# Lyapunov Theory-based Fuzzy Neural Network With MOGA And Its Application To Nonlinear Time Series Prediction

[1]SENG KAH PHOOI, [2]KAI MING TSE, [3]ZHIHONG MAN, [4]H.R.WU
[1]School of Engineering, Monash University (Malaysia), Bandar Sunway, 46150 PJ, MALAYSIA
[2]School of Microelectronics, Griffith University, Kessels Rd, Nathan QLD 4111,AUSTRALIA
[3]School of Computer Engineering, Nanyang Technological University, SINGAPORE
[4] School of CS & SE, Monash University (Clayton) VIC 3168, AUSTRALIA
kp_seng@yahoo.com, ktse@me.gu.edu.au

*Abstract:* - In this paper the authors present a Lyapunov theory-based FNN with MOGA for application to nonlinear time series prediction. The architecture employs FNN structure and the learning algorithms are the combination of MOGA and LAF. The application considered is the nonlinear time series prediction. Simulation results are obtained using the MATLAB for the nonlinear sunspot data prediction. The work not only demonstrates the advantage of the neurofuzzy approach but it also highlights the advantages of the fusion of MOGA and LAF.

*Key-Words:* Time Series Prediction, Neural Network, Fuzzy Logic, Lyapunov stability theory

## 1 Introduction

Time series prediction is a very important practical application with a diverse range of applications including economic and business planning, inventory and production control, weather forecasting, signal processing and control [1]. As a result, there has been considerable interest in the application of intelligent technologies such as neural networks (NNs) and fuzzy logic [2]-[3]. More recently, these two computationally intelligent techniques have been viewed as complementary leading to developments in fusing the two technologies [4], with a number of several successful neurofuzzy systems reported in the literature [5]. Such work has demonstrated the superior prediction capabilities of a fuzzy neural network as compared to the conventional neural network approach [5]-[6].

In this paper the authors employ the fuzzy neural network (FNN) for nonlinear time series prediction. The merit of this paper is that a combination of the new Lyapunov theory-base filtering algorithm (LAF) [7] and multi-objective genetic algorithm (MOGA) [10] are used to train the FNN. The proposed scheme provides not only the advantages of fuzzy logic and NN but it also offer additional advantages those are offered by the LAF and MOGA. In the consequence part, the weights of FNN are adaptively adjusted by the LAF so that the error convergences to the convergence to zero asymptotically. The input signals' stochastic properties are not required. The stability of the dynamic prediction error is guaranteed by Lyapunov theory [8]. It has fast convergence properties and less computation complexity [10]. The MOGA is used to tune the parameters of the membership functions (MBFs) in the premise part. Most real world problems require the simultaneous optimisation of multiple criteria/objectives. In this case, MOGA can provide the solution to these problems. In our case, 2 types of error defined in later section are the multiple criteria to be solved by MOGA.The theoretical prediction mechanism of the proposed predictor is further confirmed by simulation examples for real world data.

The paper is organized as follow: section 2 briefly describes the main features of the proposed FNN. Section 3 presents the LAF algorithm. Section 4 describes the MOGA. The prediction results are presented in section 5. The finally section 6 concludes the paper with a discussion of the significance of the results.

## 2 Fuzzy Neural Network

The fuzzy logic inference system can be implemented as a five-layer NN (Fig. 1). This type of architecture is the most common among neural fuzzy inference systems. Given the training input data $x_n$, $n=1,2,\dots N$, and the desired output $d_m$, $m=1,2,\dots M$, the inference rules of simplified fuzzy reasoning [ ]-[] can be established by experts. The rule base contains the following form:

$$R^i:\ \text{IF } x_1 \text{ is } A^i_1 \text{ and } \dots x_N \text{ is } A^i_N \quad \text{THEN } y_1 \text{ is } w^i_1 \text{ and } \dots y_M \text{ is } w^i_M, \qquad (2.1)$$

Where $i$ is a rule number, the $A^i_N$ 's are MBF's of the antecedent part and $w^i_M$'s are real numbers of the consequent part.

