RESEARCH ARTICLE OPEN ACCESS

Investigations on Hybrid Learning in ANFIS

C.Loganathan[1] & K.V.Girija[2]
1. (Principal, Maharaja Arts and Science College, Coimbatore – 641 407, Tamilnadu, India)
2. (Dept of Mathematics, Hindustan Institute of Technology, Coimbatore – 641032, Tamilnadu, India)

ABSTRACT

Neural networks have attractiveness to several researchers due to their great closeness to the structure of the brain, their characteristics not shared by many traditional systems. An Artificial Neural Network (ANN) is a network of interconnected artificial processing elements (called neurons) that co-operate with one another in order to solve specific issues. ANNs are inspired by the structure and functional aspects of biological nervous systems.

Neural networks, which recognize patterns and adopt themselves to cope with changing environments. Fuzzy inference system incorporates human knowledge and performs inferencing and decision making. The integration of these two complementary approaches together with certain derivative free optimization techniques, results in a novel discipline called Neuro Fuzzy. In Neuro fuzzy development a specific approach is called Adaptive Neuro Fuzzy Inference System (ANFIS), which has shown significant results in modeling nonlinear functions.

The basic idea behind the paper is to design a system that uses a fuzzy system to represent knowledge in an interpretable manner and have the learning ability derived from a Runge-Kutta learning method (RKLM) to adjust its membership functions and parameters in order to enhance the system performance. The problem of finding appropriate membership functions and fuzzy rules is often a tiring process of trial and error. It requires users to understand the data before training, which is usually difficult to achieve when the database is relatively large. To overcome these problems, a hybrid of Back Propagation Neural network (BPN) and RKLM can combine the advantages of two systems and avoid their disadvantages.

Keywords: ANFIS, Back propagation learning ,Gradient Descent method, membership function, Runge-Kutta Algorithm,

I. INTRODUCTION

A methodology that can generate the optimal coefficients of a numerical method with the use of an artificial neural network is presented here. Adaptive Neuro Fuzzy Inference System (ANFIS) is used for system identification based on the available data. The main aim of this work is to determine appropriate neural network architecture for training the ANFIS structure in order to adjust the parameters of learning method from a given set of input and output data. The training algorithms used in this work are Back Propagation, gradient descent learning algorithm and Runge-Kutta Learning Algorithm (RKLM). The experiments are carried out by combining the training algorithms with ANFIS and the training error results are measured for each combination [1].

The results showed that ANFIS combined with RKLM method provides better results than other two methods. The method of least squares is about estimating parameters by minimizing the squared difference between observed data and desired data [2]. Gradient Descent Learning for learning ranking functions. The back propagation algorithm uses supervised learning, which means that it provides the algorithm with examples of the inputs and outputs it wants the network to compute, and then the error (difference between actual and expected results) is calculated [3,4].

II. HYBRID FUZZY INFERENCE SYSTEM

Neuro fuzzy hybrid system is a learning mechanism that utilizes the training and learning neural networks to find parameters of a fuzzy system based on the systems created by the mathematical model. A daptive learning is the important characteristics of neural networks [5]. The new model establishes the learning power of neural networks into the fuzzy logic systems. Heuristic fuzzy logic rules and input-output fuzzy membership operations can be optimally tuned from training examples by a hybrid learning method composed of two phases: the phase of rule generation from data, and the phase of rule tuning by using the back-propagation learning with RKLM for a neural fuzzy system [6].

