ENHANCED NEURO-FUZZY ARCHITECTURE FOR ELECTRICAL LOAD FORECASTING

Hany Ferdinandoa, Felix Pasila, Henry Kuswanto
Department of Electrical Engineering, Petra Christian University
Siwalankerto 121-131 Surabaya, Phone: +62 31 2983446 Fax: +62 31 8436418
e-mail: hanyf@petra.ac.id

Abstract

Previous researches about electrical load time series data forecasting showed that the result was not satisfying. This paper elaborates the enhanced neuro-fuzzy architecture for the same application. The system uses Gaussian membership function (GMF) for Takagi-Sugeno fuzzy logic system. The training algorithm is Levenberg-Marquardt algorithm to adjust the parameters in order to get better forecasting system than the previous researches. The electrical load was taken from East Java-Bali from September 2005 to August 2007. The architecture uses 4 inputs, 3 outputs with 5 GMFs. The system uses the following parameters: momentum=0.005, gamma=0.0005 and wildness factor=1.001. The MSE for short term forecasting for January to March 2007 is 0.0010, but the long term forecasting for June to August 2007 has MSE 0.0011.

Keywords: *forecasting, LMA, neuro-fuzzy*

Abstrak

Hasil penelitian sebelumnya tentang prakiraan data beban listrik masih belum memuaskan. Makalah ini menguraikan perbaikan arsitektur neuro-fuzzy untuk aplikasi yang sama. Sistem ini menggunakan fungsi keanggotaan Gaussian untuk sistem logika fuzzy berbasis Takagi-Sugeno. Algoritma Levenberg-Marquardt dipergunakan dalam pelatihan jaringan untuk mengubah parameter, sehingga arsitektur yang baru ini memberikan hasil yang lebih baik. Data beban listrik diambil dari beban listrik Jawa Timur-Bali pada September 2005 sampai dengan Agustus 2007. Arsitektur ini menggunakan 4 masukan, 3 keluaran dengan 5 fungsi keanggotaan. Parameter yang dipergunakan adalah sebagai berikut, momentum=0,005, gamma=0,0005 dan wildness factor=1,001. Nilai MSE untuk prakiraan jangka pendek untuk Januari hingga Maret 2007 adalah 0,0010, sedangkan MSE prakiraan jangka panjang untuk Juni hingga Agustus 2007 adalah 0,0011.

Kata kunci: *LMA, neuro-fuzzy, prakiraan*

1. INTRODUCTION

Previous researches [1]-[6] about electrical load time series data forecasting show that the result was not satisfying. Gustafson-Kessel clustering algorithm shows good result, but it is not satisfying [1]. Forecasting with fuzzy C-means [2] showed that the MSE is not acceptable. For this reason, the same research but with another method, i.e. enhanced Gustafson-Kessel using evolutionary algorithm has evaluated by [3], but still the result is not acceptable. The other researches [4]-[6] also show that the result is not good. Their RMSE are 5.4%. In order to reach RMSE under 5%, the research is continued with different approach.

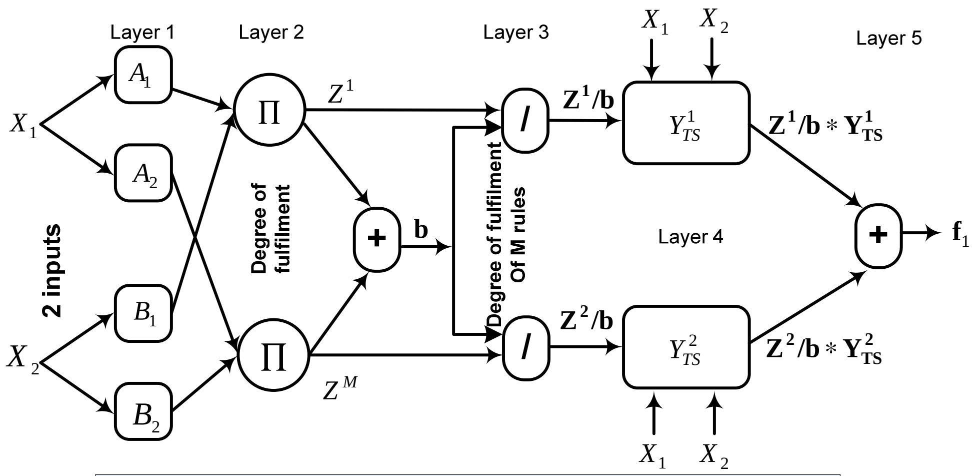
This paper elaborates the implementation of neuro-fuzzy for electrical load time series data forecasting. The proposed method uses the same data as in [1]-[6] so that the methods can be easily compared. According to those researches, the data is from East Java-Bali from September 2005 to August 2007. The neuro-fuzzy architecture will be explored in order to get the optimum architecture. It uses MIMO Takagi-Sugeno type and Levenberg-Marquardt training algorithm to make the training efficient. The architecture of neural network uses feed-forward neural network. The forecasting system uses both short and long time forecasting. Data from