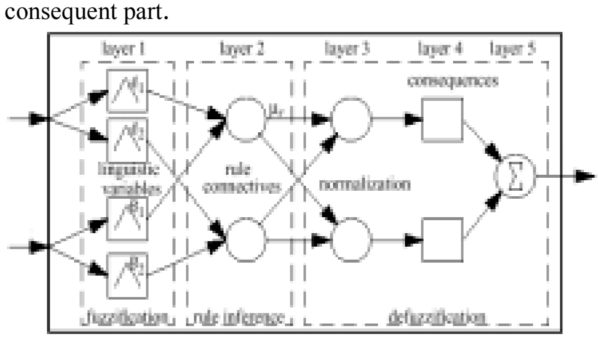


Fig. 1 The configuration of the FNN

The operation of the this system can be described layer by layer as follows:

***Layer 1: Fuzzification***

This layer consists of linguistic variables. The crisp inputs $x_n$, $n=1,2,\dots N$ are fuzzified by using MBFs of the linguistic variables $A^i{}_N$. Usually, triangular, trapezoid, Gaussian or bell-shaped membership functions are used.

***Layer 2. Rule nodes***

The second layer contains one node per each fuzzy if-then rule. Each rule node performs connective operation between rule antecedents (if-part). Usually, the minimum or the dot product is used as intersection **AN**D. The union **OR** is usually done using maximum operation. In our example case the firing strengths $\mu_i$ of the fuzzy rules are computed according to

$$\mu_i = A^i{}_1(x_1)\cdot A^i{}_2(x_2)\cdot .... A^i{}_N(x_N) \qquad (2.2)$$

***Layers 3-5: Normalization, Consequence & Summation***

In the third layer, the firing strengths of the fuzzy rules are normalized. Layer 4 is related to consequent fuzzy labels $w^i{}_M$, which are singletons in our case. The values of the singletons are multiplied by normalized firing strength. The final layer computes the overall output as the summation of the incoming signals. Therefore the output $y_m$ of the fuzzy reasoning can be represented by the following equation:

$$y_m = \frac{\sum_i^Q \mu_i w^i{}_m}{\sum_i^Q \mu_i} \qquad (2.3)$$

$$Y = [\, y_1, y_2, \ldots, y_M] \qquad (2.4)$$

After the fuzzy logic rules and network structure have been established, the learning algorithm can then applied to adjust the parameters of the MBFs in the premise part and the weights in the consequence parts. In this paper, we proposed to use LAF algorithm to adaptively adjust the weights in the consequence parts and MOGA to tune the parameters of MBFs.

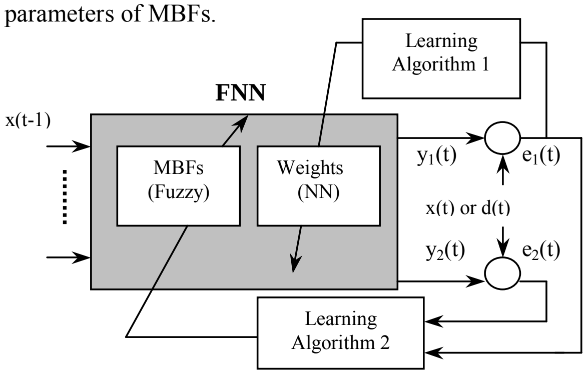


Fig. 2: The block diagram of FNN with MOGA+LAF

Fig. 2 illustrates the overall process of the proposed scheme for the prediction problem. The layer 5 consists of 1 summation node or 1 output, $y_1(t)$ which is defined as

$$y_1(t) = \frac{\sum_i^Q \mu_i w^i{}_1(t)}{\sum_i^Q \mu_i} \qquad (2.5)$$

$y_2(t)$ is not another output node of FNN as shown in Fig. 2 and it is only computed using (2.6)

$$y_2(t) = \frac{\sum_i^Q \mu_i w^i{}_1(t-1)}{\sum_i^Q \mu_i} \qquad (2.6)$$

# 3 Learning Algorithm 1- LAF

In the consequence part, the training algorithm used is LAF that adaptively adjusts the weights of FNN. The weights in the consequence part are updated as follow:

$$w^i{}_m(t) = w^i{}_m(t-1) + g^i{}_m(t)\alpha_m(t) \qquad (3.1)$$

where $g^i{}_m(t)$ is the adaptation gain and $\alpha_m(k)$ is defined as

$$\alpha_m(t) = d_m(t) - \frac{\sum_i \mu_i w^i{}_m(t-1)}{\sum_i \mu_i} \qquad (3.2)$$