2.1 Adaptive Neuro Fuzzy Inference System (ANFIS)

Fuzzy Inference System(FIS) has a clear disadvantage in that it lacks successful learning mechanisms. The excellent property of ANFIS is that it compensates for the disadvantage of FIS with the learning mechanism of NN. The architecture and learning procedure underlying ANFIS is offered, which is a fuzzy inference system executed in the framework of adaptive networks. By using a hybrid learning procedure, the proposed ANFIS can construct an input-output mapping based on both human knowledge (in the form of fuzzy if-then rules) and stipulated input-output data pairs [7]. The building of ANFIS is very suitable for the fuzzy experience and information [8]

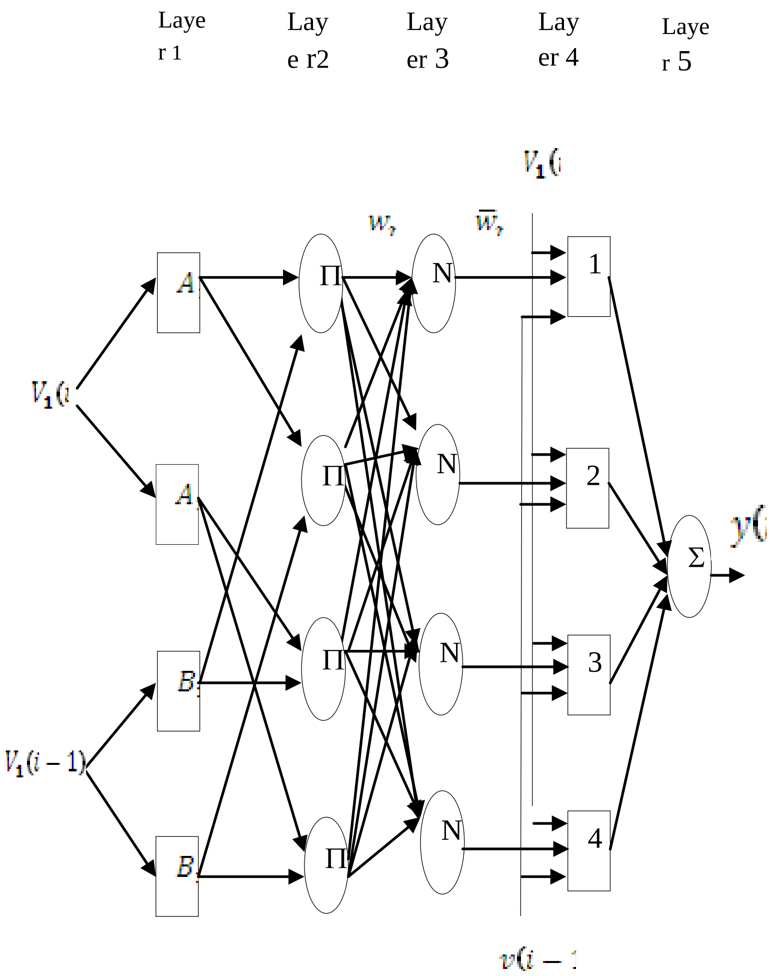

Fig 2.1 The architecture of ANFIS for a two-input

FIS has two inputs $v_1(i)$ and $v_1(i-1)$ as feature vector and one output y. For a first-order FIS, a common rule set with four fuzzy if-then rules is as follows:

$$Rule\ 1: If\ v_1(i) is\ A_1\ and\ v_1(i-1) is\ B_1, then\ y_1 = x_1 v_1(i) + \beta_1 v_1(i-1) + \delta_1$$
$$Rule\ 2: If\ v_1(i) is\ A_1\ and\ v_1(i-1) is\ B_2, then\ y_2 = x_2 v_1(i) + \beta_2 v_1(i-1) + \delta_2$$
$$Rule\ 3: If\ v_1(i) is\ A_2\ and\ v_1(i-1) is\ B_1, then\ y_3 = x_3 v_1(i) + \beta_3 v_1(i-1) + \delta_3$$
$$Rule\ 4: If\ v_1(i) is\ A_2\ and\ v_1(i-1) is\ B_2, then\ y_4 = x_4 v_1(i) + \beta_4 v_1(i-1) + \delta_4$$

where $v_1(i)$ or $v_1(i-1)$ is the input of m^{th} node and i is the length of the sequence vector; A_1, A_2, B_1 and B_2 are linguistic variables (such as "big'' or "small'') associated with this node; α_1, β_1 and δ_1 are the parameters of the initial node, α_2, β_2 and δ_2 of the subsequent node, α_3, β_3 and δ_3 of the third node and α_4, β_4 and δ_4 of the forth node.

