Load Forecasting by Fuzzy Neural Network in Box-Jenkins Models

W.K.Tang M.H.Wong Y.K. Wong T.S.Chung

Department of Electrical Engineering
The Hong Kong Polytechnic University

ABSTRACT

In this paper, the use of FNN to identify appropriate Box-Jenkins models for the electricity load forecasting in Hong Kong is presented. FNN is found to be suitable to identify the Box-Jenkin model. By such model, we can forecast the load accurately.

1 INTRODUCTION

Accurate load forecasting is essential for the proper planning of power system expansion and operation. Box-Jenkins models have been applied in time-series forecasting for a long time, and the literature clearly indicates that they are superior to standard time-series forecasting methodologies. However, there are still some reluctance to use Box-Jenkins forecasting in practice owing to the difficulties associated with model identification. Most of the identifications are made by human. Therefore, the human bias could affect the decisions seriously[5].

For the advanced methodologies, every intelligent technique has their own particular computational properties. For example, neural networks are good at recognizing patterns but they are not good at explaining how they achieve their decisions and the learning process is relatively slow and analysis of the trained network is difficult (black box)[10].

On the other hand, fuzzy logic systems, which can reason with imprecise information, are good at explaining their decisions but they cannot automatically acquire the rules they use to make decisions. No standard method exists for transforming human knowledge or experience into the rule base and database of a fuzzy inference system. There is a need for effective methods for tuning the membership functions so as to minimize the output error measure or maximize performance index.

Therefore, the Artificial Neuro-Fuzzy Inference System (ANFIS)[1][2], which is the combination of fuzzy inference system and neural network is proposed. To overcome the problem of knowledge acquisition, neural networks are extended to automatically extract fuzzy rules from numerical data. Neural networks are used to tune membership functions of fuzzy systems that are employed as decision-making systems with lesser development time and cost than by fuzzy logic only.

In this paper, the use of FNN to identify appropriate Box-Jenkins models for the electricity load forecasting of Hong Kong are presented. On the following section, theory of ARIMA and FNN are given. The details of implementation are explained in section 3. The training data set is provided in section 4. The result is reported in section 5. Finally, conclusion is given in section 6.

0-7803-4778-1/98 $10.00 © 1998 IEEE

2 THEORY

2.1 Overall System

Actually, the classical parameters modeling of Box-Jenkins time series analysis can provide very accurate forecast results. These kinds of modeling methods are based on a set of historical load and the random shock data. To get the accurate forecast results, the models of the time series have to be estimated. For the model identification will be divided into two parts. The first part of the model identification is a Stationary Process , which is a system to make sure all the input series to be stationary ; the second part of the model identification is a ARMA Process , which is a system to classify the stationary series into random shock, AR(1), AR(2), MA(1), MA(2) and ARMA(1, 1). Then, the combined system can identify the inputted series into an ARIMA models[4].

After the model is estimated, the classical model of the Box-Jenkins models can use the Least-Square methods to estimate parameters of the model. There are three main types of the models to be implemented. In this paper, the difference-equation form has been used.

2.2 Box- Jenkins time series modeling

Using the Box-Jenkins models to forecast a series, the main problem is the modeling. After, the modeling, the coefficients of the model can be found by the Least-square method or other. Therefore, the method of modeling is very important.

ARIMA(p , d ,q): The Box-Jenkins models are always denoted by ARIMA(p, d, q) formats, which is non-seasonal time-series models consist of three components. The task of the forecaster is to use basic model identification tools to identify the appropriate order for each of the three Box-Jenkins model components.

The first of those components is the AR component. A purely auto-regressive time series includes the current value of the series, which expressed as finite, linear aggregate of previous values of the process and a random shock. AR(p) is expressed as follows:

$$AR: Z(t)= \phi_1 * Z(t-1) + \phi_2 * Z(t-2) + \ldots\ldots + \phi_p * Z(t-p) + a(t) \qquad \textbf{Equation 1}$$

MA process, an observed time series is described as function of a finite number of previous random shocks. MA(q) is expressed as follows:

MA: $Z(t)=-\theta_1 * a(t-1) -\theta_2 * a(t-2) - .....- \theta_q * a(t-q)+a(t)$ **Equation 2**

The third component is the I element. The integration term is use to denote the number of regular differences taken in a time series to achieve stationary. The order of I is denoted as *d*.

The basic tools for Box-Jenkins models are correlograms of auto-correlation coefficients and partial auto-correlation coefficients.