September to December 2006 is used for data training while the rest of the data is for data testing. The goal of this research is to reach the RMSE for LTF under 5%.

2. PROPOSED METHOD
2.1. Neuro-Fuzzy Architecture

Neuro-Fuzzy architecture combines the advantage of neural network and fuzzy logic in order to get better performance than operating them separately. Here, the neuro-fuzzy uses feed-forward neural network with Levenberg-Marquardt algorithm (LMA) training.

The fuzzy part uses MIMO Takagi-Sugeno type with differentiable membership, i.e. Gaussian membership function (GMF), and its to form fuzzy inference and defuzzifier. The output itself is represented as multilayer neural network as shown in Figure 1. The Takagi-Sugeno itself uses the architecture recommended by Palit and Popovic. Figure 2 shows the architecture.



$R1: If\ X_1\ is\ A_1\ and\ X_2\ is\ B_1\ Then\ Y_{TS}^1 = w_0^1 + w_1^1 X_1 + w_2^1 X_2$

$R2: If\ X_1\ is\ A_2\ and\ X_2\ is\ B_2\ Then\ Y_{TS}^2 = w_0^2 + w_1^2 X_1 + w_2^2 X_2$

Figure 1. ANFIS architecture with Takagi-Sugeno type with 2 rules, 2 inputs and 1 output [8]

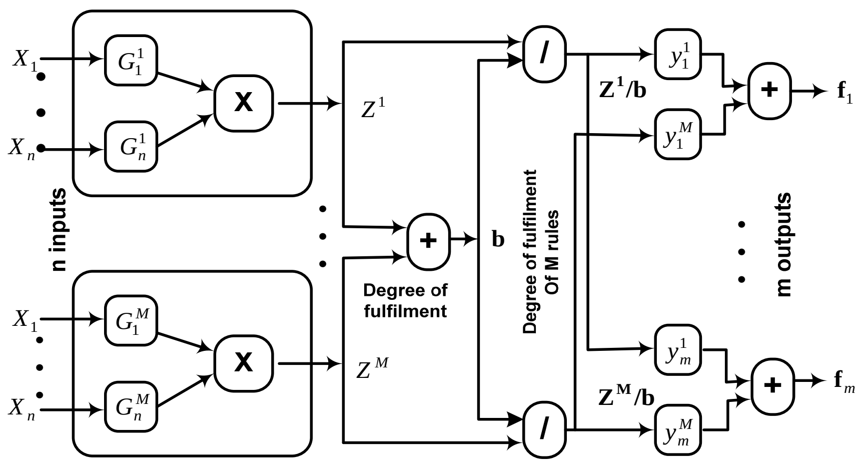


Figure 2. Fuzzy system MIMO feedforward Takagi-Sugeno-type neural-fuzzy network [7]