First layer: Node n in this layer is indicating as square nodes (Parameters in this layer are changeable). Here $O_{1,n}$ is denoted as the output of The n th node in layer 1

$$O_{1,n} = \mu_{A_{s1}}(v_1(i)), \qquad s_1 = 1,2, \qquad n = 1,2, \tag{2.1}$$

$$O_{1,n} = \mu_{B_{s1}}(v_1(i-1)), \qquad s_2 = 1,2, \qquad n = 3,4, \tag{2.2}$$

where s_1 is defined as s_1^{th} MF of the input $v_1(i)$ and s_2 as s_2^{th} MFs of the input $v_1(i-1)$. It is said that $O_{1,n}$ is the membership grade of a fuzzy set $A (= A_1, A_2, B_1\ or\ B_2)$. It specifies the degree to which the given input $v_1(i)$ (or $v_1(i-1)$) satisfies the quantifier A. Here the MF for A can be any method of parameterized MF, such as the function of the product by two sigmoid functions:

$$\pi_A(v_1) = \frac{1}{\left(1 + e^{-a_1(v_1 - c_1)}\right)\left(1 + e^{-a_2(v_1 - c_2)}\right)} \tag{2.3}$$

where parameters a_1, a_2, c_1 and c_2 are choosing the shape of two sigmoid functions. Parameters in this layer are referred to as premise parameters.

Second layer: The nodes in this layer are labeled by Π in Figure 2.1. The outputs are the development of an inputs

$$O_{2,n} = w_n = \mu_{A_{s1}(v_1(i))}\mu_{B_{s2}}(v_1(i-1)), \qquad s_1 = 1,2, \qquad s_2 = 1,2, \tag{2.4}$$
$$n = 1,2,3,4.$$

The output of every node n output represents the firing strength of a rule. Usually any other T-norm operators which execute fuzzy AND can be used as the node function in this layer.

Third layer: The nodes in this layer are labeled with N. The n^{th} node evaluate the ratio of n^{th} rule's firing strength to the summation of all rules' firing strengths:

$$O_{3,n} = \bar{w}_n = \frac{w_n}{\sum_{n=1}^{4} w_n}, \quad n = 1,2,3,4. \tag{2.5}$$

The outputs of this layer are named as normalized firing strengths.

Fourth layer: Every node in this layer is called an adaptive node. The outputs are

where $\alpha_n, \beta_n, \delta_n$ are parameter of the nodes. Parameters in this layer are called following parameters.

$$O_{4,n} = \bar{w}_n y_n = \bar{w}_n(\alpha_n v_1(i) + \beta_n v_1(i-1) + \delta_n), \quad n = 1,2,3,4. \tag{2.6}$$

Fifth layer: Every node in this layer is a fixed node labeled Σ. Its total output of the summation of all inputs is

$$y = O_{s,1} = \sum_{n=1}^{4} \bar{w}_n y_n = \frac{\sum_{n=1}^{4} w_n y_n}{\sum_{n=1}^{4} w_n} \tag{2.7}$$

By using the chain rule in the differential, the alter in the parameter q for the input $v_1(i)$ is

$$\Delta q(v_1(i)) = -\lambda \frac{\partial E}{\partial q} = -\lambda \frac{\partial E}{\partial y} \frac{\partial y}{\partial \bar{w}_n} \frac{\partial \bar{w}_n}{\partial w_n} \frac{\partial w_n}{\partial \mu_{A_{s1}}(v_1(i))} \frac{\partial \mu_{A_{s1}}(v_1(i))}{\partial q}$$

$$= \lambda.(x-y).y_n.\frac{\bar{w}_n.(1-\bar{w}_n)}{w_n}.\frac{w_n}{\mu_{A_{s1}}(v_1(i))}.\frac{\partial \mu_{A_{s1}}(v_1(i))}{\partial q} \tag{2.8}$$

$$= \frac{\lambda.y_n.\bar{w}_n.(x-y).(1-\bar{w}_n)}{\mu_{A_{s1}}(v_1(i))}.\frac{\partial \mu_{A_{s1}}(v_1(i))}{\partial q},$$

