Neuro-Fuzzy approach for the predictions of economic crisis

Eleftherios Giovanis

Abstract

In this paper we present the neuro-fuzzy technology for the prediction of economic crisis of USA economy. Our findings support ANFIS models to traditional discrete choice models of Probit and Logit, indicating that the last models are not very useful for forecasting purposes. We have developed a MATLAB routine to show how ANFIS procedure works and it is provided for replications, further research applications and experiments, for modifications, expansions and improvements. We propose the use of both models, because with discrete choice models we can examine and investigate the effects of the inputs or the independent variables, while we can simultaneously use ANFIS for forecasting purposes. The wise option and the most appropriate scientific action is to combine both models and not taking only one of them.

Keywords: Economic crisis; ANFIS; Neuro-Fuzzy, fuzzy rules; triangle function; Gaussian function; Generalized Bell function forecasting; discrete choice models; Logit; Probit; economy of USA; MATLAB

1.Introduction

The economic facts and the financial crisis in the last three years have led to doubts among the economic society and governments of which methodology must be followed and what regulations should be set up. The subprime mortgage crisis which took place in USA and became apparent in 2007 has led to great weakness in the financial system and the financial industry regulation. Many economists claim that mathematical models used are responsible for the failure prediction, but nothing is done about that. There are opinions that economics have to do with probabilities and not with possibilities or membership grades. Why not? The human behavior is not random but there are specific cause and consequences as in physics. The fact that we do not understand them or they are not clear this has nothing to do with probabilities, but has to do with the fact that the situations, our decisions or actions are fuzzy and imprecise. Even with artificial intelligence we can get biases, but we have to test them before we rush to judge it.

2.Literature review

Various approaches have been developed and applied in financial, banking and currency crisis. One of these approaches is the application of Logit and Probit models (Eichengreen and Rose,1998; Demirguc-Kunt and Detragrache ,1998; Frankel and Rose, 1996) among many others. Another approach which has been used in the crisis prediction is the noise-to-ratio model (Kaminsky and Reinhart, 1996; Kaminsky *et al*., 1998). This approach is used in order to identify variables as strong potential indicators of crisis. Since 1990 new approaches have been introduced in economics and finance, like neural networks, fuzzy logic and genetic algorithms. We propose the scientific findings and methods of artificial intelligence because most studies have found superior results, especially in stock prediction, economic data prediction, than the common Logit models, Multiple Discriminant Analysis, Autoregressive Conditional Heteroskedasticity (ARCH), Moving Average method among others (Coats and Fant, 1993; Zhang *et al.,* 1999; Nachev and Stoyanov, 2007) among many others.

A very few number of researches have been written employing neuro-fuzzy approaches for economic and financial crises prediction. Most of them found superior results supporting ANFIS approach and suggesting as an efficient alternative tool against the traditional econometric procedures as discrete choice models noise-to-ratio among others. For this reason we examine Neuro-Fuzzy system using two symmetric membership functions, the triangular and Gaussian, in order to test if this approach is superior or not to discrete choice Logit and Probit models.

3. Methodology

3.1 Binary Logit and Probit Regressions

In this section we provide a brief description of the binary Logit and Probit models estimation. The logistic distribution is defined as (Greene, 2008):

$$\Pr ob(Y=1\mid x)=\frac{e^{x'\beta}}{1+e^{x'\beta}}=\Phi(x'\beta) \qquad (1)$$

The marginal partial effects of explanatory variables are given by:

$$\frac{\partial \mathrm{E}[y\mid x)}{\partial x}=\Phi(x'\beta)[1-\Phi(x'\beta)]\beta \qquad (2)$$

The logistic regression analyzes the binomial distributed data and it is:

$$Y_i \sim B(n_i,P_i),\ \ i=1,2,......,n \qquad (3)$$

, where n_i denotes the number of Bernoulli trials and are known, while p_i denotes the probabilities of success, which are unknown. The model proposes for each i a set of explanatory variables and the model can take the following form.

$$P_i = E(\frac{Y_i}{n_i} \mid x_i) \tag{4}$$

Next the unknown probabilities are modeled as linear function of variables x_i.

$$y_i = \ln(\frac{p_i}{1-p_i}) = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \ldots.. + \beta_k x_{k,i} \tag{5}$$

Logit model can be written a general form regression as:

$$y = \alpha + \sum_{i=1}^{n} \beta_i x_i + \varepsilon \tag{6}$$

, where variable y is a binary dummy variable taking value 1 if the economy is on crisis or economic recession period and value zero otherwise (no crisis period), x_i indicates the explanatory variables, α is the constant, β_i are the regression estimators.

Next we present the Probit regression (Greene, 2008; Wooldridge, 2006). More specifically is defined as:

$$\Phi^{-1}(\mathrm{p_i}) = \mathrm{Z} = \alpha + \sum_{i=1}^{n} \beta_i x_i + \varepsilon \tag{7}$$

, where $\Phi^{-1}(p_i)$ is the inverse cumulative distribution function (CDF) of the standard normal, a, β_i and x_i are defined as in (6). Also it can be written as:

$$\Pr(\mathrm{y} = 1 \mid \mathrm{x}) = \Phi(\mathrm{x_i} \beta_i) \tag{8}$$

The inverse cumulative distribution function (CDF) is

$$p = \Phi(\mathrm{Z}) = \int_{-\infty}^{\mathrm{Z}} \frac{1}{\sqrt{2\pi}} \exp(\text{-u}^2/2)\mathrm{du} \tag{9}$$

The Log-Likelihood function for Probit is:

$$\ln L = \sum w_i \ln \Phi(\mathrm{x_i} \beta_i) + \sum w_i \ln(1 - \Phi(\mathrm{x_i} \beta_i)) \tag{10}$$

The prediction or the classification percentage is done based on the estimated coefficients from the in-sample period each time using as the cut-off point the value of 0.5. For the forecasting and the classification performance of the binary Logit and Probit models is:

If $y^* > 0.5$, then the economy is on crisis or economic recession period

If $y^* \leq 0.5$, then the economy is on no crisis or economic recovery period.

Variable y^* denotes the predicted values.

3.2 Adaptive network-based fuzzy inference system (ANFIS)

Jang (1993) and Jang and Sun (1995) introduced the adaptive network-based fuzzy inference system (ANFIS). This system makes use of a hybrid learning rule to optimize the fuzzy system parameters of a first order Sugeno system.

An ANFIS architecture with two inputs for example is consisted of two trainable parameter sets, the antecedent membership function parameters and the polynomial parameters *p,q,r*, also called the consequent parameters. The ANFIS training paradigm uses a gradient descent algorithm to optimize the antecedent parameters and a least squares algorithm to solve for the consequent parameters. Because it uses two very different algorithms to reduce the error, the training rule is called a hybrid. The consequent parameters are updated first using a least squares algorithm and the antecedent parameters are then updated by backpropagating the errors that still exist. The ANFIS architecture consists of five layers with the output of the nodes in each respective layer represented by O_i^l, where i is the i^{th} node of layer l.

We examine two cases one with two inputs and the there with three inputs. More inputs are possible to be obtained but it is not necessary, because the forecasts are very efficient. The inputs we take in our fuzzy system are the gross domestic product growth rate, the unemployment rate and the industrial production percentage change or rate. For each input three linguistic terms are incorporated and are $\{$low,medium,high$\}$.

We describe the steps of ANFIS with three inputs as it is more complicate and in a similar procedure we can follow the process with two inputs. Basically, there are two types of fuzzy set operation that are usually used in the antecedent rule, which are *AND* and *OR*. Mathematically, the *AND* operator can be realized using *Min* or *Product* operation while *OR* can be realized using *Max* or *Algebraic* sum operator. There are 3 inputs with three linguistic terms in each one so there will be 3^3=27 fuzzy rules. Also each rule has 3 parameters plus the constant hence there will be 4*27=108 parameters. The 27 rules are presented in table 1.

Table 1. Fuzzy rules

Number of rules	If GDP	and if unemployment rate	and if industrial production rate	then
1	LOW	LOW	LOW	$p_1x_1+q_1x_2+s_1x_3+r_1$
2	LOW	LOW	MEDIUM	$p_2x_1+q_2x_2+s_2x_3+r_2$
3	LOW	LOW	HIGH	$p_3x_1+q_3x_2+s_3x_3+r_3$
4	LOW	MEDIUM	LOW	$p_4x_1+q_4x_2+s_4x_3+r_4$
5	LOW	MEDIUM	MEDIUM	$p_5x_1+q_5x_2+s_5x_3+r_5$
6	LOW	MEDIUM	HIGH	$p_6x_1+q_6x_2+s_6x_3+r_6$
7	LOW	HIGH	LOW	$p_7x_1+q_7x_2+s_7x_3+r_7$
8	LOW	HIGH	MEDIUM	$p_8x_1+q_8x_2+s_8x_3+r_8$
9	LOW	HIGH	HIGH	$p_9x_1+q_9x_2+s_9x_3+r_9$
10	MEDIUM	LOW	LOW	$p_{10}x_1+q_{10}x_2+s_{10}x_3+r_{10}$
11	MEDIUM	LOW	MEDIUM	$p_{11}x_1+q_{11}x_2+s_{11}x_3+r_{11}$
12	MEDIUM	LOW	HIGH	$p_{12}x_1+q_{12}x_2+s_{12}x_3+r_{12}$
13	MEDIUM	MEDIUM	LOW	$p_{13}x_1+q_{13}x_2+s_{13}x_3+r_{13}$
14	MEDIUM	MEDIUM	MEDIUM	$p_{14}x_1+q_{14}x_2+s_{14}x_3+r_{14}$
15	MEDIUM	MEDIUM	HIGH	$p_{15}x_1+q_{15}x_2+s_{15}x_3+r_{15}$
16	MEDIUM	HIGH	LOW	$p_{16}x_1+q_{16}x_2+s_{16}x_3+r_{16}$
17	MEDIUM	HIGH	MEDIUM	$p_{17}x_1+q_{17}x_2+s_{17}x_3+r_{17}$
18	MEDIUM	HIGH	HIGH	$p_{18}x_1+q_{18}x_2+s_{18}x_3+r_{18}$
19	HIGH	LOW	LOW	$p_{19}x_1+q_{19}x_2+s_{19}x_3+r_{19}$
20	HIGH	LOW	MEDIUM	$p_{20}x_1+q_{20}x_2+s_{20}x_3+r_{20}$
21	HIGH	LOW	HIGH	$p_{21}x_1+q_{21}x_2+s_{21}x_3+r_{21}$
22	HIGH	MEDIUM	LOW	$p_{22}x_1+q_{22}x_2+s_{22}x_3+r_{22}$
23	HIGH	MEDIUM	MEDIUM	$p_{23}x_1+q_{23}x_2+s_{23}x_3+r_{23}$
24	HIGH	MEDIUM	HIGH	$p_{24}x_1+q_{24}x_2+s_{24}x_3+r_{24}$
25	HIGH	HIGH	LOW	$p_{25}x_1+q_{25}x_2+s_{25}x_3+r_{25}$
26	HIGH	HIGH	MEDIUM	$p_{26}x_1+q_{26}x_2+s_{26}x_3+r_{26}$
27	HIGH	HIGH	HIGH	$p_{27}x_1+q_{27}x_2+s_{27}x_3+r_{27}$

In the case of two inputs we have 3^2=9 fuzzy rules and each rule has 3 parameters plus the constant hence there will be 4*9=36 parameters. In the case of both two and three inputs we use *AND* operator, while in the first case (two inputs) we take *min* operator and in the second case (three inputs) we take the *product* operator.

In the first layer we generate the membership grades

$$O_i^1 = \mu_{A_i}(x_1),\ \mu_{B_i}(x_2),\ \mu_C(x_3) \qquad (11)$$

, where x_1, x_2 and x_3 are the inputs. In layer 2 we generate the firing strengths or weights

$$O_i^2 = w_i = \prod_{j=1}^{m} (\mu_{A_i}(x_1), \mu_{B_i}(x_2), \mu_{C_i}(x_3)) =$$
$$andmethod\,(\mu_{A_i}(x_1), \mu_{B_i}(x_2), \mu_{C_i}(x_3)) \tag{12}$$
$$= product\,(\mu_{A_i}(x_1) * \mu_{B_i}(x_2) * \mu_{C_i}(x_3))$$

In layer 2 we use the *AND* relation so we take the *product* operator. In layer 3 we normalize the firing strengths. Because we have twenty seven rules will be:

$$O_i^3 = \overline{w_i} = \frac{w_i}{w_1 + w_2 + \ldots + w_{26} + w_{27}} \tag{13}$$

In layer 4 we **c**alculate rule outputs based on the consequent parameters.

$$O_i^{\,4} = y_i = \overline{w_i} f = \overline{w_i}(p_i x_1 + q_i x_2 + s_i x_3 + r_i) \tag{14}$$

In layer 5 we take the sum all the inputs from layer 4

$$O_i^{\,5} = \sum_i \overline{w_i} f_i = \frac{\sum_i \overline{w_i} f}{\sum_i \overline{w_i}} \tag{15}$$

In the last layer the consequent parameters can be solved for using a least square algorithm as:

$$Y = X \cdot \theta \tag{16}$$

, where X is the matrix of inputs and θ is a vector of unknown parameters as:

$$\theta = \left[p_1, q_1, s_1, r_1, p_2, q_2, s_2, r_2, \ldots., p_{27}, q_{27}, s_{27}, r_{27}\right]^T \tag{17}$$

, where *T* indicates the transpose. Because the normal least square method leads to singular inverted matrix we use the singular value decomposition (SVD) with Moore-Penrose pseudoinverse of matrix (Petrou and Bosdogianni, 2000; Moore, 1920; Penrose, 1955). For the first layer and relation (12) we use the triangular, Gaussian and Generalized Bell membership functions. The symmetrical triangular function is defined as:

$$\mu_{ij}(x_j; a_{ij}, b_{ij}) = \left\{ \begin{array}{l} 1 - \dfrac{|x_j - a_{ij}|}{b_{ij}/2}, if\,|x_j - a_{ij}| \le \dfrac{b_{ij}}{2} \\ 0, otherwise \end{array} \right\} \tag{18}$$