2.3 Fuzzy Neural Network using Type-3

Takagi and Sugeno: Rules based of the Fuzzy inference system

The output of each rule is a linear combination of input variables plus a constant term, and the final output is the weighted average of each rule's output. Figure 1 utilizes a two-rule two-input fuzzy inference system to show different types of fuzzy rules and fuzzy reasoning.

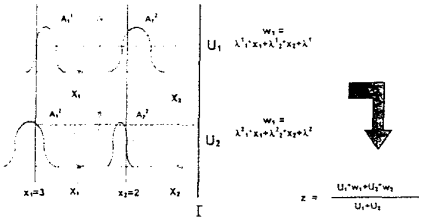

Figure 1.

$$Y_k = \frac{\sum_{j=1}^{M} Wk^j * U^j}{\sum_{j=1}^{M} U^j} \quad \text{Equation 3}$$

$wj=\lambda_0^j+\lambda_1^j x_1+\lambda_2^j x_2+.....\lambda_N^j x_N$ **Equation 4**

$$w^j = \sum_{i=0}^{N} \lambda_i^j x_i$$

$$U_j = \prod_{i=1}^{N} \mu_{A_i^j}(x_i) \quad \text{Equation 5}$$

$$\mu_{A_i^j}(X_i) = e^{-\frac{1}{2}(\frac{x_i - C_i^j}{\sigma_i^j})^2} \quad \text{Equation 6}$$

Input : $X=(x_1, x_2, x_3, x_N)$

Output: $dY=(dY_1, dY_2, dY_3, ..., dY_M)$

The equation 3 is the defuzzification, "Centroid of Area". The equation 4 and equation 5 are the fuzzy rules from Takagi and Sugeno. The equation 6 is the membership functions used here.

Where i= 1 to N, input number
j= 1 to M, rule number
k= 1 to P, output number
$x_0=1$

A_i^j are membership functions of the antecedent part, w_j are the outputs of the consequent part of each rule, μ are the membership value of the corresponding membership functions,
U_j are the membership value of the antecedent part.

Premise parameters: C_i are the mean of the membership functions, σ_i^j are the standard deviations of the membership functions,

Consequent parameters: λ_i^j coefficients of the linear expressions

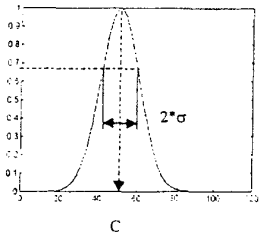

Figure 2. Bell-shaped membership function

Figure 2. shows the Bell-shaped membership function for this project. The C values are the mean of the bell-shaped membership functions. In addition, the 2*a values are the width of the bell-shaped membership functions while the membership values at 0.606531, see the Equation 6.

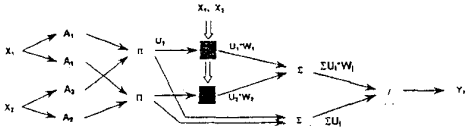

Figure 3. A simple FNN

Training Algorithm: The algorithm of training by Cauchy's method of steepest descent or gradient method is described below.

$$z_i(t+1) = z_i(t) - K\frac{\partial E(Z)}{\partial z_i} \quad \text{Equation 7}$$

where $Z=(z1,z2,z3,.....,zm)$, i=1,2,.....m, E(Z) is the errors of outputs, t is a number of iteration of

learning and K is the learning rate. The objective function E which is defined as follow:

$$E = \sum_{k=1}^{P} \frac{1}{2}(y_k - {}_d y_k)^2 \qquad \textbf{Equation 8}$$

Initialization of the Parameters: One of the important things, that the FNN can be faster than the neural network, is that FNN can have well pre-defined and reasonable initial parameters rather than the neural network use the randomly chosen. For the FNN initially includes M fuzzy rules, the parameters are appropriately chosen on the basis of the first m input and desired output sample pairs as follows:

X^t and ${}_dY^t$ are the input vector and the output vector at the t th iterations.
$X=\{X_1, X_2, X_3, \ldots\ldots, X_N\}$
${}_dY=\{ {}_dY_1, {}_dY_2, {}_dY_3, \ldots\ldots, {}_dY_p,\}$

Premise parameters:

$C_i^j = x_i^j \qquad \forall i \le N, j \le M$

$$\sigma_i^j = \frac{1}{2 * M}[\max(x^j) - \min(x^j)] \qquad \textbf{Equation 9}$$

Consequent parameters:

$\lambda_0^j = {}_dY_j \qquad \forall i \le N, j \le M$

$\lambda_i^j = 0$

where i = 1 to N, input number, N is the number of inputs

j= 1 to M, rule number, M is the number of rules

k= 1 to P, output number

The above initializations perform a fuzzification of the selected input points within the premise space. The mean values of the memberships are centered directly at these points, while the membership deviations reflect the degree of fuzzification and are selected in such way that a prescribed degree of overlapping exists between successive memberships.

