J. Ind. Soc. Agril. Statist.
62(3), 2008 : 214-220

Neuro-Fuzzy Approach for Modelling and Forecasting: An Application

Rama Krishna Singh and Prajneshu
Indian Agricultural Statistics Research Institute, New Delhi
(Received: June 2007)

SUMMARY

Artificial neural network and Fuzzy logic provide attractive ways to capture nonlinearities present in a complex system. Neuro-Fuzzy modelling, which is a newly emerging versatile area, is a judicious integration of merits of above mentioned two approaches. In this paper, an important model from this class, viz. Adaptive Neuro-Fuzzy Inference System (ANFIS) is thoroughly studied. The model is implemented on Fuzzy Logic Toolbox of MATLAB using ANFIS. As an illustration, the methodology is applied for development of a forecasting model for secondary data of yield of 100 banana plants on the basis of data at six different stages of growth using several biometrical characters like plant height, plant girth and leaf length as predictors.

Key words: Neuro-Fuzzy, ANFIS, Membership function, MATLAB, Mean square error.

1. INTRODUCTION

Multiple linear regression (MLR) methodology is extensively employed to determine relationship between response variable and a set of predictors (Sengupta *et al*. 2001). However, one limitation of this methodology is that resultant model is assumed to be "linear". In a realistic situation, this assumption is rarely satisfied. Also, if there are several predictors, it is well nigh impossible to have an idea of underlying nonlinear functional relationship between response variable and predictors. Fortunately, to handle such a situation, an extremely powerful approach of "Artificial neural networks" (ANNs) is rapidly developing. Recently, Singh and Prajneshu (2007) have thoroughly studied "Multilayered Feedforward ANN" and applied it to forecast maize crop yield on the basis of four predictors.

Another limitation of MLR is that underlying phenomenon, response variable, and predictors are all assumed to be "crisp" or "precise". In reality, one or more of these is vague, or imprecise, or fuzzy. Accordingly, area of "Fuzzy logic" has been developed (Klir and Yuan 2000). Incorporating this aspect in ANN methodology, a rapidly developing area of "Neuro-fuzzy" has been emerging (Rutkowaska 2002). In an excellent paper, Abraham *et al*. (2004) applied Neuro-fuzzy techniques for forecasting time-series meteorological subdivisions level data of Kerala. However, in that paper, development of the process only over time was considered. Extension and application of this type of work when several predictors are present, is a challenging task.

Purpose of this paper is to make an indepth study of "Adaptive neuro-fuzzy inference system (ANFIS)", which is most popular in the family of Neuro-fuzzy modelling. As an illustration, ANFIS model, using "MATLAB Fuzzy Logic Toolbox" is applied for development of a forecasting model for secondary data of yield of 100 banana plants on the basis of data at six different stages of growth using several biometrical characters like plant height, plant girth, and leaf length as predictors.

2. NEURO-FUZZY COMPUTING

In a real world situation, traditional equation based techniques are not suitable for modelling nonlinearity. Neuro-fuzzy computing (Jang *et al*. 2004) is a judicious integration of merits of neural and fuzzy approaches. This incorporates generic advantages of Artificial neural networks like massive parallelism, robustness, and learning in data-rich environments into the system.