, where α_{ij} is the peak or center parameter and b_{ij} is the spread or support parameter. The symmetrical Gaussian membership function is defined as:

$$\mu_{ij}(x_j; c_{ij}, \sigma_{ij}) = \exp\left(- \frac{(x_j - c_{ij})^2}{2\sigma^2_{ij}} \right) \tag{19}$$

, where c_{ij} is the center parameter and σ_{ij} is the spread parameter. The last function we examine is the Generalized Bell functions defined as:

$$\mu_{ij}(x_{ij}; a_{ij}, b_{ij}, c_{ij}) = \exp\left(- \left(\frac{x_{ij} - c_{ij}}{a_{ij}} \right)^{2b_{ij}} \right) \tag{20}$$

, where c locates the center of the curve and parameters a and b vary the width of the curve. All the parameters of fuzzy membership functions are also called premise parameters. In order to find the optimized antecedent parameters we use backpropagation algorithm. The premise parameters update for the triangle membership function is:

$$a_{ij}(n+1) = a_{ij}(n) - \eta_\alpha \cdot \frac{\partial E}{\partial a_{ij}} \tag{21}$$

,where η_a is the learning rate for the parameter α_{ij} and E is the error function which is:

$$E = \frac{1}{2}(y - y^t)^2 \tag{22}$$

, where y^t is the target-actual and y is ANFIS output variable. The chain rule in order to calculate the derivatives used to update the membership function parameters are

:

$$\frac{\partial E}{\partial a_{ij}} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial y_i} \cdot \frac{\partial y_i}{\partial w_i} \cdot \frac{\partial w_i}{\partial \mu_{ij}} \frac{\partial \mu_{ij}}{\partial a_{ij}} \tag{23}$$

The partial derivatives for two inputs are derived below:

$$\frac{\partial E}{\partial y_{ij}} = y - y^t = e \tag{24}$$

For the output is

$y = \sum_{i=1}^{n} y_i$, hence it will be $\frac{\partial y}{\partial y_i} = 1$ (25)

$y_i = \frac{w_i}{\sum_{i=1}^{n} w_i}(p_i x_1 + q_i x_2 + r_i)$, hence it will be $\frac{\partial y_i}{\partial w_i} = \frac{(p_i x_1 + q_i x_2 + r_i) - y}{\sum_{i=1}^{n} w_i}$ (26)

$$w_i = \prod_{j=1}^{m} \mu_{A_{ji}} \quad \text{, hence it will be } \frac{\partial w_i}{\partial \mu_{ji}} = \frac{w_i}{\mu_{ji}} \tag{27}$$

The last partial derivative, $\frac{\partial \mu_{ij}}{\partial a_{ij}}$, depends on the membership function we examine. The update equations, (eg. for triangle function) for antecedent a_{ij}, b_{ij} parameters are:

$$\alpha_{ij}(n+1) = \alpha_{ij}(n) - \eta_{\alpha} \cdot e \frac{(p_i x_1 + q_i x_2 + r_i) - y}{\sum_{i=1}^{n} w_i} \cdot \frac{w_i}{\mu_{ij}(x_j)} \cdot \frac{\partial \mu_{ij}}{\partial a_{ij}} \tag{28}$$

$$b_{ij}(n+1) = b_{ij}(n) - \eta_{\alpha} \cdot e \frac{(p_i x_1 + q_i x_2 + r_i) - y}{\sum_{i=1}^{n} w_i} \cdot \frac{w_i}{\mu_{ij}(x_j)} \cdot \frac{\partial \mu_{ij}}{\partial b_{ij}} \tag{29}$$

The next step is to set up the initial values for center and bases parameters. The first option is to set up specific values. The second option is to set up the initial values based on mean and standard deviation of specific intervals. We follow the first and second option respectively for triangle and Gaussian functions, while it should be noticed that the forecasts are slightly better and not significant different to those obtained by setting up the opposite options. In the case of triangle membership function the initial values for center parameters for the fuzzy set {low, medium, high} are for GDP {-0.01, 0.01, 0.03}, for unemployment rate {5,7,9} and for industrial production {-0.01, 0.01, 0.04} respectively. The initial values for bases are 0.025, 2 and 0.035 for GDP, unemployment rate and industrial production respectively. In the case of Gaussian functions we set up low for negative values of GDP, medium for values of GDP between 0-0.02 and for high for values higher than 0.02. In the case of unemployment rate we define low for values lower than 5, medium for values ranging in the interval 5-7 and high for values higher than 7. Finally, in the case of industrial production we set up the same values of GDP. So for example if we want to set up the initial values of low GDP and high unemployment we take the sample where GDP is ranging in the interval 0-0.02 and where unemployment is higher than 7. Then we take their average and standard deviations and we get the initial values for center ,*c*, and base ,*σ*, parameters respectively. Generally, Gaussian function works better with this option. The same procedure is followed for the parameters *c* generalized bell function, for mean values. For parameter *a* we follow a different procedure where the initial values are based on the following.

$$\alpha = \frac{range}{mfs * 2} \tag{30}$$

, where the *range* is the well known statistical measure and we take the range of data, while *mfs* is the number of membership functions. For parameters *b* the initial values are 1.1. In the case of all membership functions the learning rates for the center and bases parameters are set up at 0.1, while for the RHS at 0.5.

4.Data

We estimate the period 1950-2005 and we examine the in-sample forecasting performance. Then we apply all the models to compare their predicting performance for the period 2006-2009. The data source is the *Federal Reserve Bank of St. Louis* and the *National Bureau of Economic Research.* The choice of variables is based on various research papers and studies (Demirguc-Kunt and Detragrache, 1998; Eichengreen and Rose,1998; Glick and Moreno, 1999), as also based on National Bureau of Economic Research (NBER), which defines real GDP, real income, unemployment rate, industrial production and retail sales as the most important factors defining the economic activity in US economy. Moreover we try all the candidate variables and we choose the most significant. Specifically we use the same variables in Logit and Probit regressions.

Table 2. Variables used in estimated regressions

industrial production	unemployment rate
inflation rate	total investments at all commercial banks
total borrowings of depository institutions from federal reserve system	oil prices
interest rates of 3-monthly treasury bills	bank prime loan rate
public debt	balance of accounts
total loans at all commercial banks	gdp growth

We used these ones which were found statistically significant and that improve the forecasting performance. These are the industrial production, bank prime loan rate, balance of accounts and the gross domestic product growth rate. We do not present the estimation results, which can be easily replicated, because we are interesting on the forecasting performance of the models we examine in the current study.

5.Empirical results

In table 3 we present the correctly percentage or the forecasts of Logit regression for the in-sample period 1950-2005, while in the table 4 the forecasts of the same model for the out–of sample period 2006-2009 are provided. We observe that the forecasting performance in the in-sample period is relatively significant, while in the out-of sample is too poor. More specifically, with Logit model we successfully predict at 62.50 per cent the crisis periods and 94.56 the no crisis periods, while the respective percentages in the out-of-sample period are 70.00 and 66.67. In tables 5 and 6 we present the forecasts of Probit regression in the in-sample and out-of sample period respectively. In the out-of sample period 2006-2009 Probit predicts 95.74 per cent correct the no crisis periods and 65.00 per cent the crisis periods, which is slightly higher to the predicted percentage of Logit models. In the out-of-sample period both Logit and Probit models predict at 70.00 percent correct the crisis periods.

Table 3 Prediction results of binary Logit regression for in-sample period 1950-2005

Crisis	25	15	62.50
No Crisis	10	174	94.56
Overall percentage			88.83

Table 4 Prediction results of binary Logit regression for out-of-sample period 2006-2009

Crisis	7	3	70.00
No Crisis	2	4	66.67
Overall percentage			68.75

.

Table 5 Prediction results of binary Probit regression for in-sample period 1950-2005

Crisis	26	14	65.00
No Crisis	8	176	95.65
Overall percentage			90.17

Table 6 Prediction results of binary Probit regression for in-sample period 2006-2009

Crisis	7	3	70.00
No Crisis	1	5	83.33
Overall percentage			75.00

In tables 7 and 8 we present the ANFIS with triangle function in the in-sample and out-of-sample periods respectively. In the first period the overall forecasting performance is lower to than Logit and Probit, because of the lower forecasting performance of no crisis periods. In both periods, ANFIS outperforms significant the traditional discrete choice models concerning the crisis periods. Additionally, the overall performance of both ANFIS models is much more superior to Logit and Probit as we can predict crisis periods at 100.00 and 80.00 per cent success with triangle and Gaussian functions respectively, while the respective percentage in Logit and Probit is only 70.00.

Table 7 Prediction results of ANFIS with triangle membership function for in-sample period 1950-2005

Crisis	35	5	87.50
No Crisis	26	158	85.86
Overall percentage			86.16

Table 8 Prediction results of ANFIS with triangle membership function with three inputs for out-of-sample period 2006-2009

Crisis	10	0	100.00
No Crisis	1	5	83.33
Overall percentage			93.75

Table 9 Prediction results of ANFIS with Gaussian membership function with three inputs for in-sample period 1950-2005

Crisis	40	0	100.00
No Crisis	26	158	85.86
Overall percentage			88.39

Table 10 Prediction results of ANFIS with Gaussian membership function with three inputs for out-of-sample period 2006-2009

Crisis	8	2	80.00
No Crisis	2	4	66.67
Overall percentage			75.00

Finally in tables 11-14 we present the forecasts for ANFIS with generalized bell and Gaussian membership functions and with two inputs, the gross domestic product growth rate and the unemployment rate. The results support the ANFIS procedure. Additionally, we observe that the forecssting performance with Gaussian function is improved using two inputs concerning the out-of-sample period. For this reason ANFIS can work very well with fewer inputs.

Table 11 Prediction results of ANFIS with Generalized Bell membership function and two inputs (GDP and unemployment) for in-sample period 1950-2005

Crisis	35	5	87.50
No Crisis	25	159	86.41
Overall percentage			86.60

Table 12 Prediction results of ANFIS with Generalized Bell membership function and two inputs(GDP and unemployment) for out-of-sample period 2006-2009

Crisis	9	1	90.00
No Crisis	1	5	83.33
Overall percentage			87.50

Table 13 Prediction results of ANFIS with Gaussian membership function and two inputs(GDP and unemployment) for in-sample period 1950-2005

Crisis	38	2	95.00
No Crisis	38	146	79.34
Overall percentage			82.14

Table 14 Prediction results of ANFIS with Gaussian membership function and two inputs(GDP and unemployment) for out-of-sample period 2006-2009

Crisis	9	1	90.00
No Crisis	2	4	66.67
Overall percentage			81.25

Conclusions

The conclusion is that Neuro-fuzzy approach is a very promising tool for forecasting crisis periods and can be used also for bankruptcies forecasts, financial distress models, and almost in any modeling in economics and finance, because most of them can be fuzzified and we can set up rules. Additionally, more functions can be examined as the S-shape or the Z-shape functions, as also other training processes can be used, like genetic algorithms or Levenberg-Marqaurdt, instead to error backpropagation.

References

Coats, P.K., and Fant, F. L. (1993). Recognizing financial distress patterns using a neural network tool, *Financial Management,* Vol. 22, pp. 142-155.

Demirguc-kunt, A., and Detragiache, E. (1998). The Determinants of Banking Crises in Developing and Developed Countries. IMF Staff Papers, Vol. 45, No. 1, pp. 81-109

Eichengreen, B., and Rose, A.K. (1998). Staying Afloat When the Wind Shifts: External Factors and Emerging-Market Banking Crises. NBER Working Papers 6370. National Bureau of Economic Research, Cambridge, MA

Frankel, J., and Rose, A.K. (1996). Currency Crashes in emerging Markets: An Empirical Treatment, International Finance Discussion Papers 534. Board of Governors of the Federal Reserve System. Washington. D.C

Greene, W.H. (2008). *Econometric Analysis*, Sixth Edition, Prentice Hall: New Jersey

Jang, J.-S.R. (1993). ANFIS: Adaptive-Network-based Fuzzy Inference Systems. *IEEE Transactions on Systems*, Man, and Cybernetics, Vol. 23, No. 3, pp. 665-685

Jang, J.-S. R. and Sun, C.-T. (1995). Neuro-fuzzy Modeling and Control. *Proceedings of the IEEE*, Vol. 83, No. 3, pp. 378-406, March

Kaminsky, L.G., and Reinhart, C. M. (1996). The Twin Crises: The Causes of Banking and Balance of Payments Problems. Federal Reserve Board Discussion Papers 544. Board of Governors of the Federal Reserve System.Washington. D.C.

Kaminsky, L.G., Lizondo, S., and Reinhart, C.M. (1998). Leading Indicators of Currency Crises, IMF Staff Papers, Vol. 45, No. 1, pp. 1-48.

Moore, E. H. (1920). On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society,* Vol. 26, pp. 394–395

Nachev. A, and Stoyanov, B. (2007). A Default ARTMAP Neural Networks for Financial Diagnosis. Proceedings of the 2007 International Conference on Data Mining, DMIN 2007, June 25-28, Las Vegas, Nevada, USA. CSREA.

Petrou, M. and Bosdogianni, P. (2000). *Image Processing: The Fundamentals*, John Wiley

Penrose, R. (1955). A generalized inverse for matrices. *Proceedings of the Cambridge Philosophical Society*, Vol. 51, pp. 406–413

Zhang, G. Hu, Y., and Patuwo, E.B. (1999). Artificial neural networks in bankruptcy prediction: General framework and cross-validation analysis, *European Journal of Operation Research,* Vol. 116, pp. 16-32.