Rule Base Adaptation: Another feature, which has been employed into this FNN, is the rule base adaptation. The rule bases can extend itself while the necessary has been reached. To determine whether add a new rule or not, the degree of fulfillment, ΣU_j, is compared with the threshold value of the firing strength, β. If the degree of fulfillment is smaller than the threshold value, that meant all the rules in the existing rule base cannot describe the current input-output pairs as well as the threshold value requirement. Therefore, the new rule R should be added to the rule base system. The mean values C of the new rule are set to the inputs. Moreover, the overlapping of the new rule with the old rules is set to

$C_i^{M+1} = X_i^p$ **Equation 10**

$\sigma_i^{M+1} = \gamma * (x_i^p - C_i \text{ nearest})$ **Equation 11**

β the least acceptable degree of excitation of the existing fuzzy rule.
γ the overlapping factor.

3 IMPLEMENTATIONS

3.1Box-Jenkins models

$Z(t)(1-B)^d(1-\phi_1 B-\phi_2 B^2)=e(t)(1-\theta_1 B-\theta_2 B^2)$ **Equation 12**

The overall project is based on the model Box-Jenkins Model[4][5][6], for this model. the non-seasonal identification is used.

3.2 Modified Fuzzy Neural Network

We have made these modifications mainly at the Rule Base Adaptation. Initially, the criteria of determine whether to add a new rule does consider the $\beta<\Sigma U_j$. However, this setting is not a suitable rule for rule adding. A new criteria is proposed by us and used in this project which made the overall rule base adaptation more meaningful.

$\beta^N<\Sigma U_j$. **Equation 13**

where β is the least acceptable degree of excitation of the membership values.

The standard deviations and the overlapping factor γ

The standard deviations are:
$\sigma_i^{M+1} = \gamma * (x_i^p - C_i \text{ nearest})$

However, there may have some x_i^p ,which are same as some C_i nearest. It makes the σ_i^{M+1} tend to zero, then the reciprocal of σ_i^{M+1} would be too large and the width of the membership function would be too small. In addition, the number of rules would be increased as unnecessary. The smaller the σ_i^{M+1} is, the bigger the number of rule is

$\sigma_i^{M+1}(new)=\max(\sigma_i^{M+1}.\min(\sigma))$

3.3 Stage 1 (Stationary Process)

The number of input and output, β and γ of the fuzzy neural network was designed as 12, 2, 0.2, and 0.9 respectively. There are first six auto-correlation coefficients of the input series and the corresponding standard errors[4] of the auto-correlation coefficients. Output is {0,1} to represent stationary, output is {1,0} to represent non-stationary, The total number of inputs to the FNN is 380 series. The first one is **Degree of Difference**,

this is a loop for the input values. This system can determine the input series whether stationary.

$$acf(k) = \frac{\sum_{t=1}^{N-K}(x_t - \bar{x})(x_{t+k} - \bar{x})}{\sum_{t=1}^{N}(x_t - \bar{x})^2}$$

3.4 Stage II (ARMA Process)

Using the series, which has been difference enough to be stationary, input to the Stave II system to identify the corresponding model. This stage can resolve the stationary series into one of the six basic Box-Jenkins Models[4][5], Therefore, the input of the network fundamental structure will be 36 inputs and 6 outputs. The inputs of the 36 inputs will be 12 estimated auto-correlation 12 standard errors and 12 estimated partial auto-correlation. The 6 outputs will classify the input series is which of the model, the corresponding digital will be one when it is that model. The training data is using the TrainingSetI 4.2.

3.5 Stage III (Forecasting Process)

After all of the fuzzy neural network identifications, this part is going to find the parameters of the specified model from the previous process and then do the forecasting. Therefore, the inputs of this part is the model value and the series. Forming the model is according the input's model kind. The backcasting of the random shock also uses the estimated parameters to determine the necessary random shock in past and may be the starting values. The final block is the Forecasting, which is using the model and the parameters and past random shock to forecast the series. There is a note that all of the future random shock would be set to zero completely[4].

4 TRAINING DATA SET

For standardization of the notations, the names of the sections in this chapter are the same name of the objects in the program.

4.1 Training Set

To generate the training sets, the Box-Jenkins models have to be classified into more details. Actually, there are total 19 series for the six basic Box-Jenkins models[4].