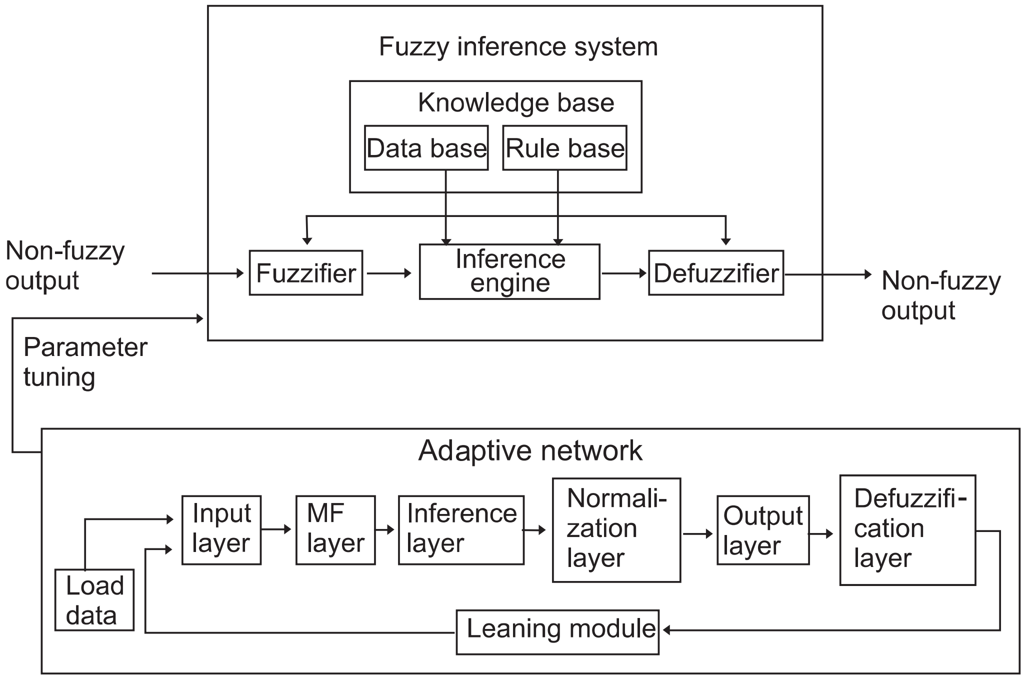

Fig. 1. Conceptual diagram of ANFIS

Modelling of imprecise and qualitative knowledge as well as transmission of uncertainty is possible through use of Fuzzy logic (Klir and Yuan 2000). Present study is based on ANFIS, which is a multilayered feedforward ANN consisting of nodes and directional links through which nodes are connected. Moreover, part or all the nodes are adaptive, which means that their outputs depend on incoming signals and on the parameter(s) pertaining to these nodes. ANFIS either uses input/output data sets to construct a fuzzy inference system whose membership functions are tuned using a learning algorithm or an expert may specify a fuzzy inference system and then the system is trained with data pairs by an adaptive network. Conceptual diagram of ANFIS is shown in Fig. 1.

A Fuzzy inference system (FIS) has five functional blocks. A fuzzifier converts real numbers of input into fuzzy sets. The database (or dictionary) contains the membership functions of fuzzy sets. The membership functions provide flexibility to fuzzy sets in modelling. A rule base consists of a set of linguistic statements of the form, if x is A then y is B, where A and B are labels of fuzzy sets on universes of discourse X and Y, respectively. An inference engine performs the inference operations on the rules to infer the output by a fuzzy reasoning method. Defuzzifier converts the fuzzy outputs obtained by inference engine into a non-fuzzy output real number domain. In order to incorporate the capability of learning from input/output data sets in fuzzy inference systems, a corresponding Adaptive network is generated. An Adaptive network is a multilayered feedforward network consisting of nodes and directional links through which nodes are connected. As shown in Fig.1, Layer 1 is the input layer; Layer 2 describes membership functions of each fuzzy input, Layer 3 is inference layer and normalization is performed in Layer 4. Layer 5 gives output and Layer 6 is defuzzification layer. Learning rule specifies how parameters of adaptive nodes should be changed to minimize a prescribed error measure. Change in values of parameters results in change in shape of membership functions associated with FIS. The modelling process based on ANFIS can broadly be classified in three steps:

Step 1: System Identification

First step in system modelling is identification of input and output variables called system's variables.

Step 2: Determining Network Structure

Once input and output variables are identified, Neuro-fuzzy system is realized using a six-layered network.

Layer 1 (Input Layer)

Each node in Layer 1 represents input variables of the model. This layer simply transmits these input variables to fuzzification layer.