$s_1 = 1,2, \quad n = 1,2,$

where λ is a learning rate. As the same, the change in the parameter q for the input $v_i(i-1)$ is

$$\Delta q(v_1(i-1)) = \frac{\lambda.y_n.\bar{w}_n.(x-y).(1-\bar{w}_n)}{\mu_{B_{s2}}(v_1(i-1))}.\frac{\partial \mu_{B_{s2}}(v_1(i-1))}{\partial q} \tag{2.9}$$

$s_1 = 1,2, \quad n = 3,4,$

where q is the parameter a_1, a_2, c_1 and c_2 which decide the shape of two sigmoid functions.

The changes in the parameter q for the input $v_1(i)$ and $v_1(i-1)$ is

$$\Delta q = \Delta q(v_1(i)) + \Delta q(v_1(i-1))$$

$$= \frac{\lambda.y_n.\bar{w}_n.(x-y).(1-\bar{w}_n)}{\mu_{A_{s1}}(v_1(i))}.\frac{\partial \mu_{A_{s1}}(v_1(i))}{\partial q} + \frac{\lambda.y_n.\bar{w}_n.(x-y).(1-\bar{w}_n)}{\mu_{B_{s2}}(v_1(i))}.\frac{\partial \mu_{B_{s2}}(v_1(i))}{\partial q}, s_1 = 1,2 \quad s_2 = 1,2, \quad n = 1,2,3,4. \tag{2.10}$$

Table 2.1 Two passes in the hybrid learning procedure for ANFIS

	Forward pass	Backward pass
Premise parameters	Fixed	Gradient descent
Consequent parameters	Least-square method	Fixed
Hybrid BPN-RKLM	BPN	RKLM

The training procedure consists of the following steps:

i. Propagate all patterns from the training data and determine the consequent parameters by iterative LSE. The antecedent parameters remain fixed.
ii. The proposed Hybrid BPN-RKLM perform function forward pass by using BPN and backward pass by RKLM [9].

2.2 Learning and reasoning based on the model of ANFIS

ANFIS uses the established parameter learning algorithm in neural networks—the back-propagation (BP) algorithm and BPN-RKLM. It adjusts the shape parameters of MF in FIS by learning from a set of given input and output data[10].

The algorithm of ANFIS is very easy. It offers learning techniques which can achieve corresponding messages (fuzzy rules) from data for fuzzy modeling. This learning method is very similar to the learning algorithm of NN. It successfully calculates the best parameter for MF. ANFIS is a modeling technique based on the given data. It is the best measuring criterion for whether or not the results of FIS model can simulate the data well [11].

2.3 Adjustment of structure and parameters for FIS

The structure in Figure 2.1 is same to the structure of NN. It first maps inputs with the MFs of inputs and parameters. Then it maps the data of input space to output space with the MFs of output variables and parameters.

The parameters finding the shapes of MFs are adjusted and altered through a learning procedure. The adjustment of these parameters is accomplished by a gradient vector. It appraises a set of particular parameters how FIS meets the data of inputs and outputs. Once this gradient is developed, the system can adjust these parameters to diminish the error between it and the expected system by optimal algorithm: (This error is usually defined as the square of difference between the outputs and targets.)[12].