To cover all the cases, the training set should be concerned all the possible cases have the same numbers. Also, all the generated series would be at least 50 in length[5][6]. Gaussian noise model (white noise) has been used to generate the random shock series in variance is one and the mean is zero (mean=0, variance = 1).

This part generates only 10 sets of time series while each has 50 elements. At the same time, there are 190 model parameters have been generated such that there are 10 random sets for each above cases. All the parameters are according to the restrictions of the principles to be invertible [4]. After the 10 series have been generated, all of them are passed into each of the above 19 cases. Thus, the 190 stationary series have been generated. In the generation loop, series is generated by using the Gaussian noise functions. The program checks the series whether mean is zero and variance is one. If they are far from expectancy, that series has to be abonded.

4.2 Training Set I for Stage I (Stationary Process)

This part generates a set of time series for the stationary Process . Therefore, the training sets are the combination of stationary and non-stationary series. The total number of series would be 380 and there are 190 series in stationary and 190 series in non-stationary. Among the 190 non-stationary, there are 95 series in first order of the difference and the remainder of the 95 series in second order. Therefore, the number of stationary series is equal to the non-stationary and the first order and second order cases can also be included.

The used equations are as follows:
First order: Z(0)=0. Z(t)=U(t)+Z(t-1)
Second order: Z(0)=Z(1)=0, Z(t)=U(t)+2*Z(t-1)-Z(t-2)
**Notes: where $\forall t > 0$, $t \in R$, Z(t) is the final series and U(t) is the input from Training Set (the 190 stationary series).*

For a model to identify a series whether stationary or non-stationary, the auto-correlation coefficients are good enough to do so. However, the theoretical auto-correlation coefficients cannot be found, so the standard errors of the corresponding ACF have to be included in the training set. After the 380 series have been generated, these series will not be used as the inputs for the system Stationary Process directly. All of the 380 series have to be generated the auto-correlation coefficients and the estimated standard errors of the corresponding series.

The Barlett's approximation is used to find the standard errors of auto-correlation coefficients of the series.

4.3 Training Set II for Stage II (ARMA Process)

This part generates a set of time series for the ARMA Process . The inputs come from the ***Training Set*** part. ALL THE INPUTS SERIES FOR THIS PART SHOULD ALSO BE STATIONARY. All of the auto-correlation coefficients and partial auto-correlation coefficients of each inputs series have to be calculated first. There are 12 ACF, 12 standard errors and 12 PACF, thus the size of the series is 36. There are three Training Set so there are 30 series for the 19 cases. The total number of series would be *570*.

5 RESULTS

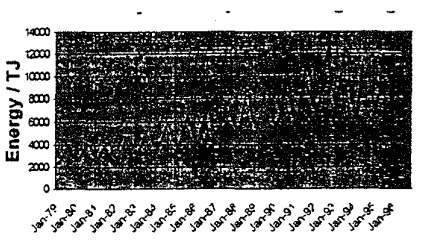

Figure 4. Electricity Consumption in Hong Kong

The data set has been chosen from 1979 to 1990 and the program is trying to forecast 2 years ahead (1991 to 1992).

$$Z(t)(1-B)^d(1-\phi_1B-\phi_2B^2)=e(t)(1-\theta_1B-\theta_2B^2)$$

a seasonal term has been added into the AR side.

$$Z(t)(1-B^{12})(1-B)^d(1-\phi_1B-\phi_2B^2)=e(t)(1-\theta_1B-\theta_2B^2)$$

The twelve years data would be handled by the program.

There is one more term has been inserted into formula, the seasonal component.

5.1 Fuzzy Neural Network

Before applying the FNN, some testing have to be done. To test the performance of the FNN, the most simple model XOR model has been illustrated.

Cases	4
Inputs	2
Outputs	1
Iterations	10
β	0.2
γ	0.7
Learning rate	0.9
start rule	1
Input space (min)	0
Input space (max)	1

X_1	X_0	Desired output (d_{YS})	Output(Y_S)
0	0	0.0	0.05597
0	1	1.0	0.95698
1	0	1.0	0.95684
1	1	0.0	0.05110

For this testing, the FNN can learn 3 new rules to tune all of them to suit the training set. In addition, we can extract the knowledge from the rule base as follows:

	Premise Parameters				Consequent Parameters		
	X_1		X_0				
	C_1	σ_1	C_0	σ_0	λ_0	λ_1	λ_2
Rule 1	0	0.5	0	0.5	-0.6	0.03	0.03
Rule 2	0	0.5	1	0.8	0.93	0.005	0.24
Rule 3	1	0.5	0	0.44	0.93	0.24	0.005
Rule 4	1	0.4	1	0.4	-0.052	-0.06	-0.06