Layer 2 (Fuzzification Layer)

This layer describes membership function of each input fuzzy set. Membership functions are used to characterize fuzziness in fuzzy sets. Output of each node i in this layer is given by $\mu_{A_i}(x_i)$, where $\mu_A(x)$ denotes membership function. Its value on unit interval [0,1] measures degree to which element x belongs to fuzzy set A, x_i is input to node i and A_i is linguistic label for each input variable associated with this node. Gausian membership functions are employed, since they are nonlinear and smooth and their derivatives are continuous and is given by

$$\mu(x) = \exp\left[-(x - b/a)\right]^2 \tag{1}$$

Layer 3 (Inference Layer)

Third layer is the inference layer. Each node in this layer is a fixed node and represents the IF part of a fuzzy rule. This layer aggregates membership grades using any fuzzy intersection operator which can perform fuzzy AND operation. The fuzzy intersection operators are commonly referred to as T-norm (triangular norm) operators. Most frequently used T-norm operators are *min* or *product* operators. For example, IF $x_1$is A_1 AND x_2 is A_2 AND x_3 is A_3, THEN y is f (x_1, x_2, x_3), where f (x_1, x_2, x_3) is a linear function of input variables or may be a constant. The output of ith node is given as

$$w_i = \mu_{A_1}(x_1)\times\mu_{A_2}(x_2)\times\mu_{A_3}(x_3) \tag{2}$$

Layer 4 (Normalization Layer)

The ith node of this layer is also a fixed node and calculates ratio of ith rule's firing strength in inference layer to sum of all the rules' firing strengths as

$$\overline{w}_i = w_i/(w_1+w_2+\square +w_R) \tag{3}$$

where i = 1, 2, ..., R and R is total number of rules.

Layer 5 (Output Layer)

This layer represents the THEN part (i.e. the consequent) of the fuzzy rule. The operation performed by the nodes in this layer is to generate qualified consequent (either fuzzy or crisp) of each rule depending on firing strength. Every node in this layer is an adaptive node. The output of the node is computed as

$$O_i = \overline{w}_i f_i \tag{4}$$

where w_i is a normalized firing strength from Layer 3 and f_i is a linear function of input.

Layer 6 (Defuzzification Layer)

This layer aggregates qualified consequents to produce a crisp output. The single node in this layer is a fixed node. It computes weighted average of output signals of output layer as

$$O = \sum_i O_i = \sum_i \overline{w}_i f_i = \sum_i w_i f_i \Big/ \sum_i w_i \tag{5}$$

Step 3: Learning Algorithm and Parameter Tuning

ANFIS model fine-tunes parameters of membership functions using either backpropagation learning algorithm or hybrid learning rule. Backpropagation algorithm is an error-based supervised learning algorithm. It uses gradient descent method to update parameters. Network output is compared with desired output values. The error measure EP, for pattern P at the output node in Layer 6 may be given as

$$E^P = 1/2\left(T^P - O_6^P\right)^2 \tag{6}$$

where T^P is target or desired output and O_6^P, single node output of defuzzification layer in the network. Further, sum of squared errors for entire training data set is

$$E^P = \sum_P E^P = \frac{1}{2}\sum_P\left(T^P - O_6^P\right)^2 \tag{7}$$

The error measure with respect to node output in Layer 6 is given by

$$\delta = \partial E/\partial O_6 = -(T - O_6) \tag{8}$$

This delta value gives the rate at which output must be changed in order to minimize error function. This delta value must be propagated backward to inner layers in order to distribute error of output unit to all layers connected to it and adjust corresponding parameters. The delta value for Layer 5 is given as

$$\partial E/\partial O_5 = (\partial E/\partial O_6)*(\partial O_6/\partial O_5) \tag{9}$$

Now, if α is a set of design parameters of the given adaptive network, then

$$\partial E/\partial\alpha = \sum_{O'\in P}(\partial E/\partial O')*(\partial O'/\partial\alpha) \tag{10}$$

where P is set of adaptive nodes whose output depends on α. Thus, update for parameter α is given by

$$\Delta\alpha = -\eta * (\partial E/\partial\alpha) \tag{11}$$

Here η is learning rate given by

$$\eta = k \Big/ \sqrt{\sum_{\alpha} (\partial E/\partial\alpha)^2} \tag{12}$$

where k is step size. Value of k must be properly chosen as change in value of k influences rate of convergence. Thus, design parameters are tuned according to real input/output data pairs of the system. Change in values of the parameters results in change in shape of membership functions initially defined by an expert. The new membership functions thus obtained after training gives a more realistic model of the system.