Appendix

MATL AB routine for ANFIS with three inputs

```
function []=neuro_fuzzy (x,y)
 %or
 %function []=neuro_fuzzy (x,y)

clear all;
load file.mat
% data is consisted by 4 columns, the first three concern the input
% variables and the last column the output variable

y=data(1:end-16,end)
x=data(1:end-16,1:3)
% y is the output dummy variable
% x is the matrix of inputs


em=2     % Select the operator, 1 for min which is for AND operator, 2 for
product, which is for
         % AND operator and 3 for for max, which is for OR operator

opt=2   % option 1 is for specific center and bases values of your choice
         % option 2 takes the center and bases values based on mean and
         % standard deviation respectively.

memb=3   % 1 for triangular membership function, 2 for Gaussian

% Set up the learning rates of your choice for center, bases and parameters
r
lr_center_gdp=0.05
lr_center_une=0.05
lr_center_indpro=0.05
lr_base_gdp=0.05
lr_base_une=0.05
lr_base_indpro=0.05
lr_beta_gdp=0.005
lr_beta_une=0.005
lr_beta_indpro=0.005
lr_r=0.5

% Take the inputs
gdp=x(:,1)
une=x(:,2)
indpro=x(:,3)

% Take the dimensions of input-output data
[t nj]=size(y)
[nk ni]=size(x)



%First option take specific values of your choice
if opt==1
% Center values for GDP
c_1=-0.01
c_2=0.01
```

```matlab
c_3=0.03

%Bases values for GDP
if memb==1
sigma_1=0.04
sigma_2=0.05
sigma_3=0.05

elseif memb==2
sigma_1=0.025
sigma_2=0.025
sigma_3=0.025

elseif memb==3
sigma1=(max(gdp)-min(gdp))/18

sigma_1=sigma1
sigma_2=sigma1
sigma_3=sigma1

end

% Center values for Unemployment rate
ce_1=5
ce_2=7
ce_3=9
%Bases values for Unemployment rate

if memb==1
sigmae_1=3
sigmae_2=3
sigmae_3=3
elseif memb==2
sigmae_1=2
sigmae_2=2
sigmae_3=2
elseif memb==3
sigma2=(max(une)-min(une))/18
sigmae_1=sigma2
sigmae_2=sigma2
sigmae_3=sigma2

end
% Center values for Industrial production
cindpro_1=-0.02
cindpro_2=0.01
cindpro_3=0.04

%Bases values for Industrial production
if memb==1
sigma_indpro_1=0.07
sigma_indpro_2=0.07
sigma_indpro_3=0.07
elseif memb==2
sigma_indpro_1=0.035
sigma_indpro_2=0.035
sigma_indpro_3=0.035
elseif memb==3
sigma3=(max(indpro)-min(indpro))/18
sigma_indpro_1=sigma3
```

```
sigma_indpro_2=sigma3
sigma_indpro_3=sigma3
end

b1_gdp=1.1
b2_gdp=1.1
b3_gdp=1.1

b1_une=1.1
b2_une=1.1
b3_une=1.1

b1_indpro=1.1
b2_indpro=1.1
b3_indpro=1.1
% Take center and bases values based on mean and standard deviation
respectively
elseif opt==2
% Values for GDP
y_high=find(gdp>=0.02)
y_medium=find(gdp>=0 & gdp<=0.02)
y_low=find(gdp<=0 )

% find the values correspond to linguistic terms
y_high=gdp(y_high)
y_medium=gdp(y_medium)
y_low=gdp(y_low)

% Find the mean and simga (standard deviation)
c_1=mean(y_low)
c_2=mean(y_medium)
c_3=mean(y_high)

if memb==1
sigma_1=std(y_low)*2
sigma_2=std(y_medium)*2
sigma_3=std(y_high)*2
elseif memb==2
sigma_1=std(y_low)
sigma_2=std(y_medium)
sigma_3=std(y_high)
elseif memb==3
sigma1=(max(gdp)-min(gdp))/18

sigma_1=sigma1
sigma_2=sigma1
sigma_3=sigma1
end


% In the case you get zero sigma you can write the following. The same is
% followed for the sigma in the ithe inputs
%if sigma_1==0 | sigma_2==0 |sigma_3==0
 %   sigma_1==0.025
%sigma_2==0.025
%sigma_3==0.025
%end
```

```
% Values for Unemployment rate
e_high=find(une>=6)
e_medium=find(une>=4.5 & une<=6)
e_low=find(une<=4.5)


% find the values correspond to linguistic terms
e_high=une(e_high)
e_medium=une(e_medium)
e_low=une(e_low)

% Find the mean and simga (standard deviation)
ce_1=mean(e_high)
ce_2=mean(e_medium)
ce_3=mean(e_low)

if memb==1
sigmae_1=std(e_high)*2
sigmae_2=std(e_medium)*2
sigmae_3=std(e_low)*2
elseif memb==2
sigmae_1=std(e_high)
sigmae_2=std(e_medium)
sigmae_3=std(e_low)
elseif memb==3
sigma2=(max(une)-min(une))/18
sigmae_1=sigma2
sigmae_2=sigma2
sigmae_3=sigma2
end
% Values for Industrial production
ind_high=find(indpro>=0.02)
ind_medium=find(indpro>=0 & indpro<=0.02)
ind_low=find(indpro<=0 )


% find the values correspond to linguistic terms
ind_high=indpro(ind_high)
ind_medium=indpro(ind_medium)
ind_low=indpro(ind_low)


% Find the mean and simga (standard deviation)
cindpro_1=mean(ind_low)
cindpro_2=mean(ind_medium)
cindpro_3=mean(ind_high)

if memb==1
sigma_indpro_1=std(ind_low)*2
sigma_indpro_2=std(ind_medium)*2
sigma_indpro_3=std(ind_high)*2
elseif memb==2
sigma_indpro_1=std(ind_low)
sigma_indpro_2=std(ind_medium)
sigma_indpro_3=std(ind_high)
elseif memb==3
sigma3=(max(indpro)-min(indpro))/18
sigma_indpro_1=sigma3
sigma_indpro_2=sigma3
sigma_indpro_3=sigma3
```

```
end

b1_gdp=1.1
b2_gdp=1.1
b3_gdp=1.1

b1_une=1.1
b2_une=1.1
b3_une=1.1

b1_indpro=1.1
b2_indpro=1.1
b3_indpro=1.1
end

if memb==1
% Take memebership degrees-grades
mf_gdp_low=trimf(gdp,[-sigma_1/2+c_1 c_1 sigma_1/2+c_1 ]);     % LOW
mf_gdp_medium=trimf(gdp,[-sigma_2/2+c_2 c_2 sigma_2/2+c_2]);   % MEDIUM
mf_gdp_high=trimf(gdp,[-sigma_3/2+c_3 c_3 sigma_3/2+c_3]);     % HIGH


mf_une_low=trimf(une,[-sigmae_1/2+ce_1 ce_1 sigmae_1/2+ce_1]);      % LOW
mf_une_medium=trimf(une,[-sigmae_2/2+ce_2 ce_2 sigmae_2/2+ce_2 ]); % MEDIUM
mf_une_high=trimf(une,[-sigmae_3/2+ce_3 ce_3 sigmae_3/2+ce_3 ]);    % HIGH

mf_indpro_low=trimf(indpro,[-sigma_indpro_1/2+cindpro_1 cindpro_1 ...
    sigma_indpro_1/2+cindpro_1 ]); % LOW
mf_indpro_medium=trimf(indpro,[-sigma_indpro_2/2+cindpro_2 cindpro_2...
    sigma_indpro_2/2+cindpro_2 ]); % MEDIUM
mf_indpro_high=trimf(indpro,[-sigma_indpro_3/2+cindpro_3 cindpro_3 ...
    sigma_indpro_3/2+cindpro_3 ]); % HIGH


elseif memb==2
mf_gdp_low=gaussmf(gdp,[sigma_1 c_1  ]);          % LOW
mf_gdp_medium=gaussmf(gdp,[sigma_2 c_2  ]);       % MEDIUM
mf_gdp_high=gaussmf(gdp,[sigma_3 c_3 ]);          % HIGH




mf_une_low=gaussmf(une,[sigmae_1 ce_1  ]);       % LOW
mf_une_medium=gaussmf(une,[sigmae_2 ce_2  ]);    % MEDIUM
mf_une_high=gaussmf(une,[sigmae_3 ce_3  ]);      % HIGH

mf_indpro_low=gaussmf(indpro,[sigma_indpro_1 cindpro_1  ]);      % LOW
mf_indpro_medium=gaussmf(indpro,[sigma_indpro_2 cindpro_2  ]);   % MEDIUM
mf_indpro_high=gaussmf(indpro,[sigma_indpro_3 cindpro_3  ]);     % HIGH

elseif memb==3

mf_gdp_low=gbell_mf(gdp,[sigma_1 b1_gdp c_1   ]);          % LOW
mf_gdp_medium=gbell_mf(gdp,[sigma_2 b2_gdp c_2  ]);       % MEDIUM
mf_gdp_high=gbell_mf(gdp,[sigma_3 b3_gdp c_3  ]);          % HIGH



mf_une_low=gbell_mf(une,[sigmae_1 b1_une ce_1  ]);       % LOW
```

```
mf_une_medium=gbell_mf(une,[sigmae_2 b2_une ce_2  ]);     % MEDIUM
mf_une_high=gbell_mf(une,[sigmae_3 b3_une ce_3  ]);       % HIGH

mf_indpro_low=gbell_mf(indpro,[sigma_indpro_1 b1_indpro cindpro_1   ]);
% LOW
mf_indpro_medium=gbell_mf(indpro,[sigma_indpro_2 b2_indpro cindpro_2  ]);
% MEDIUM
mf_indpro_high=gbell_mf(indpro,[sigma_indpro_3 b3_indpro cindpro_3  ]);
% HIGH

end

% Take the fuzzy rules
if em==1             % Min operator for AND rule
w1=min([mf_gdp_low mf_une_low mf_indpro_low]')           % LOW-LOW-LOW
w2=min([mf_gdp_low mf_une_low mf_indpro_medium]')        % LOW-LOW-MEDIUM
w3=min([mf_gdp_low mf_une_low mf_indpro_high]')          % LOW-LOW-HIGH
w4=min([mf_gdp_low mf_une_medium mf_indpro_low]')        % LOW-MEDIUM-LOW
w5=min([mf_gdp_low mf_une_medium mf_indpro_medium]')     % LOW-MEDIUM-MEDIUM
w6=min([mf_gdp_low mf_une_medium mf_indpro_high]')       % LOW-MEDIUM-HIGH
w7=min([mf_gdp_low mf_une_high mf_indpro_low]')          % LOW-HIGH-LOW
w8=min([mf_gdp_low mf_une_high mf_indpro_medium]')       % LOW-HIGH-MEDIUM
w9=min([mf_gdp_low mf_une_high mf_indpro_high]')         % LOW-HIGH-HIGH
w10=min([mf_gdp_medium mf_une_low mf_indpro_low]')       % MEDIUM-LOW-LOW
w11=min([mf_gdp_medium mf_une_low mf_indpro_medium]')    % MEDIUM-LOW-MEDIUM
w12=min([mf_gdp_medium mf_une_low mf_indpro_high]')      % MEDIUM-LOW-HIGH
w13=min([mf_gdp_medium mf_une_medium mf_indpro_low]')    % MEDIUM-MEDIUM-LOW
w14=min([mf_gdp_medium mf_une_medium mf_indpro_medium]')   % MEDIUM-MEDIUM-
MEDIUM
w15=min([mf_gdp_medium mf_une_medium mf_indpro_high]')   % MEDIUM-MEDIUM-
HIGH
w16=min([mf_gdp_medium mf_une_high mf_indpro_low]')      % MEDIUM-HIGH-LOW
w17=min([mf_gdp_medium mf_une_high mf_indpro_medium]')   % MEDIUM-HIGH-
MEDIUM
w18=min([mf_gdp_medium mf_une_high mf_indpro_high]')     % MEDIUM-HIGH-HIGH
w19=min([mf_gdp_high mf_une_low mf_indpro_low]')         % HIGH-LOW-LOW
w20=min([mf_gdp_high mf_une_low mf_indpro_medium]')      % HIGH-LOW-MEDIUM
w21=min([mf_gdp_high mf_une_low mf_indpro_high]')        % HIGH-LOW-HIGH
w22=min([mf_gdp_high mf_une_medium mf_indpro_low]')      % HIGH-MEDIUM-LOW
w23=min([mf_gdp_high mf_une_medium mf_indpro_medium]')   % HIGH-MEDIUM-
MEDIUM
w24=min([mf_gdp_high mf_une_medium mf_indpro_high]')     % HIGH-MEDIUM-HIGH
w25=min([mf_gdp_high mf_une_high mf_indpro_low]')        % HIGH-HIGH-LOW
w26=min([mf_gdp_high mf_une_high mf_indpro_medium]')     % HIGH-HIGH-MEDIUM
w27=min([mf_gdp_high mf_une_high mf_indpro_high]')       % HIGH-HIGH-HIGH




elseif em==2      % product operator for AND rule
w1=(mf_gdp_low.*mf_une_low.*mf_indpro_low)'              % LOW-LOW-LOW
w2=(mf_gdp_low.*mf_une_low.*mf_indpro_medium)'           % LOW-LOW-MEDIUM
w3=(mf_gdp_low.*mf_une_low.*mf_indpro_high)'             % LOW-LOW-HIGH
w4=(mf_gdp_low.*mf_une_medium.*mf_indpro_low)'           % LOW-MEDIUM-LOW
w5=(mf_gdp_low.*mf_une_medium.*mf_indpro_medium)'        % LOW-MEDIUM-MEDIUM
w6=(mf_gdp_low.*mf_une_medium.*mf_indpro_high)'          % LOW-MEDIUM-HIGH
w7=(mf_gdp_low.*mf_une_high.*mf_indpro_low)'             % LOW-HIGH-LOW
w8=(mf_gdp_low.*mf_une_high.*mf_indpro_medium)'          % LOW-HIGH-MEDIUM
```

```
w9=(mf_gdp_low.*mf_une_high.*mf_indpro_high)'           % LOW-HIGH-HIGH
w10=(mf_gdp_medium.*mf_une_low.*mf_indpro_low)'         % MEDIUM-LOW-LOW
w11=(mf_gdp_medium.*mf_une_low.*mf_indpro_medium)'      % MEDIUM-LOW-MEDIUM
w12=(mf_gdp_medium.*mf_une_low.*mf_indpro_high)'        % MEDIUM-LOW-HIGH
w13=(mf_gdp_medium.*mf_une_medium.*mf_indpro_low)'      % MEDIUM-MEDIUM-LOW
w14=(mf_gdp_medium.*mf_une_medium.*mf_indpro_medium)'   % MEDIUM-MEDIUM-
MEDIUM
w15=(mf_gdp_medium.*mf_une_medium.*mf_indpro_high)'     % MEDIUM-MEDIUM-
HIGH
w16=(mf_gdp_medium.*mf_une_high.*mf_indpro_low)'        % MEDIUM-HIGH-LOW
w17=(mf_gdp_medium.*mf_une_high.*mf_indpro_medium)'     % MEDIUM-HIGH-
MEDIUM
w18=(mf_gdp_medium.*mf_une_high.*mf_indpro_high)'       % MEDIUM-HIGH-HIGH
w19=(mf_gdp_high.*mf_une_low.*mf_indpro_low)'           % HIGH-LOW-LOW
w20=(mf_gdp_high.*mf_une_low.*mf_indpro_medium)'        % HIGH-LOW-MEDIUM
w21=(mf_gdp_high.*mf_une_low.*mf_indpro_high)'          % HIGH-LOW-HIGH
w22=(mf_gdp_high.*mf_une_medium.*mf_indpro_low)'        % HIGH-MEDIUM-LOW
