution of functional link networks," *IEEE Trans. on Evolutionary Computation*, vol. 5, no. 1, pp 54-65, 2001.
[16] A. K Goel. and S Bhanot, "Modelling of continually stirred tank heater with ANNs using successive over-relaxation backpropagation algorithm," *ASCC'02, Proc. of the Asian Control Conf., (Singapore)*, pp. 614-617, 2002.
[17] A.K. Goel., S. C. Saxena and S. Bhanot, "Fast learning algorithm for training feedforward neural networks," *Int. J. of Systems Science*, communicated for publication.