2.4 Hybrid Learning Algorithm

Because the basic learning algorithm, backpropagation method, which presented previously, is based on the RKLM, a hybrid learning algorithm is used to speed up the learning process. Let ***V*** be a matrix that contains one row for every pattern of the training set. For a performance of order 2, **V** can be defined as

$$V = (v_1(i), v_1(i-1)). \qquad (2.11)$$

Let **Y** be the vector of the target output values from the training data and let

$$A = \left(x(i), \beta(i), \delta(i)\right) \qquad (2.12)$$

be the vector of all the consequent parameters of all the rules for a nonlinear passive dynamic of order 2. The consequent parameters are determined by

$$VA = Y \qquad (2.13)$$

A Least Squares Estimate (LSE) of A, A* is sought to minimize the squared error $\|VA - Y\|$. The most well-known formula for A* uses the pseudo-inverse of A

$$A^* = (V^TV)^{-1}V^TY \qquad (2.14)$$

where V^T is the transposition of V, and $(V^TV)^{-1}V^T$ is the pseudo-inverse of V if (V^TV) is non-singular. The sequential formulae of LSE are very efficient and are utilized for training ANFIS. Let $v(i)^T$ be the *i*th row of matrix V and let $y(i)^T$ be the *i*th element of vector Y, then *A* can be evaluated iteratively by using following formulae:

$$A(i+1) = A(i) + S(i+1).v(i+1).[(y(i+1)]^T - v(i+1).A(i)) \qquad (2.15)$$

$$S(i+1) = S(i) - \frac{S(i).v(i+1).v^T(i+1).S(i)}{1 + v(i+1).S(i).v(i+1)}, i = 0,1,\dots,MN-1 \qquad (2.16)$$

where **S** *(i)* is often called the covariance matrix, M is the number of neurons and N is the number of nodes [13].

Now the RKLM is combined with the least square estimator method to update the parameters of MFs in an adaptive inference system. Each epoch in the hybrid learning algorithm includes a forward pass and a backward pass. In the forward pass, the input data and functions go forward to calculate each node output. The function still goes forward until the error measure is calculated. In the backward pass, the error rates propagate from output end toward the input end, and the parameters are updated by the RKLM[14].

Now the gradient technique is joined with the Runge-Kutta method to report the parameters of MFs in an adaptive inference system. Every epoch in the hybrid learning algorithm contains a forward pass and a backward pass. In the forward pass, the input data and functional operation go forward to evaluate each node output. The functional process still goes forward until the error measure is calculated. In the backward pass, the error rates propagate from output end toward the input end, and the parameters are updated by the gradient method [15]. Table 2.1 summarizes the activities in each pass.

2.5 Training ANFIS Model

ANFIS can be trained to study from given data. As it can study from the ANFIS architecture, in order to configure an ANFIS model for a particular problem, need to denote the fuzzy rules and the activation functions (i.e. membership functions) of fuzzification neurons. The fuzzy rules, can denote the antecedent fuzzy sets once it knows the specific problem domain; while for the sequence of the fuzzy rules, the parameters (e.g., $k_{i0}, k_{i1}, k_{i2}, \dots\dots, k_{1n}$) are constructed and adjusted by learning algorithm in the training process[13,16].

For ANFIS models, the mostly used activation function is the so-called bell-shaped function, described as:

$$y = \frac{1}{1 + \left[\left(x - \frac{s}{r}\right)^2\right]^t} \qquad (2.17)$$

where *r*, *s* and *t* are parameters that respectively control the slope, centre and width of the bell-shaped function. And in the training operation, these parameters can be specified and adjusted by the learning algorithm [17].

Specifically, an ANFIS uses a 'hybrid' learning (training) algorithm. This learning algorithm combines the so-called least-squares estimator and the gradient descent method finally with Runge-Kutta learning method. In the beginning, initial bell-shaped functions with particular parameters are assigned to each fuzzification neuron. The function centre of the neurons connected to input x_i are set so that the domain of x_i is separated uniformly, and the function widths and slopes are set to allow sufficient overlapping(s) of the respective functions.