w23=(mf_gdp_high.*mf_une_medium.*mf_indpro_medium)'     % HIGH-MEDIUM-
MEDIUM
w24=(mf_gdp_high.*mf_une_medium.*mf_indpro_high)'       % HIGH-MEDIUM-HIGH
w25=(mf_gdp_high.*mf_une_high.*mf_indpro_low)'          % HIGH-HIGH-LOW
w26=(mf_gdp_high.*mf_une_high.*mf_indpro_medium)'       % HIGH-HIGH-MEDIUM
w27=(mf_gdp_high.*mf_une_high.*mf_indpro_high)'         % HIGH-HIGH-HIGH



elseif em==3      % max operator for OR rule
w1=max([mf_gdp_low mf_une_low mf_indpro_low]')              % LOW-LOW-LOW
w2=max([mf_gdp_low mf_une_low mf_indpro_medium]')           % LOW-LOW-
MEDIUM
w3=max([mf_gdp_low mf_une_low mf_indpro_high]')             % LOW-LOW-HIGH
w4=max([mf_gdp_low mf_une_medium mf_indpro_low]')           % LOW-MEDIUM-
LOW
w5=max([mf_gdp_low mf_une_medium mf_indpro_medium]')        % LOW-MEDIUM-
MEDIUM
w6=max([mf_gdp_low mf_une_medium mf_indpro_high]')          % LOW-MEDIUM-
HIGH
w7=max([mf_gdp_low mf_une_high mf_indpro_low]')             % LOW-HIGH-LOW
w8=max([mf_gdp_low mf_une_high mf_indpro_medium]')          % LOW-HIGH-
MEDIUM
w9=max([mf_gdp_low mf_une_high mf_indpro_high]')            % LOW-HIGH-HIGH
w10=max([mf_gdp_medium mf_une_low mf_indpro_low]')          % MEDIUM-LOW-
LOW
w11=max([mf_gdp_medium mf_une_low mf_indpro_medium]')       % MEDIUM-LOW-
MEDIUM
w12=max([mf_gdp_medium mf_une_low mf_indpro_high]')         % MEDIUM-LOW-
HIGH
w13=max([mf_gdp_medium mf_une_medium mf_indpro_low]')       % MEDIUM-
MEDIUM-LOW
w14=max([mf_gdp_medium mf_une_medium mf_indpro_medium]')    % MEDIUM-
MEDIUM-MEDIUM
w15=max([mf_gdp_medium mf_une_medium mf_indpro_high]')      % MEDIUM-
MEDIUM-HIGH
w16=max([mf_gdp_medium mf_une_high mf_indpro_low]')         % MEDIUM-HIGH-
LOW
w17=max([mf_gdp_medium mf_une_high mf_indpro_medium]')      % MEDIUM-HIGH-
MEDIUM
w18=max([mf_gdp_medium mf_une_high mf_indpro_high]')        % MEDIUM-HIGH-
HIGH
w19=max([mf_gdp_high mf_une_low mf_indpro_low]')            % HIGH-LOW-LOW
```

```
w20=max([mf_gdp_high mf_une_low mf_indpro_medium]')          % HIGH-LOW-
MEDIUM
w21=max([mf_gdp_high mf_une_low mf_indpro_high]')            % HIGH-LOW-HIGH
w22=max([mf_gdp_high mf_une_medium mf_indpro_low]')          % HIGH-MEDIUM-
LOW
w23=max([mf_gdp_high mf_une_medium mf_indpro_medium]')       % HIGH-MEDIUM-
MEDIUM
w24=max([mf_gdp_high mf_une_medium mf_indpro_high]')         % HIGH-MEDIUM-
HIGH
w25=max([mf_gdp_high mf_une_high mf_indpro_low]')            % HIGH-HIGH-LOW
w26=max([mf_gdp_high mf_une_high mf_indpro_medium]')         % HIGH-HIGH-
MEDIUM
w27=max([mf_gdp_high mf_une_high mf_indpro_high]')           % HIGH-HIGH-
HIGH
end



%Layer 3
for j=1:t
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& ...
w8(:,j)==0& w9(:,j)==0& w10(:,j)==0 & w11(:,j)==0 & w12(:,j)==0 &
w13(:,j)==0& w14(:,j)==0 &...
w16(:,j)==0 & w17(:,j)==0 & w18(:,j)==0 & w19(:,j)==0 & w20(:,j)==0&
w21(:,j)==0& w22(:,j)==0&...
w23(:,j)==0 & w24(:,j)==0 & w25(:,j)==0 & w26(:,j)==0 & w27(:,j)==0)

nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;...
nw9(:,j)=0;nw10(:,j)=0;nw11(:,j)=0;nw12(:,j)=0;nw13(:,j)=0;
nw14(:,j)=0;nw15(:,j)=0;nw16(:,j)=0;...
nw17(:,j)=0;nw18(:,j)=0;nw19(:,j)=0;nw20(:,j)=0;
nw21(:,j)=0;nw22(:,j)=0;nw23(:,j)=0;nw24(:,j)=0;...
nw25(:,j)=0;nw26(:,j)=0;nw27(:,j)=0;

else
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));
```

```
nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw10(:,j)=w10(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw11(:,j)=w11(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw12(:,j)=w12(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw13(:,j)=w13(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw14(:,j)=w14(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
```

```
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw15(:,j)=w15(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw16(:,j)=w16(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j
)+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw17(:,j)=w17(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw18(:,j)=w18(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw19(:,j)=w19(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw20(:,j)=w20(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw21(:,j)=w21(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw22(:,j)=w22(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw23(:,j)=w23(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw24(:,j)=w24(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
```

```
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw25(:,j)=w25(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw26(:,j)=w26(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw27(:,j)=w27(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));


end

end


% Layer 4-5
X_train=[nw1.*gdp';nw2.*gdp';nw3.*gdp';nw4.*gdp';nw5.*gdp';nw6.*gdp';nw7.*g
dp';nw8.*gdp';nw9.*gdp';...
nw10.*gdp';nw11.*gdp';nw12.*gdp';nw13.*gdp';nw14.*gdp';
nw15.*gdp';nw16.*gdp';nw17.*gdp';nw18.*gdp';...
nw19.*gdp';nw20.*gdp';nw21.*gdp';nw22.*gdp';nw23.*gdp';
nw24.*gdp';nw25.*gdp';nw26.*gdp';nw27.*gdp';...
nw1.*une';nw2.*une';nw3.*une';nw4.*une';nw5.*une';nw6.*une';nw7.*une';nw8.*
une';nw9.*une';...
nw10.*une';nw11.*une';nw12.*une';nw13.*une';nw14.*une';
nw15.*une';nw16.*une';nw17.*une';nw18.*une';...
nw19.*une';nw20.*une';nw21.*une';nw22.*une';nw23.*une';
nw24.*une';nw25.*une';nw26.*une';nw27.*une';...
nw1.*indpro';nw2.*indpro';nw3.*indpro';nw4.*indpro';nw5.*indpro';nw6.*indpr
o';nw7.*indpro';...
nw8.*indpro';nw9.*indpro';nw10.*indpro';nw11.*indpro';nw12.*indpro';nw13.*i
ndpro';nw14.*indpro';...
nw15.*indpro';nw16.*indpro';nw17.*indpro';nw18.*indpro';nw19.*indpro';nw20.
*indpro';nw21.*indpro';...
nw22.*indpro';nw23.*indpro';
nw24.*indpro';nw25.*indpro';nw26.*indpro';nw27.*indpro';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9;nw10;nw11;nw12;nw13;nw14;nw15;nw16;nw17
;nw18;nw19;nw20;nw21;...
nw22;nw23;nw24;nw25;nw26;nw27]

% Least square algorithm with SVD
params_train=pinv(X_train')*y
y1=X_train'*params_train;
e=y1-y;
error=sum(sum(e.^2))
```

```
bases_1=[sigma_1;sigma_2;sigma_3]
bases_2=[sigmae_1;sigmae_2;sigmae_3]
bases_3=[sigma_indpro_1;sigma_indpro_2;sigma_indpro_3]

bases=[bases_1;bases_2 ]

centers_1=[c_1;c_2;c_3]
centers_2=[ce_1;ce_2;ce_3]

centers=[centers_1;centers_2 ]

beta_1=[b1_gdp;b2_gdp;b3_gdp]
beta_2=[b1_une;b2_une;b3_une]

be=[beta_1;beta_2 ]


W=[mf_gdp_low';mf_gdp_medium';mf_gdp_high';mf_une_low';mf_une_medium';mf_un
e_high';mf_indpro_low';...
    mf_indpro_medium';mf_indpro_high']
W=W'
[nkk nii]=size(W)

[nk,ni]=size(x);
[n_rules,ni1]=size(centers);


maxepochs=5
SSE_goal=5;
epochs=0

e.field=e
W.field=W

% Start the error backpropagation algorithm
while (epochs<maxepochs) & (error>SSE_goal)

for i=1:n_rules/2


if memb==3
eta_base1{i,:}=lr_base_gdp
eta_base2{i,:}=lr_base_une
eta_base3{i,:}=lr_base_indpro
eta_center1{i,:}=lr_center_gdp
eta_center2{i,:}=lr_center_une
eta_center3{i,:}=lr_center_indpro
eta_beta1{i,:}=lr_beta_gdp
eta_beta2{i,:}=lr_beta_une
eta_beta3{i,:}=lr_beta_indpro

    else
eta_base1{i,:}=lr_base_gdp
eta_base2{i,:}=lr_base_une
eta_base3{i,:}=lr_base_indpo
eta_center1{i,:}=lr_center_gdp
eta_center2{i,:}=lr_center_une
```

```
eta_center3{i,:}=lr_center_indpro

end
eta_r{:,:}=lr_r
end
if memb==3
eta_base=[eta_base1;eta_base2;eta_base3 ]

eta_center=[eta_center1;eta_center2;eta_center3 ]

eta_beta=[eta_beta1;eta_beta2;eta_beta3 ]

else
eta_base=[eta_base1;eta_base2;eta_base3 ]

eta_center=[eta_center1;eta_center2;eta_center3 ]

end

if memb==1
    for i=1:n_rules
    for j=1:ni



    ind_rule{:,i}=find((x(:,j)-centers(i,:))<=(bases(i,:)/2));

    end
end


x.field=x
 y1.field=y1
params_train.field=params_train
X_train.field=X_train
for kk=1:n_rules


delta_center{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})'*...
    (((2*sign(x.field(ind_rule{:,kk}))-centers(kk,:)))/bases(kk,:)))

delta_base{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})'*...
    (((1-W.field(ind_rule{:,kk}))))/centers(kk,:))';


delta_r{:,:}=(e.field(ind_rule{1,kk}))'*(W.field(ind_rule{:,kk}));

del_center{:,kk}=-((eta_center{kk,:}/(2*nk))*sum(delta_center{:,kk}));


del_base{:,kk}=-((eta_base{kk,:}/(2*nk))*sum(delta_base{:,kk}));

del_r{:,:}=-((eta_r{:,:}/(2*nk))*sum(delta_r{:,:}))

del_center_num(kk,:)=(del_center{:,kk})'

del_base_num(kk,:)=(del_base{:,kk})'
```

```
del_r_num(:,:)=(del_r{:,:})'

        end

elseif memb==2
for i=1:n_rules
    for j=1:ni



    ind_rule{:,i}=find((x(:,j)-centers(i,:))<=(bases(i,:)^2/2));

    end
end

x.field=x
y1.field=y1




for kk=1:n_rules

delta_center{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'*...
    ((((x.field(ind_rule{1,kk})-centers(kk,:)))/bases(kk,:)^2))

delta_base{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'*...
    ((((x.field(ind_rule{1,kk})-centers(kk,:))).^2/bases(kk,:)^3))

delta_r{:,:}=(e.field(ind_rule{1,kk}))'*(W.field(ind_rule{:,kk}));


del_center{:,kk}=(-((eta_center{kk,:})/(2*nk))*sum(delta_center{:,kk}));


del_base{:,kk}=(-((2*eta_base{kk,:})/(2*nk))*sum(delta_base{:,kk}));

del_center_num(kk,:)=(del_center{:,kk})'

del_base_num(kk,:)=(del_base{:,kk})'

del_r{:,:}=-((eta_r{:,:}/(2*nk))*sum(delta_r{:,:}))


del_r_num(:,:)=(del_r{:,:})'
end

elseif memb==3
for i=1:n_rules
    for j=1:ni



    ind_rule{:,i}=find(exp(-(x(:,j)-centers(i,:)))<=(exp(bases(i,:))));
    end
end

x.field=x
y1.field=y1




for kk=1:n_rules
delta_center{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'*...
    ((2*((x.field(ind_rule{1,kk})-centers(kk,:))/...
    bases(kk,:))'*be(kk,:)))'


delta_base{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'*...
    (2*((x.field(ind_rule{1,kk})-centers(kk,:))/...
    bases(kk,:))'*be(kk,:))'

delta_be{:,kk}=(y1.field(ind_rule{1,kk})'*e.field(ind_rule{1,kk}))*...
    (-2*((x.field(ind_rule{1,kk})-centers(kk,:))/...
    bases(kk,:))'.^2*be(kk,:))

delta_r{:,:}=(e.field(ind_rule{1,kk}))'*(W.field(ind_rule{:,kk}));

del_center{:,kk}=(-((eta_center{kk,:})/(2*nk))*sum(delta_center{:,kk}));


del_base{:,kk}=(-((eta_base{kk,:})/(2*nk))*sum(delta_base{:,kk}));

del_be{:,kk}=(-((eta_beta{kk,:})/(2*nk))*sum(delta_be{:,kk}));

del_r{:,:}=-((eta_r{:,:}/(2*nk))*sum(delta_r{:,:}))

del_center_num=(del_center{:,kk})'
del_base_num=(del_base{:,kk})'
del_be_num=(del_be{:,kk})'

del_r_num(:,:)=(del_r{:,:})'


    end
end

if memb==3
centers=centers+del_center_num
bases=bases+del_base_num
be=be+del_be_num

y1.field=y1.field+del_r_num

else
    centers=centers+del_center_num