2.2. Neuro-Fuzzy Training

The neuro-fuzzy is trained with LMA. The LMA was developed by Kenneth Levenberg and Donald Marquardt to accelerate neuro-fuzzy training. If function V(w) is parameter vector w then with Newton method, the update parameter can be defined as

$$\Delta w = -\left[\nabla^2 V(w)\right]^{-1} \cdot \nabla V(w) \tag{1}$$

$$w(k+1) = w(k) + \Delta w \tag{2}$$

where $V(w)$ is taken from sum of squares for error (SSE) formula. $\nabla^2 V(w)$ is Hessian matrix and $\nabla V(w)$ is gradient of $V(w)$, where they are defined as

$$\nabla V(w) = J^T(w) \cdot e(w) \tag{3}$$

$$\nabla^2 V(w) = J^T(w) \cdot J(w) + \sum_{r=1}^{N} e_r(w) \cdot \nabla^2 e_r(w) \tag{4}$$

with Jacobian matrix $J(w)$

$$J(w) = \begin{bmatrix} \frac{\partial e_1(w)}{\partial w_1} & \frac{\partial e_1(w)}{\partial w_2} & \cdots & \frac{\partial e_1(w)}{\partial w_{N_p}} \\ \frac{\partial e_2(w)}{\partial w_1} & \frac{\partial e_2(w)}{\partial w_2} & \cdots & \frac{\partial e_2(w)}{\partial w_{N_p}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial e_N(w)}{\partial w_1} & \frac{\partial e_N(w)}{\partial w_2} & \cdots & \frac{\partial e_N(w)}{\partial w_{N_p}} \end{bmatrix} \tag{5}$$

With Gauss Newton method, the last term in (4) becomes zero, then update parameter becomes

$$\Delta w = -\left[J^T(w) \cdot J(w) + \mu \cdot I\right]^{-1} \cdot J^T(w) \cdot e(w) \tag{6}$$

Then (2) will be

$$w(k+1) = w(k) - \left[J^T(w) \cdot J(w) + \mu \cdot I\right]^{-1} \cdot J^T(w) \cdot e(w) \tag{7}$$

To make the LMA training gives better performance, some researcher add momentum and modified error index, then the SSE becomes

$$SSE_m(w) = 0.5 \cdot \gamma \cdot \sum_{r=1}^{N} \left(e_r(w) - e_{avg}\right)^2 \tag{8}$$

$$e_{avg} = \frac{1}{N} \cdot \sum_{r=1}^{N} e_r(w) \tag{9}$$

where e_{avg} average error and SSE formula after modification is:

$$SSE_{new}(w) = SSE(w) + SSE_m(w) \tag{10}$$

SSE(w) is SSE without modification, therefore the new gradient can be defined with Jacobian matrix as shown equation (11), where learning rate (γ) must be smaller than 1 for LMA training.

$$\nabla SSE_{new}(w) = J^T(w)\cdot\left[e(w)+\gamma\cdot\left(e(w)-e_{avg}\right)\right] \tag{11}$$

In LMA training, the most important thing is calculation process for each layer of Jacobian matrix. The calculation can be derived from SEE for each adjustable parameter of fuzzy logic system, i.e. $W_{oj}^l, W_{ij}^l, c_i^l$ and τ_i^l. So, the Jacobian matrix becomes

$$J^T\left(W_{0j}^l\right) = \left(z^l\right) \tag{12}$$

$$J\left(W_{0j}^l\right) \equiv \left[J^T\left(W_{0j}^l\right)\right]^T = \left[z^l\right]^T \tag{13}$$

$$J^T\left(W_{ij}^l\right) = \left(z^l/b\right)\cdot x_i \tag{14}$$

$$J\left(W_{ij}^l\right) \equiv \left[J^T\left(W_{ij}^l\right)\right]^T = \left[\left(z^l/b\right)\cdot x_i\right]^T \tag{15}$$

When modified error index is added and the result is not always good, it is necessary to add control oscillation. To use control oscillation, it is compulsory to save two set adjustable parameters. If the next epoch in LMA training has lower SSE, then the next iteration must be processed with the new parameter. But when the next epoch over the multiplication of wildness factor (WF) and SSE, the current parameter is used for the next iteration.

3. RESEARCH METHOD

The data from September 2005 to August 2007 is divided into data training (September 2005 to December 2006) and data testing (Januari to August 2007). The electrical load data is taken every 30 minutes, so in a day the number of data is 48. The number of data training is 23,376 for 487 days. For preparation the data is normalized.