The adaptation gain is given by (3.3)

$$g^i{}_m(t) = \frac{\mu_i(t)}{\|U(t)\|^2}\left(1 - k\frac{|e_m(t-1)|}{\alpha_m(t)}\right) \qquad (3.3)$$

where $0 < k \le 1$. $U(t) = [\,\mu_1, \mu_2, \ldots \mu_Q]$

It is noticeable that the values of $U(t)$ and $\alpha_m$ in (3.3) may be zero and rise singularities problem. Therefore the adaptation gain may be modified as (3.4)

$$g^i{}_m(t) = \frac{\mu_i(t)}{\|U(t)\|^2 + \lambda_1}\left(1 - k\frac{|e_m(t-1)|}{\alpha_m(t)+\lambda_2}\right) \qquad (3.4)$$

where $0 < k \le 1$, and $\lambda_1$, $\lambda_2$ are small positive numbers.

# 4 Learning Algorithm 2 - MOGA

The MOGA is used to tune the parameters of the membership functions (MBFs) in the premise part. Without the need of linearly combining multiple attributes into a composite scalar objective function, evolutionary algorithms incorporate the concept of Pareto's optimality or modified selection schemes to evolve a family of solutions along the tradeoff surface. In this paper we employ the weighted sum-based optimization method.

## 4.1 Weighted Sum Based Optimization

In a weighted sum-based optimization, multiobjective function $F=(f_1,\ldots,f_2)$ is transformed into $F_\omega = \sum_{i=1}^{k} \omega_i . f_i$ so that single objective opimization methods can be used. Preferences are used for specifying weights. With reference to Fig. 2, two fitness $F_1$ and $F_2$ can be evaluated from $e_1(t)$ and $e_2(t)$, $\forall t$. Thus, these two objective functions is transformed in an overall fitness function $F = \omega_1.F_1 + \omega_2.F_2$, where $\omega_1 + \omega_2 = 1$. In our experiment, we choose $\omega_1 = 0.3$ and $\omega_2 = 0.7$.

## 4.2 Computational Algorithms

1. Initialisation: Training data is clustered to generated 9 centroids based on which the Gaussian MBFs (mean and variance) are evaluated. 80 potential candidates P(t) are created by varying $\pm$20% of the MBFs.
2. Evaluate the overall fitness F. Select candidates proportional to their fitness relative to the others in P(t) using the Stochastic Universal Sampling technieque.
3. Applying genetic operators, whole arithmetic crossover, mutation, and adaptation with the best candidate by adding a perturbation to the relative best-fit candidate, to reproduce new candidates.

4. Combine all new candidates with the P(t) to form the new population for the next generation.
5. Repeat step 2, 3, 4 until termination condition is satisfied.

# 5 Simulation

Simulations have been done for a one-step ahead prediction of the Sunspot data. Sunspot data is used as a benchmark for many years by researchers. Data file of the Sunspot times series is download from [9]. It consists the sunspot data from the year 1700 to 1999 (300 Samples). Fig. 3 shows the plot of the sunspot time series. Fig. 4 shows the Mean Squared Error of $e_2(t)$ giving MSE=0.0159 at the 30$^{th}$ generation. The overall simulation is successful and fig. 3 show no distinct difference between the $y_2(t)$ and d(t). All work is done in Matlab.