bases=bases+del_base_num
y1.field=y1.field+del_r_num
end
    if memb==3
```

```
center_gdp=centers(1:3,1)
center_une=centers(4:6,1)
center_indpro=centers(7:9,1)

bases_gdp=bases(1:3,1)
bases_une=bases(4:6,1)
bases_indpro=bases(7:9,1)
beta_gdp=bases(1:3,1)
beta_une=bases(4:6,1)
beta_indpro=bases(7:9,1)
centers=[center_gdp;center_une;center_indpro ]

bases=[bases_gdp;bases_une;bases_indpro ]
be=[beta_gdp;beta_une;beta_indpro]
else
center_gdp=centers(1:3,1)
center_une=centers(4:6,1)
center_indpro=centers(7:9,1)

bases_gdp=bases(1:3,1)
bases_une=bases(4:6,1)
bases_indpro=bases(7:9,1)

centers=[center_gdp;center_une;center_indpro ]

bases=[bases_gdp;bases_une;bases_indpro ]

end

if memb==1
mf_gdp_low=trimf(gdp,[-bases_gdp(1,1)/2+center_gdp(1,1) center_gdp(1,1)...
    bases_gdp(1,1)/2+center_gdp(1,1)  ]);  % LOW
mf_gdp_medium=trimf(gdp,[-bases_gdp(2,1)/2+center_gdp(2,1)
center_gdp(2,1)...
    bases_gdp(2,1)/2+center_gdp(2,1) ]);  % LOW
mf_gdp_high=trimf(gdp,[-bases_gdp(3,1)/2+center_gdp(3,1) center_gdp(3,1)...
    bases_gdp(3,1)/2+center_gdp(3,1)]);   % LOW



mf_une_low=trimf(une,[-bases_une(1,1)/2+center_une(1,1) center_une(1,1)...
    bases_une(1,1)/2+center_une(1,1)]); % LOW
mf_une_medium=trimf(une,[-bases_une(2,1)/2+center_une(2,1)
center_une(2,1)...
    bases_une(2,1)/2+center_une(2,1)]); % MEDIUM
mf_une_high=trimf(une,[-bases_une(3,1)/2+center_une(3,1) center_une(3,1)...
    bases_une(3,1)/2+center_une(3,1)]); % HIGH


mf_indpro_low=trimf(indpro,[-bases_indpro(1,1)/2+center_indpro(1,1)
center_indpro(1,1)...
  bases_indpro(1,1)/2+center_indpro(1,1) ]); % LOW
mf_indpro_medium=trimf(indpro,[-bases_indpro(2,1)/2+center_indpro(2,1)
center_indpro(2,1)...
  bases_indpro(2,1)/2+center_indpro(2,1)  ]); % MEDIUM
mf_indpro_high=trimf(indpro,[-bases_indpro(3,1)/2+center_indpro(3,1)
center_indpro(3,1)...
  bases_indpro(3,1)/2+center_indpro(3,1) ]); % HIGH
```

```
elseif memb==2
mf_gdp_low=gaussmf(gdp,[bases_gdp(1,1) center_gdp(1,1)  ]);       % LOW
mf_gdp_medium=gaussmf(gdp,[bases_gdp(2,1) center_gdp(2,1)  ]);    % MEDIUM
mf_gdp_high=gaussmf(gdp,[bases_gdp(3,1) center_gdp(3,1) ]);       % HIGH


mf_une_low=gaussmf(une,[bases_une(1,1) center_une(1,1)  ]);       % LOW
mf_une_medium=gaussmf(une,[bases_une(2,1) center_une(2,1)  ]);    % MEDIUM
mf_une_high=gaussmf(une,[bases_une(3,1) center_une(3,1)  ]);      % HIGH


mf_indpro_low=gaussmf(indpro,[bases_indpro(1,1) center_indpro(1,1)   ]);
% LOW
mf_indpro_medium=gaussmf(indpro,[bases_indpro(2,1) center_indpro(2,1)   ]);
% MEDIUM
mf_indpro_high=gaussmf(indpro,[bases_indpro(3,1) center_indpro(3,1)  ]);
% HIGH

elseif memb==3
mf_gdp_low=gbell_mf(gdp,[bases_gdp(1,1) beta_gdp(1,1) center_gdp(1,1)  ]);
% LOW
mf_gdp_medium=gbell_mf(gdp,[bases_gdp(2,1) beta_gdp(2,1) center_gdp(2,1)
]);   % MEDIUM
mf_gdp_high=gbell_mf(gdp,[bases_gdp(3,1) beta_gdp(3,1) center_gdp(3,1) ]);
% HIGH


mf_une_low=gbell_mf(une,[bases_une(1,1) beta_une(1,1) center_une(1,1)  ]);
% LOW
mf_une_medium=gbell_mf(une,[bases_une(2,1) beta_une(2,1) center_une(2,1)
]);   % MEDIUM
mf_une_high=gbell_mf(une,[bases_une(3,1) beta_une(3,1) center_une(3,1)  ]);
% HIGH


mf_indpro_low=gbell_mf(indpro,[bases_indpro(1,1) beta_indpro(1,1)
center_indpro(1,1)   ]);     % LOW
mf_indpro_medium=gbell_mf(indpro,[bases_indpro(2,1) beta_indpro(2,1)
center_indpro(2,1)   ]);  % MEDIUM
mf_indpro_high=gbell_mf(indpro,[bases_indpro(3,1) beta_indpro(3,1)
center_indpro(3,1)  ]);


end

% Take the fuzzy rules
if em==1             % Min operator for AND rule
w1=min([mf_gdp_low mf_une_low mf_indpro_low]')           % LOW-LOW-LOW
w2=min([mf_gdp_low mf_une_low mf_indpro_medium]')        % LOW-LOW-MEDIUM
w3=min([mf_gdp_low mf_une_low mf_indpro_high]')          % LOW-LOW-HIGH
w4=min([mf_gdp_low mf_une_medium mf_indpro_low]')        % LOW-MEDIUM-LOW
w5=min([mf_gdp_low mf_une_medium mf_indpro_medium]')     % LOW-MEDIUM-MEDIUM
w6=min([mf_gdp_low mf_une_medium mf_indpro_high]')       % LOW-MEDIUM-HIGH
w7=min([mf_gdp_low mf_une_high mf_indpro_low]')          % LOW-HIGH-LOW
w8=min([mf_gdp_low mf_une_high mf_indpro_medium]')       % LOW-HIGH-MEDIUM
w9=min([mf_gdp_low mf_une_high mf_indpro_high]')         % LOW-HIGH-HIGH
w10=min([mf_gdp_medium mf_une_low mf_indpro_low]')       % MEDIUM-LOW-LOW
```

```
w11=min([mf_gdp_medium mf_une_low mf_indpro_medium]')    % MEDIUM-LOW-MEDIUM
w12=min([mf_gdp_medium mf_une_low mf_indpro_high]')      % MEDIUM-LOW-HIGH
w13=min([mf_gdp_medium mf_une_medium mf_indpro_low]')    % MEDIUM-MEDIUM-LOW
w14=min([mf_gdp_medium mf_une_medium mf_indpro_medium]')   % MEDIUM-MEDIUM-
MEDIUM
w15=min([mf_gdp_medium mf_une_medium mf_indpro_high]')   % MEDIUM-MEDIUM-
HIGH
w16=min([mf_gdp_medium mf_une_high mf_indpro_low]')      % MEDIUM-HIGH-LOW
w17=min([mf_gdp_medium mf_une_high mf_indpro_medium]')   % MEDIUM-HIGH-
MEDIUM
w18=min([mf_gdp_medium mf_une_high mf_indpro_high]')     % MEDIUM-HIGH-HIGH
w19=min([mf_gdp_high mf_une_low mf_indpro_low]')         % HIGH-LOW-LOW
w20=min([mf_gdp_high mf_une_low mf_indpro_medium]')      % HIGH-LOW-MEDIUM
w21=min([mf_gdp_high mf_une_low mf_indpro_high]')        % HIGH-LOW-HIGH
w22=min([mf_gdp_high mf_une_medium mf_indpro_low]')      % HIGH-MEDIUM-LOW
w23=min([mf_gdp_high mf_une_medium mf_indpro_medium]')   % HIGH-MEDIUM-
MEDIUM
w24=min([mf_gdp_high mf_une_medium mf_indpro_high]')     % HIGH-MEDIUM-HIGH
w25=min([mf_gdp_high mf_une_high mf_indpro_low]')        % HIGH-HIGH-LOW
w26=min([mf_gdp_high mf_une_high mf_indpro_medium]')     % HIGH-HIGH-MEDIUM
w27=min([mf_gdp_high mf_une_high mf_indpro_high]')       % HIGH-HIGH-HIGH


elseif em==2       % product operator for AND rule
w1=(mf_gdp_low.*mf_une_low.*mf_indpro_low)'              % LOW-LOW-LOW
w2=(mf_gdp_low.*mf_une_low.*mf_indpro_medium)'           % LOW-LOW-MEDIUM
w3=(mf_gdp_low.*mf_une_low.*mf_indpro_high)'             % LOW-LOW-HIGH
w4=(mf_gdp_low.*mf_une_medium.*mf_indpro_low)'           % LOW-MEDIUM-LOW
w5=(mf_gdp_low.*mf_une_medium.*mf_indpro_medium)'        % LOW-MEDIUM-MEDIUM
w6=(mf_gdp_low.*mf_une_medium.*mf_indpro_high)'          % LOW-MEDIUM-HIGH
w7=(mf_gdp_low.*mf_une_high.*mf_indpro_low)'             % LOW-HIGH-LOW
w8=(mf_gdp_low.*mf_une_high.*mf_indpro_medium)'          % LOW-HIGH-MEDIUM
w9=(mf_gdp_low.*mf_une_high.*mf_indpro_high)'            % LOW-HIGH-HIGH
w10=(mf_gdp_medium.*mf_une_low.*mf_indpro_low)'          % MEDIUM-LOW-LOW
w11=(mf_gdp_medium.*mf_une_low.*mf_indpro_medium)'       % MEDIUM-LOW-MEDIUM
w12=(mf_gdp_medium.*mf_une_low.*mf_indpro_high)'         % MEDIUM-LOW-HIGH
w13=(mf_gdp_medium.*mf_une_medium.*mf_indpro_low)'       % MEDIUM-MEDIUM-LOW
w14=(mf_gdp_medium.*mf_une_medium.*mf_indpro_medium)'    % MEDIUM-MEDIUM-
MEDIUM
w15=(mf_gdp_medium.*mf_une_medium.*mf_indpro_high)'      % MEDIUM-MEDIUM-
HIGH
w16=(mf_gdp_medium.*mf_une_high.*mf_indpro_low)'         % MEDIUM-HIGH-LOW
w17=(mf_gdp_medium.*mf_une_high.*mf_indpro_medium)'      % MEDIUM-HIGH-
MEDIUM
w18=(mf_gdp_medium.*mf_une_high.*mf_indpro_high)'        % MEDIUM-HIGH-HIGH
w19=(mf_gdp_high.*mf_une_low.*mf_indpro_low)'            % HIGH-LOW-LOW
w20=(mf_gdp_high.*mf_une_low.*mf_indpro_medium)'         % HIGH-LOW-MEDIUM
w21=(mf_gdp_high.*mf_une_low.*mf_indpro_high)'           % HIGH-LOW-HIGH
w22=(mf_gdp_high.*mf_une_medium.*mf_indpro_low)'         % HIGH-MEDIUM-LOW
w23=(mf_gdp_high.*mf_une_medium.*mf_indpro_medium)'      % HIGH-MEDIUM-
MEDIUM
w24=(mf_gdp_high.*mf_une_medium.*mf_indpro_high)'        % HIGH-MEDIUM-HIGH
w25=(mf_gdp_high.*mf_une_high.*mf_indpro_low)'           % HIGH-HIGH-LOW
w26=(mf_gdp_high.*mf_une_high.*mf_indpro_medium)'        % HIGH-HIGH-MEDIUM
w27=(mf_gdp_high.*mf_une_high.*mf_indpro_high)'          % HIGH-HIGH-HIGH



elseif em==3       % max operator for OR rule
w1=max([mf_gdp_low mf_une_low mf_indpro_low]')               % LOW-LOW-LOW
```

```
w2=max([mf_gdp_low mf_une_low mf_indpro_medium]')           % LOW-LOW-
MEDIUM
w3=max([mf_gdp_low mf_une_low mf_indpro_high]')             % LOW-LOW-HIGH
w4=max([mf_gdp_low mf_une_medium mf_indpro_low]')           % LOW-MEDIUM-
LOW
w5=max([mf_gdp_low mf_une_medium mf_indpro_medium]')        % LOW-MEDIUM-
MEDIUM
w6=max([mf_gdp_low mf_une_medium mf_indpro_high]')          % LOW-MEDIUM-
HIGH
w7=max([mf_gdp_low mf_une_high mf_indpro_low]')             % LOW-HIGH-LOW
w8=max([mf_gdp_low mf_une_high mf_indpro_medium]')          % LOW-HIGH-
MEDIUM
w9=max([mf_gdp_low mf_une_high mf_indpro_high]')            % LOW-HIGH-HIGH
w10=max([mf_gdp_medium mf_une_low mf_indpro_low]')          % MEDIUM-LOW-
LOW
w11=max([mf_gdp_medium mf_une_low mf_indpro_medium]')       % MEDIUM-LOW-
MEDIUM
w12=max([mf_gdp_medium mf_une_low mf_indpro_high]')         % MEDIUM-LOW-
HIGH
w13=max([mf_gdp_medium mf_une_medium mf_indpro_low]')       % MEDIUM-
MEDIUM-LOW
w14=max([mf_gdp_medium mf_une_medium mf_indpro_medium]')    % MEDIUM-
MEDIUM-MEDIUM
w15=max([mf_gdp_medium mf_une_medium mf_indpro_high]')      % MEDIUM-
MEDIUM-HIGH
w16=max([mf_gdp_medium mf_une_high mf_indpro_low]')         % MEDIUM-HIGH-
LOW
w17=max([mf_gdp_medium mf_une_high mf_indpro_medium]')      % MEDIUM-HIGH-
MEDIUM
w18=max([mf_gdp_medium mf_une_high mf_indpro_high]')        % MEDIUM-HIGH-
HIGH
w19=max([mf_gdp_high mf_une_low mf_indpro_low]')            % HIGH-LOW-LOW
w20=max([mf_gdp_high mf_une_low mf_indpro_medium]')         % HIGH-LOW-
MEDIUM
w21=max([mf_gdp_high mf_une_low mf_indpro_high]')           % HIGH-LOW-HIGH
w22=max([mf_gdp_high mf_une_medium mf_indpro_low]')         % HIGH-MEDIUM-
LOW
w23=max([mf_gdp_high mf_une_medium mf_indpro_medium]')      % HIGH-MEDIUM-
MEDIUM
w24=max([mf_gdp_high mf_une_medium mf_indpro_high]')        % HIGH-MEDIUM-
HIGH
w25=max([mf_gdp_high mf_une_high mf_indpro_low]')           % HIGH-HIGH-LOW
w26=max([mf_gdp_high mf_une_high mf_indpro_medium]')        % HIGH-HIGH-
MEDIUM
w27=max([mf_gdp_high mf_une_high mf_indpro_high]')          % HIGH-HIGH-
HIGH
end



for j=1:t
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& ...
w8(:,j)==0& w9(:,j)==0& w10(:,j)==0 & w11(:,j)==0 & w12(:,j)==0 &
w13(:,j)==0& w14(:,j)==0 &...
w16(:,j)==0 & w17(:,j)==0 & w18(:,j)==0 & w19(:,j)==0 & w20(:,j)==0&
w21(:,j)==0& w22(:,j)==0&...
w23(:,j)==0 & w24(:,j)==0 & w25(:,j)==0 & w26(:,j)==0 & w27(:,j)==0)

nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;...
```

```
nw9(:,j)=0;nw10(:,j)=0;nw11(:,j)=0;nw12(:,j)=0;nw13(:,j)=0;
nw14(:,j)=0;nw15(:,j)=0;nw16(:,j)=0;...
nw17(:,j)=0;nw18(:,j)=0;nw19(:,j)=0;nw20(:,j)=0;
nw21(:,j)=0;nw22(:,j)=0;nw23(:,j)=0;nw24(:,j)=0;...
nw25(:,j)=0;nw26(:,j)=0;nw27(:,j)=0;

else
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));