For the membership function is Gaussian, the inference rule and weighted average defuzzifier can be defined as

$$f_j = \sum_{l=1}^{M} y_j^l \cdot h^l \tag{16}$$

$$y_j^l = W_{0j}^l + W_{1j}^l x_1 + W_{2j}^l x_2 + \ldots + W_{nj}^l x_n \tag{17}$$

$$h^l = \left(\frac{z^l}{b}\right) \text{, and } \quad b = \sum_{l=1}^{M} z^l \tag{18}$$

$$z^l = \prod_{i=1}^{n} \exp\left(-\left(\frac{x_i - c_i^l}{\sigma_i^l}\right)^2\right) \tag{19}$$

The system needs 2 parameters for Gaussian Membership Function (GMF), i.e. means (χ) and variances (τ), with W_0 and W_i as Takagi-Sugeno parameters. The starting values for these parameters are random.

Means and variance has dimension Mxn, where M is number of GMF and n is number of inputs. Dimension for W_i is Mxk; k is number of output, while W_0 is Mxnxk. The output of Neuro-Fuzzy is calculated with Takagi-Sugeno rule, i.e.

$$y_j^l = W_{0j}^l + W_{1j}^l x_1 + W_{2j}^l x_2 + \ldots + W_{nj}^l x_n \tag{20}$$

with degree of fulfillment

$$z^l = \prod_{i=1}^{n} \exp\left(-\left(\frac{x_i - c_i^l}{\tau_i^l} \right)^2 \right) \tag{21}$$

The output of Neuro-Fuzzy is

$$f_j = \sum_{l=1}^{M} y_j^l \cdot h^l \tag{22}$$

Error will be used in training and it is used according to equation (7) to update the parameter, i.e. means, variances, W_0 and W_i. Parameter μ can be multiplied/divided by a constant according to the SSE value. μ will be multiplied with 200 if current SSE is greater than the previous one. When current SSE is less than the previous one, μ is divided by 10, otherwise, it will be multiplied with 10. If μ is too large or too small, then the value must be set to 10.

The modified error index is used to accelerate the training. For this purpose e(w) in equation (7) is changed to

$$e(w) = e_{MEI}(w) \tag{23}$$

with

$$e_{MEI} = e_r + \gamma \cdot \left(e_r - \frac{1}{N} \cdot \sum_{r=1}^{N} e_r(w) \right) \tag{24}$$

and the formula is added with

$$mo \cdot (w(k) - w(k-1)) \tag{25}$$

γ is set to 0.005.

To find the update parameter of Neuro-Fuzzy, Jacobian matrices and its transpose must calculated. Formula (25)-(28) are used to calculate them for W_0 dan W_{ij}^l

$$J^T\left(W_{0j}^l\right) = \left(z^l\right) \tag{25}$$

$$J\left(W_{0j}^l\right) = \left[z^l\right]^T \tag{26}$$

$$J^T\left(W_{ij}^l\right) = \left(z^l \cdot x_i\right) \tag{27}$$

$$J\left(W_{ij}^{l}\right)=\left[z^{l}\cdot x_{i}\right]^{T} \qquad (28)$$

To get the Jacobian matrices and its transpose for means and variance, the authors use

$$J^{T}\left(c_{i}^{l}\right)=\left\{2\cdot D_{eqv}\cdot z^{l}\cdot\left(x_{i}-c_{i}^{l}\right)/\left(\sigma_{i}^{l}\right)^{2}\right\} \qquad (29)$$

$$J\left(c_{i}^{l}\right)=\left[J^{T}\left(c_{i}^{l}\right)\right]^{T}=\left[2\cdot D_{eqv}\cdot z^{l}\cdot\left(x_{i}-c_{i}^{l}\right)/\left(\sigma_{i}^{l}\right)^{2}\right]^{T} \qquad (30)$$

$$J^{T}\left(\sigma_{i}^{l}\right)=\left\{2\cdot D_{eqv}\cdot z^{l}\cdot\left(x_{i}-c_{i}^{l}\right)^{2}/\left(\sigma_{i}^{l}\right)^{3}\right\} \qquad (31)$$

$$J\left(\sigma_{i}^{l}\right)=\left[J^{T}\left(\sigma_{i}^{l}\right)\right]^{T}=\left[2\cdot D_{eqv}\cdot z^{l}\cdot\left(x_{i}-c_{i}^{l}\right)^{2}/\left(\sigma_{i}^{l}\right)^{3}\right]^{T} \qquad (32)$$

To avoid the oscillation becomes bigger, it is necessary to control it within 1%.