3. AN ILLUSTRATION

An hectare of banana yields 40 million calories of energy as compared to 2.5 million calories by wheat (Rao 2005). India ranks second among banana producing countries of the world. Venugopalan and Shamasundaran (2005) developed a statistical model for evolving crop-logging parameters across different growth stages of banana plants collected from farmers' fields located at Kestur, Bangalore. In present study, data culled from Venugopalan and Shamasundaran (2005), has been used to apply ANFIS to forecast banana yield at different stages of its growth using a number of predictors like number of leaves, plant height (cm.), plant girth (cm.), leaf length (cm.), leaf breadth (cm.), number of hands/bunch, and number of fingers/hand. Out of total data for 100 banana plants, data for 80 banana plants is used for "Training" while data for remaining 20 plants is used for "Validation". In order to have a visual idea, a MATLAB plot of variables for first stage growth is exhibited in Fig. 2. Fuzzy Logic Toolbox available in MATLAB is used for training ANFIS. A computer program was written in MATLAB and the same is appended as Annexure-I. In view of availability of data at different stages of growth corresponding to different numbers of predictors, a five input-one output system for first four growth stages of banana and seven input-one output system for last two stages is considered for development of ANFIS model.

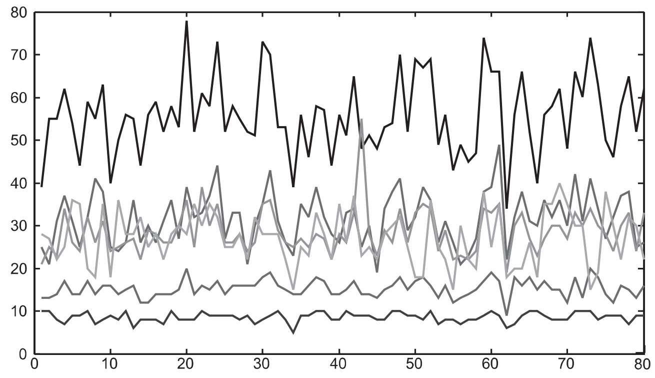

Fig. 2. MATLAB Plot of input-output variables in training set for first stage of banana plant

The input variables are represented with Gaussian membership function. For building ANFIS, membership function of each input was tuned using hybrid method consisting of backpropagation for parameters associated with input membership function (MF) and least square estimation for parameters associated with output membership functions. Computations of membership function parameters are facilitated by a gradient vector, which provides a measure of how well the FIS system is modelling input/output data. For a given set of parameters, numbers of nodes in training data were found. The numbers of linear parameters and nonlinear parameters were identified. The hypothesized initial number of membership functions and type used for each input were taken as six. Now, hypothesized FIS model is trained to emulate training data by modifying MF parameters according to chosen error criterion. A suitable configuration has to be chosen for best performance of the network. Goal for the error was set to be 0.1 and number of training epochs was given as 200. After training (with 200 epochs) was complete, final

Table1. Fuzzy Information Structure for different stages of banana plant

Stages / Fuzzy Information System	I	II	III	IV	V	VI
Number of inputs	5	5	5	5	7	7
Number of MF for each input	6	6	6	6	6	6
Number of fuzzy rules	30	30	30	30	42	42
Number of linear parameters	140	146	164	184	108	80
Number of nonlinear parameters	300	310	340	360	154	110
Number of training epochs	200	200	200	200	200	200
Number of training data set	80	80	80	80	80	80
Number of test data set	20	20	20	20	20	20
Error goal	0.01	0.01	0.01	0.01	0.01	0.01