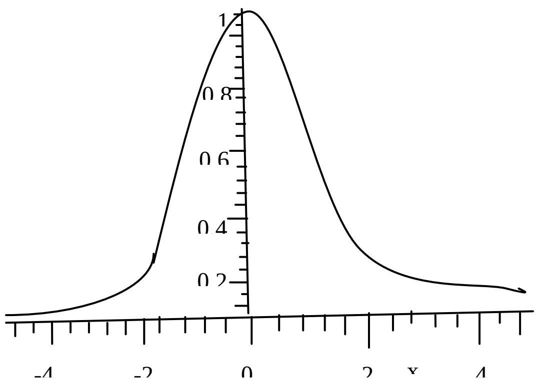

Fig 2.2 A bell-shaped function with r = t = 1 and s = 0.

During the training process, the training dataset is offered to the ANFIS cyclically. Every cycle through all the training examples is called an epoch. In the ANFIS learning algorithm, each epoch comprises of a forward pass and a backward pass. The function of the forward pass is to form and adjust the consequent parameters, while that of the backward pass is to alter the parameters of the activation functions [18].

2.6 Hybrid Learning Algorithm in ANFIS with BPN

Additionally, for the back-propagation learning the main part concerns to how to recursively obtain a gradient vector in which each element is defined as the derivative of an error measure with respect to a parameter. Considering the output function of node *i* in layer *l*.

$$x_{l,i} = f_{l,i}\left(x_{l-1,1}, \ldots, x_{l-1,N(l-1)}, \alpha, \beta, \gamma, \ldots\right) \quad (2.18)$$

where $\alpha, \beta, \gamma, \ldots$ are the parameters of this node.

Hence, the sum of the squared error defined for a set of P entries, is defined as:

$$E_p = \sum_{k-1}^{N(L)} \left(d_k - x_{l,k}\right)^2 \quad (2.19)$$

Where d_k is the desired output vector and $X_{l,k}$ both for the k^{th} of the p^{th} desired output vector. The basic concept in calculating the gradient vector is to pass from derivative information starting from the output layer and going backward layer by layer until the input layer is reached. The error is defined as follows.

$$\epsilon_{l,i} = \frac{\partial E_p}{\partial x_{l,i}} \rightarrow \epsilon_{l,i} = -2\left(d_i - x_{L,i}\right) \quad (2.20)$$

If α is a parameter of the i^{th} node at layer *l*. thus, it is obtained the derivative of the overall error measure E with respect to α is shown below.

$$\frac{\partial E}{\partial \alpha} = \sum \frac{\partial E}{\partial \alpha} \quad (2.21)$$

Thus the generic parameter α is shown below

$$\Delta\alpha = -\eta \frac{\partial E}{\partial \alpha} \quad (2.22)$$

Where η is the learning rate. So, parameter α is defined as

$$\alpha_{new} = \alpha_{old} - \frac{(-\eta)\partial E}{\partial \alpha} \quad (2.23)$$

For hybrid learning algorithm, each epoch have two pass, one is forward pass and another one is backward pass. Then (2.19) is applied to find out the derivative of those error measures. Thus, the error is obtained. In the backward pass, these errors propagate from the output end towards the input end. The gradient vector is found for each training data entry. At the end of the backward pass for all training data pairs, the input parameters are updated by steepest descent method as given by equation (2.23) [19].

2.7 Hybrid learning algorithm in ANFIS with RKLM

For the subsequent development of ANFIS approach, a number of techniques have been proposed for learning rules and for obtaining an optimal set of rules. In this work, an application of RKLM, which is essential for classifying the behavior of datasets, is used for learning in ANFIS network for Training data. The output of ANFIS is given as an input to the ANFIS RKLM. The inputs given to ANFIS RKLM are number of membership function, type of membership function for each input, type of membership function, output, training data and train target [20].