After all parameters are updated and the SSE converges to certain value, the forecasting process for both Short Term Forecasting (STF) and Long Term Forecasting (LTF). The matrix, it is called XIO, for STF could be MISO (Multiple Input Single Output) or MIMO (Multiple Input Multiple Output). The XIO for STF is

$$\textbf{XIO\_STF}=\left[\underbrace{x_1 \quad x_2 \quad x_3 \quad x_4}_{input}\rightarrow\underbrace{\hat{x}_5 \quad \hat{x}_6 \quad \hat{x}_7}_{output}\right] \qquad (33)$$

The equation (33) can be represented for daily or weekly data. It can also represent in interval.

The XIO for LTF is MISO for the result will be used to forecast the next value. The equation (34) shows the XIO for LTF. It is 4 inputs and 1 output.

$$\textbf{XIO\_LTF}=\begin{bmatrix} x_1 & x_2 & x_3 & x_4 & \hat{x}_5 \\ x_2 & x_3 & x_4 & \hat{x}_5 & \hat{x}_6 \\ x_3 & x_4 & \hat{x}_5 & \hat{x}_6 & \hat{x}_7 \\ x_4 & \hat{x}_5 & \hat{x}_6 & \hat{x}_7 & \hat{x}_8 \\ \hat{x}_5 & \hat{x}_6 & \hat{x}_7 & \hat{x}_8 & \hat{x}_9 \\ \hat{x}_6 & \hat{x}_7 & \hat{x}_8 & \hat{x}_9 & \hat{x}_{10} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \qquad (34)$$

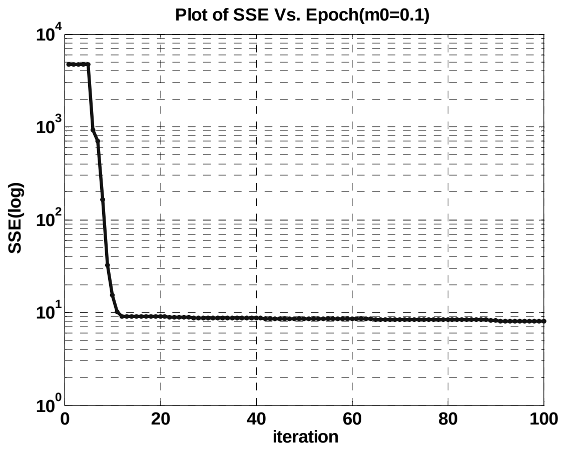
4. RESULT AND DISCUSSION

The first experiment is to find relationship between momentum (mo), modified error index (g) and wildness factor (WF). These experiments use 4 inputs and 3 outputs with the first 2000 data and iteration up to 100. The chosen value for g and WF is 0.0005 and 1.0010. Table 1 shows the summary of these experiments. Table 1 show that the momentum cannot be set so large or even zero. So the recommended value for momentum is small enough but not zero.

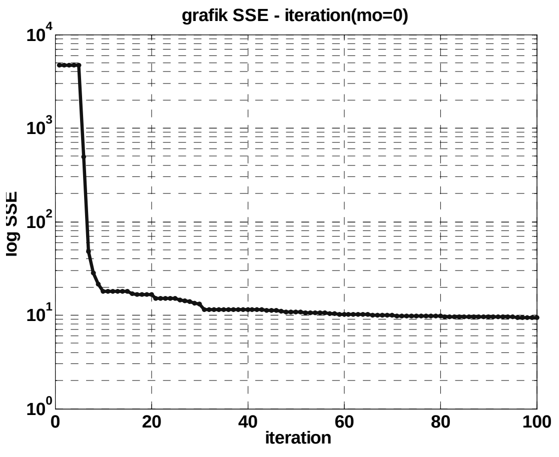
All SSE plot has almost the same shape. Figure 3 shows two of the SSE plot during the experiments. It is shown that the momentum can increase the speed of LMA training. Experiment with mo=0 reach SSE around 10 at 60th iteration while the other get there at 10th iteration. The experiment also shows that the bigger the momentum, the convergence speed of LMA training is not good. The SSE is even larger than the other experiments.