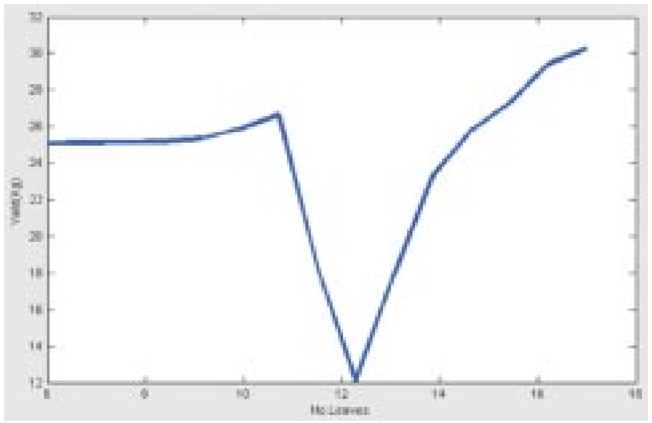

Fig. 3. Two-dimensional plot for "second stage" between yield (Kg.) and number of leaves

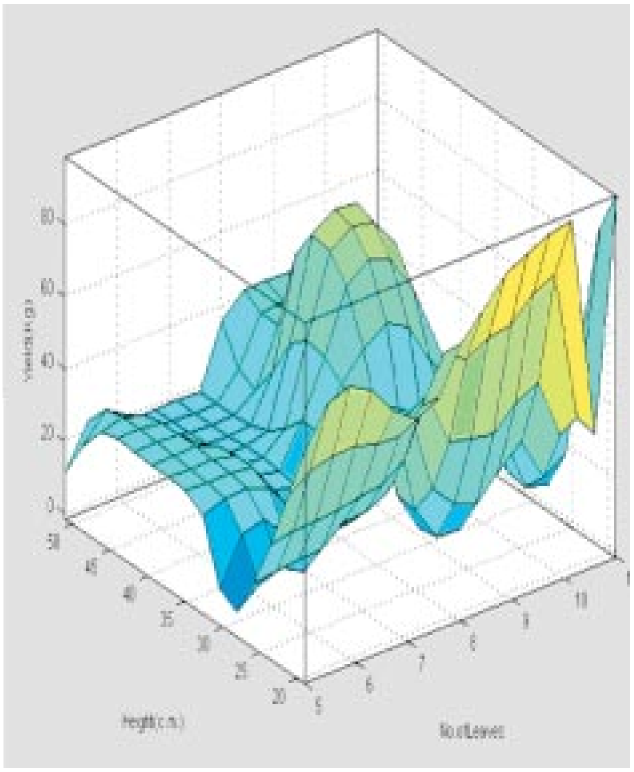

Fig. 4. Three-dimensional plot for "second stage" between yield, height and number of leaves

configuration for FIS is reported in Table 1. Several two and three-dimensional plots were drawn for proposed ANFIS model, two of which are depicted in Figs. 3 and 4. Evidently, the surface is complex and highly nonlinear. Rule viewer for first stage is depicted in Fig. 5. The Rule Viewer displays a road map to the whole fuzzy inference process. Each row of plots represents one rule.

Model was trained for 200 epochs and it was observed that most of the learning was complete in 150 epochs as error goal settles down to almost zero percent at around 150th epoch for all the six stages. Mean square error (MSE) for validation set is computed to compare performance of ANFIS model for different stages and the same is reported in Table 2. Evidently, MSE decreases as number of stage of plant growth increases, which is quite logical. Further, MSE at first and second stages are very high. As there is a considerable decrease in third stage, banana yield may be forecasted at third stage with reasonable accuracy. However, if much more accuracy is required, one would have to wait until 6th stage.