III. EXPERIMENTAL RESULTS

The experiment was carried out using synthetic dataset which is generated randomly 100 samples. From the generated data 70% of the data are used for training and remaining used for testing. The experiment was done using MATLAB 7 under windows environment [21]. Hybrid ANFIS is relatively fast to convergence due to its hybrid learning strategy and its easy interpretation but proposed hybrid learning boost the convergence speed of the standard hybrid learning with the help of Runge-Kutta learning model. This provides a better understanding of the data and gives the researchers a clear explanation of how the convergence achieved[22].

Table 3.1 Accuracy for ANFIS methods

Hybrid Methods	Classification Accuracy (%)
BPN	75
Hybrid (BPN and LSE)	82
Hybrid (BPN and RKLM)	95

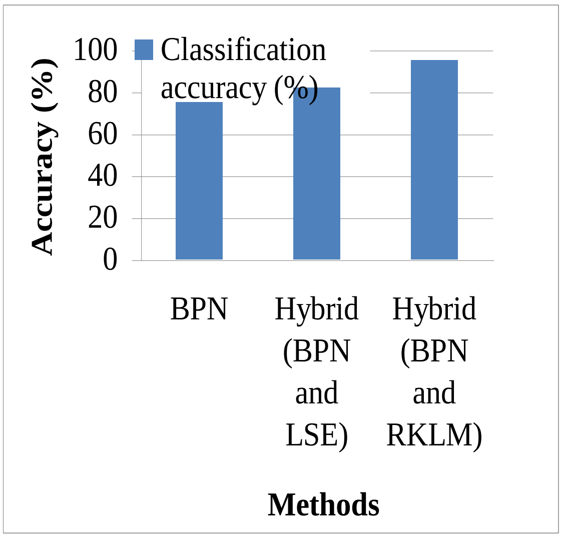

Fig 3.1 Accuracy for Hybrid BPN and RKLM

Table 3.1 and Figure 3.1 show the accuracy of proposed Hybrid BPN and RKLM. Thus the accuracy of proposed method is higher when compare ANFIS with BP rid BP

Table 3.2 Execution time for ANFIS methods

Hybrid Methods	**Time (Seconds) Execution**
BPN	50
Hybrid (BPN and LSE)	38
Hybrid (BPN and RKLM)	21

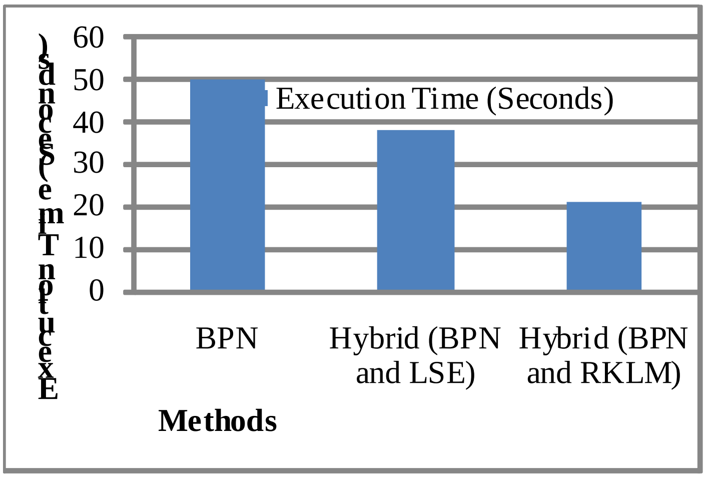

Fig 3.2 Execution Time for Hybrid BPN and RKLM

The table 3.2 and Figure 3.2 show the execution time for proposed method. The proposed method of Hybrid ANFIS has less execution time when compared with others.

IV. SUMMARY

ANFIS algorithm is the most popular algorithm used in Neuro Fuzzy models. It shows that the new method is more efficient than the classical methods and thus proves the capability of the constructed neural network along with Runge-Kutta methods. The basic theory for Runge-Kutta methods and Artificial Neural Networks are given respectively. The work is in progress in the direction of improving the estimation performance through the use of the combined approaches of ANFIS and RKLM.