Table 1. Summary of experiments with γ=0.0005 and WF=1.0010

momentum	sum of squares for error
0.00	9.4666
0.05	8.7722
0.10	8.7039
0.75	11.8017



(a)



(b)

Figure 3. SSE training up to 100 epoch, (a). mo=0.00, and (b). mo=0.10

The next experiment is to find suitable value for g. The other parameters is mo=0.1 and WF=1.001. These experiments show that g has small effect in LMA training. The smaller the modified error index, the smaller the SSE. Unfortunately, when g is so small, it looks like there is no significant effect for the SSE (see 0.0005 and 0.00001). With g=0.0005 and mo=0.1, the experiment is to find the suitable value for WF. Table 2 show these experiments result. Table 3 show that the bigger the WF, the larger the SSE. But the interesting point is found when the SSE plot is displayed. Figure 4 shows the effect of WF in the SSE.

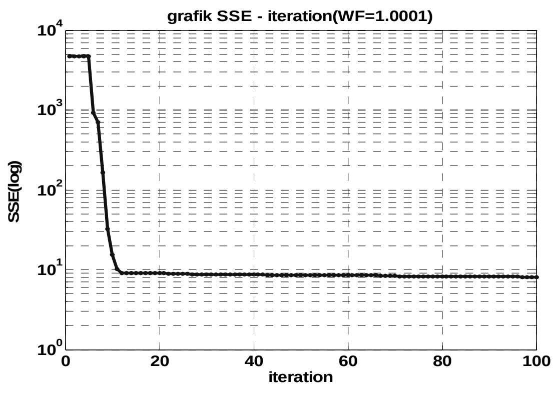
Before using the architecture for forecasting application, it is necessary to check which combination of input-output will give good result. System uses the following parameters: mo=0.005, WF=1.005, GMF=5, gamma=0.0005. Table 4 summaries the result of this experiment. Combination of 4 inputs and 1 output gives the best MSE according to Table 4. Another interesting fact is that system with 1 output is preferable. So the forecasting will use MISO.

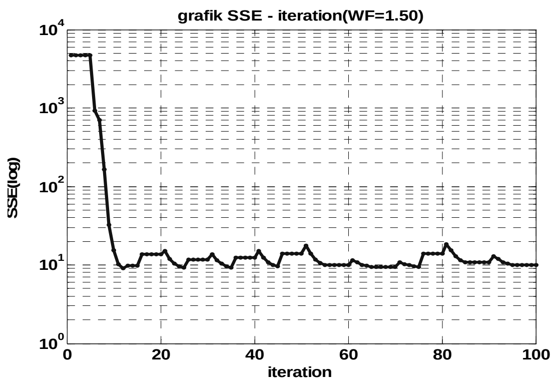
Table 2. Summary of experiments with mo=0.1 and WF=1.0010

learning rate	sum of squares for error
0.70000	8.5873
0.05000	8.7039
0.00050	8.0124
0.00001	8.0224

Table 3. Summary of experiments with g=0.0006 and mo=0.1

wildness factor	sum of squares for error
1.5000	10.0349
1.2500	9.0988
1.0100	8.4222
1.0001	8.1199





(a) (b)

Figure 4. The effect of WF in the SSE, (a). SSE training up to 100 epoch for WF=1.0001, and (b) SSE training up to 100 epoch for WF=1.6000

The next experiment explores the suitable data retrieving process. These experiments use data representation every 30 minutes, every 24h and every week. It is also combined with input variation but the number of output is 1. Table 5 shows the results. The table shows that the best data representation is data per 30 minutes. The result is far better than the other conditions.