Table 1 The final parameters of XOR model after 10 iterations

5.2 Stage I (Stationary Process)

Cases	380
Inputs	12
Outputs	2
Iterations	100
β	0.2
γ	0.7
Learning rate	0.9
start rule	1
Input space (min)	-1
Input space (max)	1

Table 7. FNN setting for Stationary Process

Total data set	380
Right	363 (95.5%)
Wrong	17 (4.5%)
Over-differenced	12 (6.3 %)
Under-differenced	5 (2.6 %)

Table 8. Results of the testing data set

After training, the testing data also is input. From the Table 8, even though the Stationary Process cannot give the 100% right, the result is still excellent (95.5%). Usually, the under-differences is worse than over-differences. It is because that the under-differences would make the series is non-stationary, but the over-differences would add the artificial patterns in a series. Among of the wrong cases, there are 12 cases that are over-differences. The percentage is calculated by the over-differences over the number of stationary series. On the percentage, since the percentage of over-differences is bigger than under-differences, the result is acceptable.

The result of the electricity consumption series is 1. (first order difference needed)

5.3 Stage II (ARMA Process)

Cases	570
Inputs	36
Outputs	6
Iterations	500
β	0.2
γ	0.7
Learning rate	0.75
start rule	1
Input space (min)	-1
Input space (max)	1

Table 9. FNN setting for ARMA Process

Total data set	570
Right	517 (90.7 %)
Wrong	53 (9.3%)

Table 10. Results of the testing data set

For this section 3, this FNN is the most complex compared with the Stationary Process. The size of the inputs is the triple of the previous one. The size of the outputs is also trebled. The result is also acceptable (90.7% right).

5.4 Stage III(Forecasting Process)

The model is identified as a MA(1) model. The following graph show the measured and forecasting load. The mean of the absolute error is 5.75% and the max. absolute error is 13.81%. This is a good result for forecasting. Therefore, the overall system is a good electricity load consumption forecasting model.

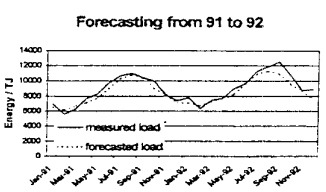

Figure 12. The final forecasting results

6 CONCLUSION

The aim of this paper is to use the FNN to identify appropriate Box-Jenkins models for time series especially for electricity consumption in Hong Kong. The FNN can give the acceptable accuracy, after the training process. Then, the electricity consumption can be forecasted by the Box-Jenkins models.

On the whole, Box-Jenkins models are superior in performance. However, there are many clouds while a person tries to determine which model it is. One approach to the Box-Jenkins model identification problem that had not been investigated was the use of fuzzy neural network. In this project, fuzzy neural network (FNN) is shown to be powerful tools in relating sets of inputs to appropriate outputs in cases where no algorithmic relationship exists. Box-Jenkins models can help us to solve the problems on the forecasting and the FNN can solve the problems of the identifications.

In future development, the non-seasonal Box-Jenkins models can be replaced by the seasonal models. If the available data is also sufficient, the seasonal Box- Jenkins models should give more accurate results. The membership functions and fuzzy reasoning can also be changed into other methodologies to achieve different performance of the FNN.

7 REFERENCE

[1] Jyh-Shing Roger Jang, ANFIS: Adaptive-Network-Based Fuzzy Inference System , **IEEE Transactions on Systems, Man, and Cybernetics,** P.665-685, vol.23. No.3, May/June 1993.

[2] Jyh-Shing Roger Jang and Chuen-Tsai Sun, *Neuro-Fuzzy Modeling and Control* , **Proceedings of the IEEE,** P.378-403, vol.83. No.3, March 1995.

[3] A.G.Bakirtzis, (M), J.B. Theocharis, (M), S.J. Kiartzis, (S), and K.J. Satsios, *Short term load forecasting using fuzzy neural networks* , **IEEE Transactions on Power Systems,** Vol.10, No.3, August 1995: 1518-1524.

[4] Box G. E. P., and G. M. Jenkins, **Time Series Analysis Forecasting and Control.** , Oakland, Calif.: Holden-Day. 1976.

[5] Steven B. Reynolds, Joseph M. Mellichamp, and Robert E. Smith, *Box-Jenkins Forecast Model Identification* , AI **EXPERT:** 15-28, June 1995.

[6] Chris Chatfield, **The Analysis of Time Series -** *An Introduction* , Fifth Edition, Chapman & hall 1996.

[7] Hong Kong Monthly Digest of Statistics , **Census and Statistics Department, Hong Kong Government.**