Table 2. Comparison of MSE (Validation data set) for all six stages of growth in banana

Stages / MSE	I	II	III	IV	V	VI
Validation data set	366.04	236.55	102.47	95.35	79.29	39.08

4. CONCLUDING REMARKS

Purpose of this paper is to highlight the importance of a very powerful and versatile methodology of Neuro-fuzzy modelling and not development of a forecasting model for banana plant yield *per se*. In the illustration considered, we had data of only 100 banana plants. Had data of more number of banana plants say, 500 were available, resultant model would have been more efficient! As a rule of thumb, at least 500 data points are required for training and validation of an ANFIS model efficiently. It is hoped that, in future, research workers would start applying ANFIS Neuro-fuzzy modelling approach to their data sets.

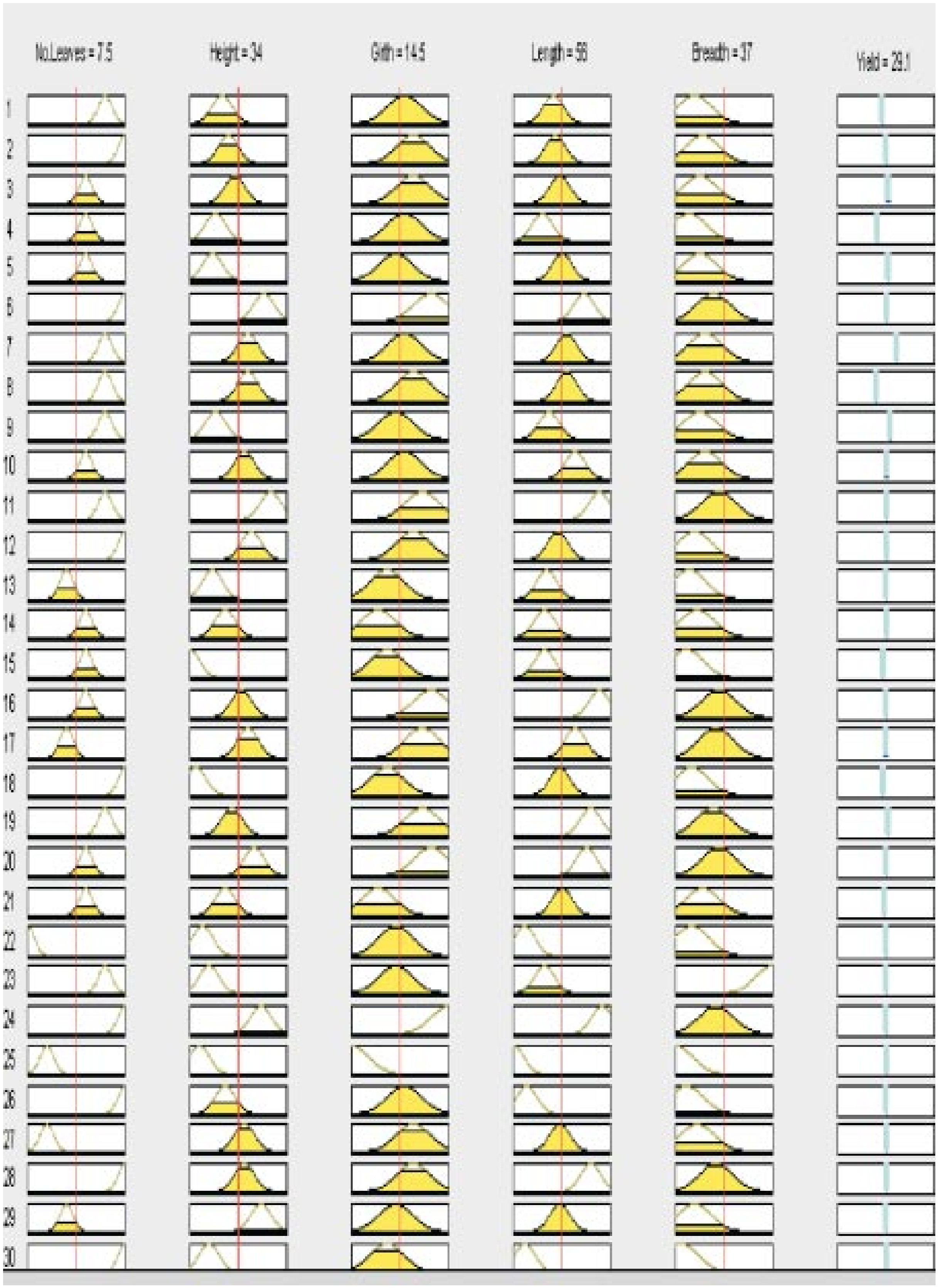

Fig. 5. Rule-viewer for membership functions of variables under study in "first stage" of growth in banana plant