Table 4. Combination of input-output and its MSE result

input	output	sum of squares for error	mean of squares for error
4	1	11.7701	0.0010
4	3	37.8668	0.0032
5	1	16.5340	0.0014
5	3	32.8696	0.0028
6	1	23.1269	0.0020
6	3	67.5215	0.0058
7	1	30.0149	0.0026
7	3	38.1239	0.0033

Table 5. Effect of data retrieving in LMA training

input	data retrieving	sum of squares for error	mean of squares for error
4	every 30 m	11.7701	0.0010
4	every 24 h	28.7771	0.0025
4	every week	26.4383	0.0024
7	every 30 m	30.0149	0.0026
7	every 24 h	23.3673	0.0020
7	every week	39.0880	0.0038

Table 6. The number of maximum epochs with SSE results

Maximum Epochs	sum of squares for error training	sum of squares for error forecasting
1	893.7377	567.1948
10	1.4134	1.4934
25	0.5029	0.4368
50	0.4873	0.4265
100	0.4812	0.4260
150	0.4777	0.4215
200	0.4748	0.4196
250	0.4735	0.4205
300	0.4733	0.4227
400	0.4717	0.4211
500	***0.4707***	***0.4108***
1000	0.4670	0.4191
2000	0.4645	0.4190
3000	0.4633	0.4207

The number of epoch in training is also interesting issue. Table 6 shows several maximum epochs with SSE results. All experiments start with the same condition. According to Table 6, it is shown that the number of epochs is 500. When it is increased, the SSE forecasting is decreased until it reaches 500. Now, the architecture is tested for STF. The data is from January to March 2007. Figure 5 shows the plot of error for the whole data. The figure shows that the forecasting cannot follow all real data, especially when the real data is zero (e.g. during black out), the error is very big. This is normal because the condition such as black out is abnormal. To see how the STF follows the real data, Figure 6 shows the first 48 data for January to March 2007, i.e. at January 17, 2007. The STF shows that the forecasting can follow the real data as long as there is no abnormality.

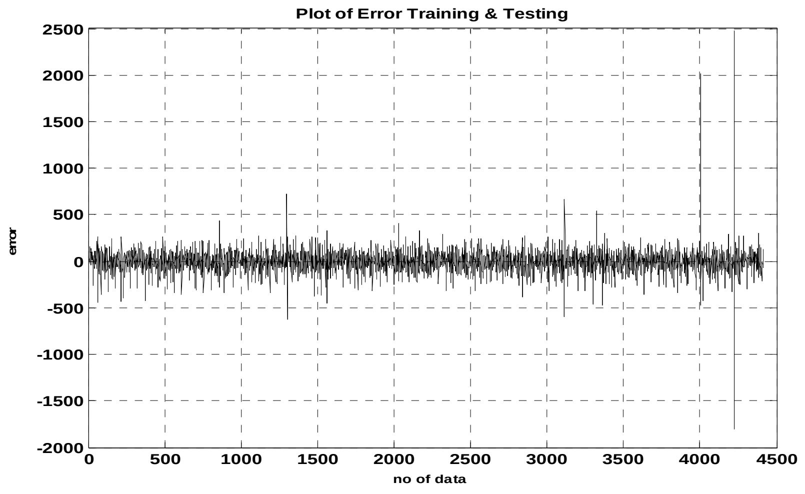


Figure 5. Error plot for STF (January to March 2007)

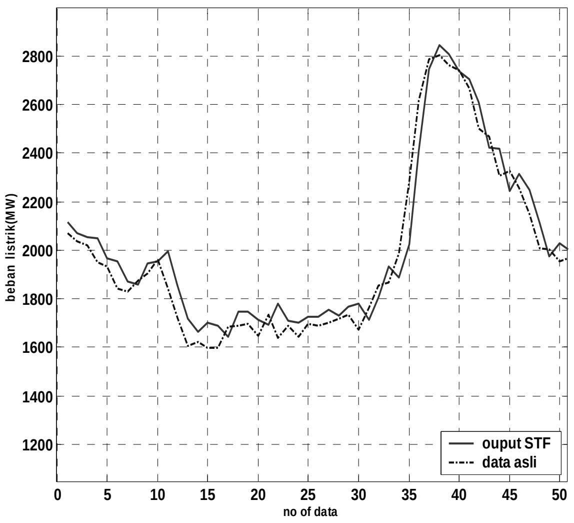


Figure 6. Zoom of first 48 data in STF (January–March 2007)