```

```
nw10(:,j)=w10(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw11(:,j)=w11(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw12(:,j)=w12(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw13(:,j)=w13(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw14(:,j)=w14(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw15(:,j)=w15(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw16(:,j)=w16(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j
)+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw17(:,j)=w17(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw18(:,j)=w18(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw19(:,j)=w19(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
```

```
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw20(:,j)=w20(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw21(:,j)=w21(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw22(:,j)=w22(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw23(:,j)=w23(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw24(:,j)=w24(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw25(:,j)=w25(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw26(:,j)=w26(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw27(:,j)=w27(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

end

end


X_train=[nw1.*gdp';nw2.*gdp';nw3.*gdp';nw4.*gdp';nw5.*gdp';nw6.*gdp';nw7.*g
dp';nw8.*gdp';nw9.*gdp';...
nw10.*gdp';nw11.*gdp';nw12.*gdp';nw13.*gdp';nw14.*gdp';
nw15.*gdp';nw16.*gdp';nw17.*gdp';nw18.*gdp';...
```

```
nw19.*gdp';nw20.*gdp';nw21.*gdp';nw22.*gdp';nw23.*gdp';
nw24.*gdp';nw25.*gdp';nw26.*gdp';nw27.*gdp';...
nw1.*une';nw2.*une';nw3.*une';nw4.*une';nw5.*une';nw6.*une';nw7.*une';nw8.*
une';nw9.*une';...
nw10.*une';nw11.*une';nw12.*une';nw13.*une';nw14.*une';
nw15.*une';nw16.*une';nw17.*une';nw18.*une';...
nw19.*une';nw20.*une';nw21.*une';nw22.*une';nw23.*une';
nw24.*une';nw25.*une';nw26.*une';nw27.*une';...
nw1.*indpro';nw2.*indpro';nw3.*indpro';nw4.*indpro';nw5.*indpro';nw6.*indpr
o';nw7.*indpro';...
nw8.*indpro';nw9.*indpro';nw10.*indpro';nw11.*indpro';nw12.*indpro';nw13.*i
ndpro';nw14.*indpro';...
nw15.*indpro';nw16.*indpro';nw17.*indpro';nw18.*indpro';nw19.*indpro';nw20.
*indpro';nw21.*indpro';...
nw22.*indpro';nw23.*indpro';
nw24.*indpro';nw25.*indpro';nw26.*indpro';nw27.*indpro';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9;nw10;nw11;nw12;nw13;nw14;nw15;nw16;nw17
;nw18;nw19;nw20;nw21;...
nw22;nw23;nw24;nw25;nw26;nw27]
x=x.field

params_train=pinv(X_train')*y % [p1 p2 q1 q2 s1 s2 r1 r2....pn qn sn rn]
y1=X_train'*params_train;
e=y1-y;
error=sum(sum(e.^2))
e.field=e
epochs=epochs+1
array_y ( epochs )= error;
index_epochs ( epochs ) = epochs;
end

antecedent_par=[nw1',nw2',nw3',nw4',nw5',nw6',nw7',nw8',nw9',nw10',nw11',nw
12',...
nw13',nw14',nw15',nw16',nw17',nw18',nw19',nw20',nw21',nw22',nw23',nw24',nw2
5',...
nw26',nw27']
consequent_par=y1

output_2=sum(antecedent_par'*consequent_par)/sum(sum(antecedent_par))

out=output_2


for kkk=1:nk
    if y1(kkk,:)>out
        S_in_sample(kkk,:)=1

    elseif y1(kkk,:)<out
        S_in_sample(kkk,:)=0
    end
end

% Compuation for clasiffication table preparation
Actual_positive_in_sample=find(y==1)
Actual_negative_in_sample=find(y==0)

Predicted_positive_in_sample=find(S_in_sample==1)
Predicted_negative_in_sample=find(S_in_sample==0)
```

```
Sum_actual_positive_in_sample=length(Actual_positive_in_sample)
Sum_actual_negative_in_sample=length(Actual_negative_in_sample)

Sum_predicted_positive_in_sample=length(Predicted_positive_in_sample)
Sum_predicted_negative_in_sample=length(Predicted_negative_in_sample)

Total_predicted_in_sample=find(y==S_in_sample)

Perc=(length(Total_predicted_in_sample)/nk)*100

W_in_sample=y(Total_predicted_in_sample)

Positive_in_sample=find(W_in_sample==1)
Negative_in_sample=find(W_in_sample==0)


Final_predicted_Positive_in_sample=length(Positive_in_sample)/Sum_actual_po
sitive_in_sample

Final_predicted_Negative_in_sample=length(Negative_in_sample)/Sum_actual_ne
gative_in_sample

Final_predicted_Positive_in_sample=Final_predicted_Positive_in_sample*100

Final_predicted_Negative_in_sample=Final_predicted_Negative_in_sample*100

Total_performance_in_sample=((length(Positive_in_sample) +
length(Negative_in_sample))/nk)*100


clear nw1
clear nw2
clear nw3
clear nw4
clear nw5
clear nw6
clear nw7
clear nw8
clear nw9
clear nw10
clear nw11
clear nw12
clear nw13
clear nw14
clear nw15
clear nw16
clear nw17
clear nw18
clear nw19
clear nw20
clear nw21
clear nw22
clear nw23
clear nw24
clear nw25
clear nw26
clear nw27
```

```
load file.mat
y_tes=data(end-15:end,end)
x_tes=data(end-15:end,1:3)
gdp=x_tes(:,1)
une=x_tes(:,2)
indpro=x_tes(:,3)
t1=length(y_tes)

if memb==1
mf_gdp_low=trimf(gdp,[-bases_gdp(1,1)/2+center_gdp(1,1) center_gdp(1,1)...
    bases_gdp(1,1)/2+center_gdp(1,1)  ]);  % LOW
mf_gdp_medium=trimf(gdp,[-bases_gdp(2,1)/2+center_gdp(2,1)
center_gdp(2,1)...
    bases_gdp(2,1)/2+center_gdp(2,1) ]);  % LOW
mf_gdp_high=trimf(gdp,[-bases_gdp(3,1)/2+center_gdp(3,1) center_gdp(3,1)...
    bases_gdp(3,1)/2+center_gdp(3,1)]);   % LOW


mf_une_low=trimf(une,[-bases_une(1,1)/2+center_une(1,1) center_une(1,1)...
    bases_une(1,1)/2+center_une(1,1)]); % LOW
mf_une_medium=trimf(une,[-bases_une(2,1)/2+center_une(2,1)
center_une(2,1)...
    bases_une(2,1)/2+center_une(2,1)]); % MEDIUM
mf_une_high=trimf(une,[-bases_une(3,1)/2+center_une(3,1) center_une(3,1)...
    bases_une(3,1)/2+center_une(3,1)]); % HIGH


mf_indpro_low=trimf(indpro,[-bases_indpro(1,1)/2+center_indpro(1,1)
center_indpro(1,1)...
  bases_indpro(1,1)/2+center_indpro(1,1) ]); % LOW
mf_indpro_medium=trimf(indpro,[-bases_indpro(2,1)/2+center_indpro(2,1)
center_indpro(2,1)...
  bases_indpro(2,1)/2+center_indpro(2,1)  ]); % MEDIUM
mf_indpro_high=trimf(indpro,[-bases_indpro(3,1)/2+center_indpro(3,1)
center_indpro(3,1)...
  bases_indpro(3,1)/2+center_indpro(3,1) ]); % HIGH


elseif memb==2
mf_gdp_low=gaussmf(gdp,[bases_gdp(1,1) center_gdp(1,1)  ]);      % LOW
mf_gdp_medium=gaussmf(gdp,[bases_gdp(2,1) center_gdp(2,1)  ]);   % MEDIUM
mf_gdp_high=gaussmf(gdp,[bases_gdp(3,1) center_gdp(3,1) ]);      % HIGH


mf_une_low=gaussmf(une,[bases_une(1,1) center_une(1,1)  ]);      % LOW
mf_une_medium=gaussmf(une,[bases_une(2,1) center_une(2,1)  ]);   % MEDIUM
mf_une_high=gaussmf(une,[bases_une(3,1) center_une(3,1)  ]);     % HIGH


mf_indpro_low=gaussmf(indpro,[bases_indpro(1,1) center_indpro(1,1)   ]);
% LOW
mf_indpro_medium=gaussmf(indpro,[bases_indpro(2,1) center_indpro(2,1)   ]);
% MEDIUM
mf_indpro_high=gaussmf(indpro,[bases_indpro(3,1) center_indpro(3,1)  ]);
% HIGH

elseif memb==3
```

```
mf_gdp_low=gbell_mf(gdp,[bases_gdp(1,1) be(1,1) center_gdp(1,1)  ]);      %
mf_gdp_low=gbell_mf(gdp,[bases_gdp(1,1) beta_gdp(1,1) center_gdp(1,1)  ]);
% LOW
mf_gdp_medium=gbell_mf(gdp,[bases_gdp(2,1) beta_gdp(2,1) center_gdp(2,1)
]);   % MEDIUM
mf_gdp_high=gbell_mf(gdp,[bases_gdp(3,1) beta_gdp(3,1) center_gdp(3,1) ]);
% HIGH


mf_une_low=gbell_mf(une,[bases_une(1,1) beta_une(1,1) center_une(1,1)  ]);
% LOW
mf_une_medium=gbell_mf(une,[bases_une(2,1) beta_une(2,1) center_une(2,1)
]);   % MEDIUM
mf_une_high=gbell_mf(une,[bases_une(3,1) beta_une(3,1) center_une(3,1)  ]);
% HIGH


mf_indpro_low=gbell_mf(indpro,[bases_indpro(1,1) beta_indpro(1,1)
center_indpro(1,1)   ]);     % LOW
mf_indpro_medium=gbell_mf(indpro,[bases_indpro(2,1) beta_indpro(2,1)
center_indpro(2,1)   ]);  % MEDIUM
mf_indpro_high=gbell_mf(indpro,[bases_indpro(3,1) beta_indpro(3,1)
center_indpro(3,1)  ]);


end

% Take the fuzzy rules
if em==1             % Min operator for AND rule
w1=min([mf_gdp_low mf_une_low mf_indpro_low]')           % LOW-LOW-LOW
w2=min([mf_gdp_low mf_une_low mf_indpro_medium]')        % LOW-LOW-MEDIUM
w3=min([mf_gdp_low mf_une_low mf_indpro_high]')          % LOW-LOW-HIGH
w4=min([mf_gdp_low mf_une_medium mf_indpro_low]')        % LOW-MEDIUM-LOW
w5=min([mf_gdp_low mf_une_medium mf_indpro_medium]')     % LOW-MEDIUM-MEDIUM
w6=min([mf_gdp_low mf_une_medium mf_indpro_high]')       % LOW-MEDIUM-HIGH
w7=min([mf_gdp_low mf_une_high mf_indpro_low]')          % LOW-HIGH-LOW
w8=min([mf_gdp_low mf_une_high mf_indpro_medium]')       % LOW-HIGH-MEDIUM
w9=min([mf_gdp_low mf_une_high mf_indpro_high]')         % LOW-HIGH-HIGH
w10=min([mf_gdp_medium mf_une_low mf_indpro_low]')       % MEDIUM-LOW-LOW
w11=min([mf_gdp_medium mf_une_low mf_indpro_medium]')    % MEDIUM-LOW-MEDIUM
w12=min([mf_gdp_medium mf_une_low mf_indpro_high]')      % MEDIUM-LOW-HIGH
w13=min([mf_gdp_medium mf_une_medium mf_indpro_low]')    % MEDIUM-MEDIUM-LOW
w14=min([mf_gdp_medium mf_une_medium mf_indpro_medium]')  % MEDIUM-MEDIUM-
MEDIUM
w15=min([mf_gdp_medium mf_une_medium mf_indpro_high]')   % MEDIUM-MEDIUM-
HIGH
w16=min([mf_gdp_medium mf_une_high mf_indpro_low]')      % MEDIUM-HIGH-LOW
w17=min([mf_gdp_medium mf_une_high mf_indpro_medium]')   % MEDIUM-HIGH-
MEDIUM
w18=min([mf_gdp_medium mf_une_high mf_indpro_high]')     % MEDIUM-HIGH-HIGH
w19=min([mf_gdp_high mf_une_low mf_indpro_low]')         % HIGH-LOW-LOW
w20=min([mf_gdp_high mf_une_low mf_indpro_medium]')      % HIGH-LOW-MEDIUM
w21=min([mf_gdp_high mf_une_low mf_indpro_high]')        % HIGH-LOW-HIGH
w22=min([mf_gdp_high mf_une_medium mf_indpro_low]')      % HIGH-MEDIUM-LOW
w23=min([mf_gdp_high mf_une_medium mf_indpro_medium]')   % HIGH-MEDIUM-
MEDIUM
w24=min([mf_gdp_high mf_une_medium mf_indpro_high]')     % HIGH-MEDIUM-HIGH
w25=min([mf_gdp_high mf_une_high mf_indpro_low]')        % HIGH-HIGH-LOW
w26=min([mf_gdp_high mf_une_high mf_indpro_medium]')     % HIGH-HIGH-MEDIUM
w27=min([mf_gdp_high mf_une_high mf_indpro_high]')       % HIGH-HIGH-HIGH
```

```
elseif em==2      % product operator for AND rule
w1=(mf_gdp_low.*mf_une_low.*mf_indpro_low)'                  % LOW-LOW-LOW
w2=(mf_gdp_low.*mf_une_low.*mf_indpro_medium)'               % LOW-LOW-MEDIUM
w3=(mf_gdp_low.*mf_une_low.*mf_indpro_high)'                 % LOW-LOW-HIGH
w4=(mf_gdp_low.*mf_une_medium.*mf_indpro_low)'               % LOW-MEDIUM-LOW
w5=(mf_gdp_low.*mf_une_medium.*mf_indpro_medium)'            % LOW-MEDIUM-MEDIUM
w6=(mf_gdp_low.*mf_une_medium.*mf_indpro_high)'              % LOW-MEDIUM-HIGH
w7=(mf_gdp_low.*mf_une_high.*mf_indpro_low)'                 % LOW-HIGH-LOW
w8=(mf_gdp_low.*mf_une_high.*mf_indpro_medium)'              % LOW-HIGH-MEDIUM
w9=(mf_gdp_low.*mf_une_high.*mf_indpro_high)'                % LOW-HIGH-HIGH
w10=(mf_gdp_medium.*mf_une_low.*mf_indpro_low)'              % MEDIUM-LOW-LOW