REFERENCES

[1]. McGraw -Hill, New York, Mitchel.T, *Machine Learning*, 1997.

[2]. Niegemann, Jens, Richard Diehl and Kurt Busch, *Efficient low-storage Runge–Kutta schemes with optimized stability regions, Journal of Computational Physics 231(2)*, 2012, 364-372.

[3]. Jain. L, Johnson.R and Van Rooij.A, *Neural Network Training Using Genetic Algorithms, World Scientific*,1996

[4]. Lok and Michael R. Lyu, *A hybrid particle swarm optimization back-propagation algorithm for feedforward neural network training, Applied Mathematics and Computation, 185(2)*, 2007, 1026-1037.

[5]. Parveen Sehgal, Sangeeta Gupta and Dharminder Kumar, *Minimization of Error in Training a Neural Network Using Gradient Descent Method, International Journal of Technical Research, Vol. 1, Issue1*, 2012, 10-12.

[6]. Bishop.C.M, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, 1995.

[7]. Zadeh. L, *Fuzzy algorithms, Information and Control*, 12, 1968, 94–102.

[8]. Akkoç. Soner, *An empirical comparison of conventional techniques, neural networks and the three stage hybrid Adaptive Neuro Fuzzy Inference System (ANFIS) model for credit scoring analysis: The case of Turkish credit card data. European Journal of Operational Research, 222(1)*, 2012, 168-178.

[9]. Ixaru.L.Gr, *Runge–Kutta method with equation dependent coefficients, Computer*

Physics Communications, 183(1), 2012, 63-69.

[10]. Loganathan.C and Girija K. V, *Hybrid Learning For Adaptive Neuro Fuzzy Inference System, International Journal of Engineering and Science Vol.2, Issue 11,* 2013, 06-13.

[11]. Mallot and Hanspeter, *An Artificial Neural Networks, Computational Neuroscience, Springer International Publishing* 2013, 83-112.

[12]. Robert Hecht-Nielsen, *Theory of the backpropagation neural network in Neural Network, International Joint Conference on Neural Networks (IJCNN), vol I*, 1989, 593-605.

[13]. Jyh-Shing Roger Jang and Chuen-Tsai Sun, Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence, Prentice-Hall Inc., 1996.

[14]. Angelos A. Anastassi, *Constructing Runge–Kutta methods with the use of artificial neural networks, Neural Computing and Applications*, 2011, 1-8.

[15]. M. Jalali Varnamkhasti, *A hybrid of adaptive neuro-fuzzy inference system and genetic algorithm, Journal of Intelligent and Fuzzy Systems, 25(3)*, 2013, 793-796.

[16]. Lipo Wang and Xiuju Fu, *Artificial neural networks*, John Wiley and Sons Inc., 2008.

[17]. Jang.J.S.R, *Neuro-Fuzzy Modeling Architectures*, *Analyses and Applications*, *Ph.D thesis*, University of California, Berkeley, 1992.

[18]. Jang.J.S.R and Sun.C.T, *Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence,* Prentice-Hall, NJ, USA 1997.

[19]. Jovanovic, Branimir B, Irini S. Reljin and Branimir D. Reljin, Modified ANFIS architecture-improving efficiency of ANFIS technique in Neural Network Applications in Electrical Engineering, *Proc.on 7th Seminar on NEUREL 2004, 215-220.*

[20]. Nazari Ali and Shadi Riahi, *Experimental investigations and ANFIS prediction of water absorption of geopolymers produced by waste ashes, Journal of Non-Crystalline Solids, 358(1*), 2012, 40-46.

[21]. Nguyen. H.S, Szczuka. M.S and Slezak.D, *Neural networks design: Rough set approach to continuous data, Proceedings of the First European Symposium on Principles of Data Mining and Knowledge Discovery, Springer-Verlag, London, UK,* 1997, 359–366.

[22]. Kitano. H, *Designing neural networks using genetic algorithms with graph generation system, Complex Systems*, 1990, 461–476.