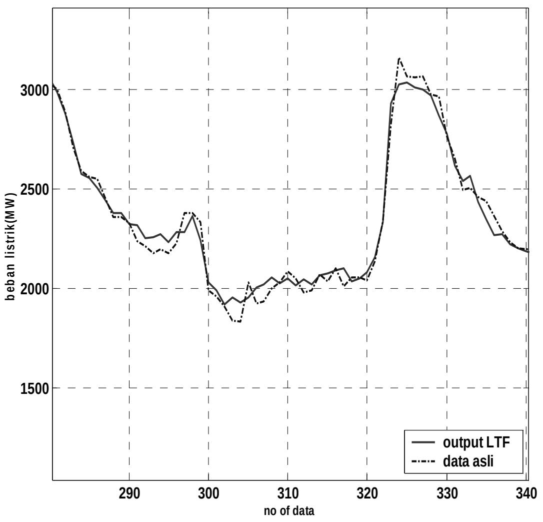


Figure 7. Forecasting for electrical load on June 6, 2007.

In LTF testing, the result will be used as input for the next data forecasting. It means the error will propagate to the next phase. The LTF uses data from June to August 2007. The MSE is 1.1E-3. Figure 7 shows the zoom for LTF on June 6, 2007. The parameters are input=4, output=1, GMF=5, mo=0.005, g=0.0005 and WF=1.001. The LTF result in Figure 7 is interesting for it can follow the real data. But off course, when it is going further, the result is not good anymore. The error from previous phase propagates to the current phase.

5. CONCLUSION

The momentum parameter cannot be set to great value. The greater the value, then the worst the result will be. The recommended value for momentum is less than 0.1. The MSE for momentum value under 0.1 is almost the same. The experiments show that the ability of neuro-fuzzy to forecast electrical load for East Java-Bali is good. The STF for January to March 2007 is 0.0010 while LTF for June to August 2007 is 0.0011. Wildness factor (WF) is interesting. The experiment recommended 5-10% to limit the oscillation.

REFERENCES

[1]. Pasila F, Palit AK, Thiele G. *Long-term Forecasting of Electrical Load using Gustafson-Kessel clustering algorithm on Takagi-Sugeno type MISO Neuro-Fuzzy*. Proceeding of National Seminar on Information technology (SNTI 2006). Jakarta. 2006
[2]. Wijaya LH. Fuzzy clustering dengan metode C-Means untuk forecasting data electrical load time-series. Thesis. Surabaya: Dept. of Electrical Engineering UK Petra; 2007.
[3]. Pasila F. *Evolutionari Algorithm on Fuzzy Clustering Systems using Gustafson-kessel methods for Electrical Load Time-Series*. Industrial Electronic Seminar. Surabaya. 2008
[4]. Pasila F. *Neuro-Fuzzy Forecaster for Modeling and Forecasting Electrical Load Competition using Multivariate Inputs on Takagi-Sugeno Networks*. Conference on Soft Computing, Intelligent System and Information Technology (ICSIIT). Bali. 2007
[5]. Pasila F, Ferdinando H, Lesmana T. *Elman Neural Network with Accelerated LMA Training for East Java-Bali Electrical Load Time Series Data Forecasting*. Proc. of International Confenrence on Information and Communication Technology and System (ICTS). Surabaya. 2009.
[6]. Ferdinando H, Pasila F, Gunawan D, William. *Implementation of Evolutionary Algortihm to Neuro-Fuzzy and Fuzzy Clustering for Electrical Load Time Series Forecasting*. Proc. of 3rd International Conference on Electrical Engineering Design and Technology (ICEEDT). Tunisia. 2009.
[7]. Palit AK, Popovic D. Computational Intelligence in Time Series Forecasting. London: Springer-Verlag, 2005.
[8]. Pasila F. Forecasting of Electrical Load using Takagi-Sugeno type MIMO Neuro-Fuzzy network. Thesis Master. Bremen: University of Bremen; 2006.