ACKNOWLEDGEMENT

Authors are grateful to Dr. R. Venugopalan for providing data.

REFERENCES

Abraham, A., Philip, S. and Mahanti, P.K. (2004). Softcomputing models for weather forecasting. *Int. J. Appl. Sci. Compu.*, **11**, 106-117.

Jang, J.S.R., Sun, C.T. and Mizutani, E. (2004). *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Pearson Education, New Delhi.

Klir, G.J. and Yuan, B. (2000). *Fuzzy Sets and Fuzzy Logic: Theory and Applications.* Prentice-Hall, India

Kukolj, D. (2002). Design of adaptive Takagi-Sugeno-Kang fuzzy models. *Appl. Soft. Comput.*, **2**, 89-103.

Matlab@ (2006). *Fuzzy Logic Toolbox User's Guide*. The Math Works Inc., U.S.A.

Rao, V.N.M. (2005). *Banana*. Directorate of Information and Publications of Agriculture, Indian Council of Agricultural Research, New Delhi, 69.

Rutkowaska, D. (2002). *Neuro-fuzzy Architectures and Hybrid Learning.* Physica-Verlag, U.S.A.

Sengupta, K., Nandi, S. and Chakraborty, N. (2001). Effect of sulphur-containing fertilizers on productivity of rainfed greengram *(Phaseolus radiatus)*. *Ind. J. Agri. Sci*., **71**, 408-410.

Singh, R.K. and Prajneshu (2007). Artificial neural network methodology for modelling and forecasting maize crop yield. *Ag. Econ. Res. Rev*. (To appear).

Venugopalan, R. and Samasundaran, K.S. (2005). Statistical model for evolving crop-logging technique in banana. *Trop. Agril.*, **82**, 23-27.

ANNEXURE-I

MATLAB CODES FOR ANFIS

```
load 'stage1.txt';
input=stage1(1:80,1:5);
output=stage1(1:80,6);
input_chk=stage1(81:100,1:5);
out_chk=stage1(81:100,6);
trndata=[input output];
chkdata=[input_chk out_chk];
stepsize = 0.1;
fismat=genfis2(input,output, [.2 .3 .5 .3 .5 .5],[ ],[1.25 .5 .15 0]);
[fismat1, error1, stepsize, fismat 2, error 2] = anfis(trndata,fismat,[200 0.1 .9 1.1],[1 1 1 1], chkdata);
result=evalfis(input, fismat1);
result1=evalfis(input_chk,fismat1);
plot(output,result, 'bd',out_chk, result1, 'ks');
plot(result, 'DisplayName', 'result', 'YDataSource', 'result'); figure(gcf)
plot(result, 'DisplayName', 'result', 'YDataSource', 'result'); hold all; plot(output, 'DisplayName', 'output',
'YDataSource', 'output'); hold off; figure(gcf)
plot(result, output, 'DisplayName', 'output vs result', 'XDataSource', 'result', 'YDataSource', 'output');
figure(gcf)
scatter(result, output, 'DisplayName', 'output vs result', 'XDataSource', 'result', 'YDataSource', 'output');
figure(gcf)
plot(output, 'DisplayName', 'output', 'YDataSource', 'output'); hold all; plot(result, 'DisplayName', 'result',
'YDataSource', 'result'); hold off; figure(gcf)
surf(input); figure(gcf)
surfview(fismat)
plot(trndata, 'DisplayName', 'trndata', 'YDataSource', 'trndata'); figure(gcf)
```