w11=(mf_gdp_medium.*mf_une_low.*mf_indpro_medium)'           % MEDIUM-LOW-MEDIUM
w12=(mf_gdp_medium.*mf_une_low.*mf_indpro_high)'             % MEDIUM-LOW-HIGH
w13=(mf_gdp_medium.*mf_une_medium.*mf_indpro_low)'           % MEDIUM-MEDIUM-LOW
w14=(mf_gdp_medium.*mf_une_medium.*mf_indpro_medium)'        % MEDIUM-MEDIUM-
MEDIUM
w15=(mf_gdp_medium.*mf_une_medium.*mf_indpro_high)'          % MEDIUM-MEDIUM-
HIGH
w16=(mf_gdp_medium.*mf_une_high.*mf_indpro_low)'             % MEDIUM-HIGH-LOW
w17=(mf_gdp_medium.*mf_une_high.*mf_indpro_medium)'          % MEDIUM-HIGH-
MEDIUM
w18=(mf_gdp_medium.*mf_une_high.*mf_indpro_high)'            % MEDIUM-HIGH-HIGH
w19=(mf_gdp_high.*mf_une_low.*mf_indpro_low)'                % HIGH-LOW-LOW
w20=(mf_gdp_high.*mf_une_low.*mf_indpro_medium)'             % HIGH-LOW-MEDIUM
w21=(mf_gdp_high.*mf_une_low.*mf_indpro_high)'               % HIGH-LOW-HIGH
w22=(mf_gdp_high.*mf_une_medium.*mf_indpro_low)'             % HIGH-MEDIUM-LOW
w23=(mf_gdp_high.*mf_une_medium.*mf_indpro_medium)'          % HIGH-MEDIUM-
MEDIUM
w24=(mf_gdp_high.*mf_une_medium.*mf_indpro_high)'            % HIGH-MEDIUM-HIGH
w25=(mf_gdp_high.*mf_une_high.*mf_indpro_low)'               % HIGH-HIGH-LOW
w26=(mf_gdp_high.*mf_une_high.*mf_indpro_medium)'            % HIGH-HIGH-MEDIUM
w27=(mf_gdp_high.*mf_une_high.*mf_indpro_high)'              % HIGH-HIGH-HIGH


elseif em==3      % product operator for OR rule
w1=max([mf_gdp_low mf_une_low mf_indpro_low]')                   % LOW-LOW-LOW
w2=max([mf_gdp_low mf_une_low mf_indpro_medium]')                % LOW-LOW-
MEDIUM
w3=max([mf_gdp_low mf_une_low mf_indpro_high]')                  % LOW-LOW-HIGH
w4=max([mf_gdp_low mf_une_medium mf_indpro_low]')                % LOW-MEDIUM-
LOW
w5=max([mf_gdp_low mf_une_medium mf_indpro_medium]')             % LOW-MEDIUM-
MEDIUM
w6=max([mf_gdp_low mf_une_medium mf_indpro_high]')               % LOW-MEDIUM-
HIGH
w7=max([mf_gdp_low mf_une_high mf_indpro_low]')                  % LOW-HIGH-LOW
w8=max([mf_gdp_low mf_une_high mf_indpro_medium]')               % LOW-HIGH-
MEDIUM
w9=max([mf_gdp_low mf_une_high mf_indpro_high]')                 % LOW-HIGH-HIGH
w10=max([mf_gdp_medium mf_une_low mf_indpro_low]')               % MEDIUM-LOW-
LOW
w11=max([mf_gdp_medium mf_une_low mf_indpro_medium]')            % MEDIUM-LOW-
MEDIUM
```

```
w12=max([mf_gdp_medium mf_une_low mf_indpro_high]')          % MEDIUM-LOW-
HIGH
w13=max([mf_gdp_medium mf_une_medium mf_indpro_low]')        % MEDIUM-
MEDIUM-LOW
w14=max([mf_gdp_medium mf_une_medium mf_indpro_medium]')     % MEDIUM-
MEDIUM-MEDIUM
w15=max([mf_gdp_medium mf_une_medium mf_indpro_high]')       % MEDIUM-
MEDIUM-HIGH
w16=max([mf_gdp_medium mf_une_high mf_indpro_low]')          % MEDIUM-HIGH-
LOW
w17=max([mf_gdp_medium mf_une_high mf_indpro_medium]')       % MEDIUM-HIGH-
MEDIUM
w18=max([mf_gdp_medium mf_une_high mf_indpro_high]')         % MEDIUM-HIGH-
HIGH
w19=max([mf_gdp_high mf_une_low mf_indpro_low]')             % HIGH-LOW-LOW
w20=max([mf_gdp_high mf_une_low mf_indpro_medium]')          % HIGH-LOW-
MEDIUM
w21=max([mf_gdp_high mf_une_low mf_indpro_high]')            % HIGH-LOW-HIGH
w22=max([mf_gdp_high mf_une_medium mf_indpro_low]')          % HIGH-MEDIUM-
LOW
w23=max([mf_gdp_high mf_une_medium mf_indpro_medium]')       % HIGH-MEDIUM-
MEDIUM
w24=max([mf_gdp_high mf_une_medium mf_indpro_high]')         % HIGH-MEDIUM-
HIGH
w25=max([mf_gdp_high mf_une_high mf_indpro_low]')            % HIGH-HIGH-LOW
w26=max([mf_gdp_high mf_une_high mf_indpro_medium]')         % HIGH-HIGH-
MEDIUM
w27=max([mf_gdp_high mf_une_high mf_indpro_high]')           % HIGH-HIGH-
HIGH
end



for j=1:t1
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& ...
w8(:,j)==0& w9(:,j)==0& w10(:,j)==0 & w11(:,j)==0 & w12(:,j)==0 &
w13(:,j)==0& w14(:,j)==0 &...
w16(:,j)==0 & w17(:,j)==0 & w18(:,j)==0 & w19(:,j)==0 & w20(:,j)==0&
w21(:,j)==0& w22(:,j)==0&...
w23(:,j)==0 & w24(:,j)==0 & w25(:,j)==0 & w26(:,j)==0 & w27(:,j)==0)

nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;...
nw9(:,j)=0;nw10(:,j)=0;nw11(:,j)=0;nw12(:,j)=0;nw13(:,j)=0;
nw14(:,j)=0;nw15(:,j)=0;nw16(:,j)=0;...
nw17(:,j)=0;nw18(:,j)=0;nw19(:,j)=0;nw20(:,j)=0;
nw21(:,j)=0;nw22(:,j)=0;nw23(:,j)=0;nw24(:,j)=0;...
nw25(:,j)=0;nw26(:,j)=0;nw27(:,j)=0;

else
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)+w
8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw10(:,j)=w10(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw11(:,j)=w11(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));
```

```
nw12(:,j)=w12(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw13(:,j)=w13(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw14(:,j)=w14(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw15(:,j)=w15(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw16(:,j)=w16(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j
)+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw17(:,j)=w17(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw18(:,j)=w18(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw19(:,j)=w19(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw20(:,j)=w20(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw21(:,j)=w21(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));
```

```
nw22(:,j)=w22(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw23(:,j)=w23(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw24(:,j)=w24(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw25(:,j)=w25(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw26(:,j)=w26(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));

nw27(:,j)=w27(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)+w6(:,j)+w7(:,j)
+w8(:,j)+w9(:,j)+...
w10(:,j)+w11(:,j)+w12(:,j)+w13(:,j)+w14(:,j)+w15(:,j)+w16(:,j)+w17(:,j)+w18
(:,j)+w19(:,j)+...
w20(:,j)+w21(:,j)+w22(:,j)+w23(:,j)+w24(:,j)+w25(:,j)+w26(:,j)+w27(:,j));


end

end


X_test=[nw1.*gdp';nw2.*gdp';nw3.*gdp';nw4.*gdp';nw5.*gdp';nw6.*gdp';nw7.*gd
p';nw8.*gdp';nw9.*gdp';...
nw10.*gdp';nw11.*gdp';nw12.*gdp';nw13.*gdp';nw14.*gdp';
nw15.*gdp';nw16.*gdp';nw17.*gdp';nw18.*gdp';...
nw19.*gdp';nw20.*gdp';nw21.*gdp';nw22.*gdp';nw23.*gdp';
nw24.*gdp';nw25.*gdp';nw26.*gdp';nw27.*gdp';...
nw1.*une';nw2.*une';nw3.*une';nw4.*une';nw5.*une';nw6.*une';nw7.*une';nw8.*
une';nw9.*une';...
nw10.*une';nw11.*une';nw12.*une';nw13.*une';nw14.*une';
nw15.*une';nw16.*une';nw17.*une';nw18.*une';...
nw19.*une';nw20.*une';nw21.*une';nw22.*une';nw23.*une';
nw24.*une';nw25.*une';nw26.*une';nw27.*une';...
nw1.*indpro';nw2.*indpro';nw3.*indpro';nw4.*indpro';nw5.*indpro';nw6.*indpr
o';nw7.*indpro';...
nw8.*indpro';nw9.*indpro';nw10.*indpro';nw11.*indpro';nw12.*indpro';nw13.*i
ndpro';nw14.*indpro';...
nw15.*indpro';nw16.*indpro';nw17.*indpro';nw18.*indpro';nw19.*indpro';nw20.
*indpro';nw21.*indpro';...
nw22.*indpro';nw23.*indpro';
nw24.*indpro';nw25.*indpro';nw26.*indpro';nw27.*indpro';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9;nw10;nw11;nw12;nw13;nw14;nw15;nw16;nw17
;nw18;nw19;nw20;nw21;...
nw22;nw23;nw24;nw25;nw26;nw27]

yf1=X_test'*params_train;
for kkk=1:t1
    if yf1(kkk,:)>out
        S_out_sample(kkk,:)=1

    elseif yf1(kkk,:)<out
        S_out_sample(kkk,:)=0
    end
end

Actual_positive=find(y_tes==1)
Actual_negative=find(y_tes==0)

Predicted_positive=find(S_out_sample==1)
Predicted_negative=find(S_out_sample==0)

Sum_actual_positive=length(Actual_positive)
Sum_actual_negative=length(Actual_negative)

Sum_predicted_positive=length(Predicted_positive)
Sum_predicted_negative=length(Predicted_negative)

Total_predicted=find(y_tes==S_out_sample)

WWW=y_tes(Total_predicted)
Positive=find(WWW==1)
Negative=find(WWW==0)
 Final_predicted_Positive=length(Positive)/Sum_actual_positive

Final_predicted_Negative=length(Negative)/Sum_actual_negative

Final_predicted_Positive=Final_predicted_Positive*100

Final_predicted_Negative=Final_predicted_Negative*100

Total_performance=((length(Positive) + length(Negative))/t1)*100
% Classification Table
re = '==========================';
li= '--------------------------';
sp = '   ';
disp([re ' Classification Table ' re])
disp([li '         True          ' li])
disp([ '                   1                                 0
Total ' ])
disp([li ' ----------------------' li])
disp(sprintf('              %f:                        %f    %f',
length(Positive_in_sample),...
    Sum_actual_positive_in_sample-length(Positive_in_sample) ))
disp(sprintf('              %f:                       %f    %f',
Sum_actual_negative_in_sample-...
    length(Negative_in_sample) , length(Negative_in_sample)))
```

```
disp([li ' ---------------------' li])
disp(sprintf('Sum             %f:                        %f    %f',
Sum_actual_positive_in_sample,...
    Sum_actual_negative_in_sample, nk))
disp([li ' ---------------------' li])

disp(sprintf('Correct Predict Financial Stage 1 (Crisis)          %f',...
    Final_predicted_Positive_in_sample))

disp(sprintf('Correct Predict Financial Stage 2 (No Crisis)       %f',...
    Final_predicted_Negative_in_sample))

disp(sprintf('Overall Prediction Performance                      %f',...
    Total_performance_in_sample))

disp(blanks(1)')

disp([re ' Classification Table ' re])
disp([li '        True         ' li])
disp([ '                   1                                0
Total ' ])
disp([li ' ---------------------' li])
disp(sprintf('              %f:                        %f    %f',
length(Positive),...
    Sum_actual_positive-length(Positive)))
disp(sprintf('              %f:                        %f    %f',   ...
    Sum_actual_negative-length(Negative), length(Negative)))
disp([li ' ---------------------' li])
disp(sprintf('Sum             %f:                        %f    %f',
Sum_actual_positive ,...
    Sum_actual_negative, t1))
disp([li ' ---------------------' li])

disp(sprintf('Correct Predict Financial Stage 1 (Crisis)          %f',
Final_predicted_Positive))

disp(sprintf('Correct Predict Financial Stage 2 (No Crisis)       %f',
Final_predicted_Negative))

disp(sprintf('Overall Prediction Performance                      %f',
Total_performance))
figure, plot(y,'-r'); hold on; plot(y1,'-b');
xlabel('Periods')
ylabel('Values')
%title('In_sample forecasts')
h1 = legend('Actual','forecasts',1);
figure, plot(y_tes,'-r'); hold on; plot(yf1,'-b');
xlabel('Periods')
ylabel('Values')
%title('Out_of_sample forecasts')
h = legend('Actual','forecasts',1);
figure, plot (index_epochs , array_y );
xlabel('epochs')
ylabel('Error')
title('Number of Epochs')
```