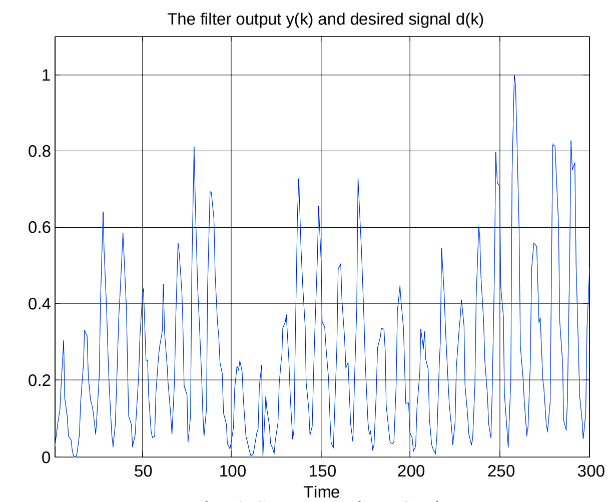


Fig. 3 Sunspot Time Series

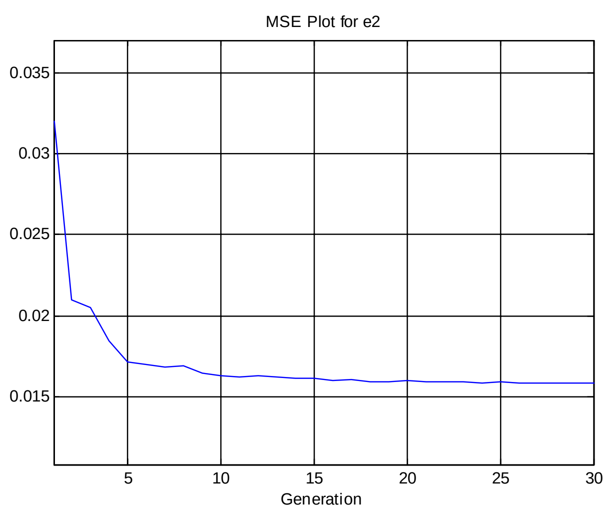


Fig. 4 MSE of $e_2(t)$

# 6 Conclusion

This paper has presented a new approach in designing a FNN with MOGA and LAF techniques.The previous section clearly demonstrate the performance of the proposed FNN for the prediction of nonlinear time series. The FNN approach has also the added advantage of NN and Fuzzy logic. LAF has provided the fast error convergence to the training of FNN. On the other hand, MOGA has added advantage of global optimization to the FNN training based on two criteria/objectives. The results have emphasized the benefits of the fusion of fuzzy and NN technologies as well as the advantages of the fusion of the new LAF and MOGA or GA. This increase in transparency of the neurofuzzy approach overcomes the drawback of FNN with gradient techniques and/or GA in the conventional NNs or FNNs. In general the prediction capability (accuracy) of this system is proportional to its granularity (the number of fuzzy sets) in the premise part and the numbers of weights in the consequence part. Future works need to be conducted in this area. Many issues need to be addressed regarding simulations, practical implementations, and the further analysis on the theoretical parts of the proposed scheme.

*References:*


[1] Box, GEP; Jenkins, GM: "Time series analysis: forecasting and control", Oakland CA: Holden-Day, 1976.

[2] Wang, LX; Mendel, JM: "Generating fuzzy rules by learning from examples", IEEE Trans. Systems Man and Cybernetics, Vol 22, No. 6, pp. 1414-1427, 1992.

[3] Moody, J; Darken, C: "Fast learning in networks of locally tuned processing units", Neural Computation, pp. 281-294, Vol. 1, 1989.

[4] Brown, M; Harris, C: "Neurofuzzy adaptive modelling and control", Prentice-Hall, 1994.

[5] Jang, Roger JS: "Predicting chaotic time series with fuzzy IF-THEN rules" Proc. IEEE 2nd Int. Conf. on Fuzzy Systems, pp. 1079-1084 Vol. 2, 1993.

[6] Jang, Roger J S; Sun, C-T: "Neuro-fuzzy modelling and Control" Proc. of the IEEE, pp. 378-406, Vol 83, No 3 March 1995.

[7] Seng Kah Phooi, Zhihong Man, H.R. Wu, "Lyapunov stability-based RBF neural networks for nonlinear adaptive predictive coding", 2nd International Conference on Information, Communications and signal processing, ICICS'99, pp. paper no. 421, 1999.

[8] Slotine, J-J. E. and Li, W. *Applied nonlinear control, Prentice-Hall, Englewood Cliffs, NJ*, 1991.

[9] http://www.astro.oma.be/SIDC/index.html

[10] Deb, K., "Evolutionary Algorithms for Mutli-Criterion Optimization in Engineering Design. *Proceedings of Evolutionary Algorithms in Engineering and Computer Science (EUROGEN-99)*, 1999.