MATLAB routine for ANFIS with two inputs

```
function []=neuro_fuzzy (x,y)
or
%function []=neuro_fuzzy (x,y) for script

clear all;
load file.mat
% data is consisted by 3 columns, the first two concern the input
% variables and the last column the output variable

y=data(1:end-16,end)
x=data(1:end-16,1:2)
% y is the output dummy variable
% x is the matrix of inputs


em=1     % Select the operator, 1 for min which is for AND operator, 2 for
product, which is for
         % AND operator and 3 for for max, which is for OR operator

opt=2   % option 1 is for specific center and bases values of your choice
         % option 2 takes the center and bases values based on mean and
         % standard deviation respectively.

memb=3   % 1 for triangular membership function, 2 for Gaussian

% Set up the learning rates of your choice for center, bases and parameters
r
lr_center_gdp=0.5
lr_center_une=0.5
lr_base_gdp=0.5
lr_base_une=0.5
lr_beta_gdp=1
lr_beta_une=1
lr_r=0.5

% Take the inputs
gdp=x(:,1)
une=x(:,2)

% Take the dimensions of input-output data
[t nj]=size(y)
[nk ni]=size(x)




%First option take specific values of your choice
if opt==1
% Center values for GDP
c_1=-0.01
c_2=0.01
c_3=0.03

%Bases values for GDP
if memb==1
sigma_1=0.04
sigma_2=0.05
sigma_3=0.05
```

```
elseif memb==2
sigma_1=0.025
sigma_2=0.025
sigma_3=0.025

elseif memb==3
sigma1=(max(gdp)-min(gdp))/18

sigma_1=sigma1
sigma_2=sigma1
sigma_3=sigma1

end

% Center values for Unemployment rate
ce_1=5
ce_2=7
ce_3=9
%Bases values for Unemployment rate

if memb==1
sigmae_1=3
sigmae_2=3
sigmae_3=3
elseif memb==2
sigmae_1=2
sigmae_2=2
sigmae_3=2
elseif memb==3
sigma2=(max(une)-min(une))/18
sigmae_1=sigma2
sigmae_2=sigma2
sigmae_3=sigma2

end
% Center values for Industrial production
cindpro_1=-0.02
cindpro_2=0.01
cindpro_3=0.04




b1_gdp=1.1
b2_gdp=1.1
b3_gdp=1.1

b1_une=1.1
b2_une=1.1
b3_une=1.1


% Take center and bases values based on mean and standard deviation
respectively
elseif opt==2
% Values for GDP
y_high=find(gdp>=0.02)
y_medium=find(gdp>=0 & gdp<=0.02)
y_low=find(gdp<=0 )
```

```
% find the values correspond to linguistic terms
y_high=gdp(y_high)
y_medium=gdp(y_medium)
y_low=gdp(y_low)

% Find the mean and simga (standard deviation)
c_1=mean(y_low)
c_2=mean(y_medium)
c_3=mean(y_high)

if memb==1
sigma_1=std(y_low)*2
sigma_2=std(y_medium)*2
sigma_3=std(y_high)*2
elseif memb==2
sigma_1=std(y_low)
sigma_2=std(y_medium)
sigma_3=std(y_high)
elseif memb==3
sigma1=(max(gdp)-min(gdp))/18

sigma_1=sigma1
sigma_2=sigma1
sigma_3=sigma1
end


% In the case you get zero sigma you can write the following. The same is
% followed for the sigma in the ithe inputs
%if sigma_1==0 | sigma_2==0 |sigma_3==0
 %   sigma_1==0.025
%sigma_2==0.025
%sigma_3==0.025
%end


% Values for Unemployment rate
e_high=find(une>=6)
e_medium=find(une>=4.5 & une<=6)
e_low=find(une<=4.5)


% find the values correspond to linguistic terms
e_high=une(e_high)
e_medium=une(e_medium)
e_low=une(e_low)

% Find the mean and simga (standard deviation)
ce_1=mean(e_high)
ce_2=mean(e_medium)
ce_3=mean(e_low)

if memb==1
sigmae_1=std(e_high)*2
sigmae_2=std(e_medium)*2
sigmae_3=std(e_low)*2
elseif memb==2
sigmae_1=std(e_high)
```

```
sigmae_2=std(e_medium)
sigmae_3=std(e_low)
elseif memb==3
sigma2=(max(une)-min(une))/18
sigmae_1=sigma2
sigmae_2=sigma2
sigmae_3=sigma2
end


b1_gdp=1.1
b2_gdp=1.1
b3_gdp=1.1

b1_une=1.1
b2_une=1.1
b3_une=1.1


end

if memb==1
% Take memebership degrees-grades
mf1=trimf(gdp,[-sigma_1/2+c_1 c_1 sigma_1/2+c_1   ]);          % LOW
mf2=trimf(gdp,[-sigma_2/2+c_2 c_2 sigma_2/2+c_2  ]);       % MEDIUM
mf3=trimf(gdp,[-sigma_3/2+c_3 c_3 sigma_3/2+c_3  ]);          % HIGH



mf4=trimf(une,[-sigmae_1/2+ce_1 ce_1 sigmae_1/2+ce_1 ]);       % LOW
mf5=trimf(une,[-sigmae_1/2+ce_2 ce_2 sigmae_1/2+ce_2  ]);    % MEDIUM
mf6=trimf(une,[-sigmae_1/2+ce_3 ce_3 sigmae_1/2+ce_3  ]);      % HIGH


elseif memb==2
mf1=gaussmf(gdp,[sigma_1 c_1  ]);          % LOW
mf2=gaussmf(gdp,[sigma_2 c_2  ]);       % MEDIUM
mf3=gaussmf(gdp,[sigma_3 c_3 ]);          % HIGH



mf4=gaussmf(une,[sigmae_1 ce_1  ]);       % LOW
mf5=gaussmf(une,[sigmae_2 ce_2  ]);    % MEDIUM
mf6=gaussmf(une,[sigmae_3 ce_3  ]);      % HIGH


elseif memb==3

mf1=gbell_mf(gdp,[sigma_1 b1_gdp c_1   ]);          % LOW
mf2=gbell_mf(gdp,[sigma_2 b2_gdp c_2  ]);       % MEDIUM
mf3=gbell_mf(gdp,[sigma_3 b3_gdp c_3  ]);          % HIGH



mf4=gbell_mf(une,[sigmae_1 b1_une ce_1  ]);       % LOW
mf5=gbell_mf(une,[sigmae_2 b2_une ce_2  ]);    % MEDIUM
mf6=gbell_mf(une,[sigmae_3 b3_une ce_3  ]);      % HIGH
```

```
end

if em==1              % Min operator for AND rule
w1=min([mf1 mf4]')  % LOW-LOW
w2=min([mf1 mf5]')  % LOW-MEDIUM
w3=min([mf1 mf6]')  % LOW-HIGH
w4=min([mf2 mf4]')  % MEDIUM-LOW
w5=min([mf2 mf5]')  % MEDIUM-MEDIUM
w6=min([mf2 mf6]')  % MEDIUM-HIGH
w7=min([mf3 mf4]')  % HIGH-LOW
w8=min([mf3 mf5]')  % HIGH-MEDIUM
w9=min([mf3 mf6]')  % HIGH-HIGH


elseif em==2      % product operator for AND rule
w1=(mf1.*mf4)'  % LOW-LOW
w2=(mf1.*mf5)'  % LOW-MEDIUM
w3=(mf1.*mf6)'  % LOW-HIGH
w4=(mf2.*mf4)'  % MEDIUM-LOW
w5=(mf2.*mf5)'  % MEDIUM-MEDIUM
w6=(mf2.*mf6)'  % MEDIUM-HIGH
w7=(mf3.*mf4)'  % HIGH-LOW
w8=(mf3.*mf5)'  % HIGH-MEDIUM
w9=(mf3.*mf6)'  % HIGH-HIGH


elseif em==3      % max operator for OR rule
w1=max([mf1 mf4]')  % LOW-LOW
w2=max([mf1 mf5]')  % LOW-MEDIUM
w3=max([mf1 mf6]')  % LOW-HIGH
w4=max([mf2 mf4]')  % MEDIUM-LOW
w5=max([mf2 mf5]')  % MEDIUM-MEDIUM
w6=max([mf2 mf6]')  % MEDIUM-HIGH
w7=max([mf3 mf4]')  % HIGH-LOW
w8=max([mf3 mf5]')  % HIGH-MEDIUM
w9=max([mf3 mf6]')  % HIGH-HIGH
end



for j=1:t
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& w8(:,j)==0& w9(:,j)==0)
nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;nw9(:,j)=0;
else
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
```

```
nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));

end

end


X_train=[nw1.*gdp';nw2.*gdp';nw3.*gdp';nw4.*gdp';nw5.*gdp';nw6.*gdp';nw7.*g
dp';nw8.*gdp';nw9.*gdp';...
nw1.*une';nw2.*une';nw3.*une';nw4.*une';nw5.*une';
nw6.*une';nw7.*une';nw8.*une';nw9.*une';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9];

% Least square algorithm with SVD
params_train=pinv(X_train')*y
y1=X_train'*params_train;
e=y1-y;
error=sum(sum(e.^2))

bases_1=[sigma_1;sigma_2;sigma_3]
bases_2=[sigmae_1;sigmae_2;sigmae_3]

bases=[bases_1 bases_2 ]

centers_1=[c_1;c_2;c_3]
centers_2=[ce_1;ce_2;ce_3]

centers=[centers_1 centers_2 ]

beta_1=[b1_gdp;b2_gdp;b3_gdp]
beta_2=[b1_une;b2_une;b3_une]

be=[beta_1 beta_2 ]


W=[mf1';mf2';mf3';mf4';mf5';mf6']
W=W'
[nkk nii]=size(X_train)

[nk,ni]=size(x);
[n_rules,ni1]=size(centers);

center_gdp=centers(1:3,1)
center_une=centers(4:6,1)

bases_gdp=bases(1:3,1)
bases_une=bases(4:6,1)

maxepochs=20
SSE_goal=5;
epochs=0
```

```
e.field=e
W.field=W

% Start the error backpropagation algorithm
while (epochs<maxepochs) & (error>SSE_goal)

  for i=1:n_rules/2


if memb==3
eta_base1{i,:}=lr_base_gdp
eta_base2{i,:}=lr_base_une

eta_center1{i,:}=lr_center_gdp
eta_center2{i,:}=lr_center_une

eta_beta1{i,:}=lr_beta_gdp
eta_beta2{i,:}=lr_beta_une

    else
eta_base1{i,:}=lr_base_gdp
eta_base2{i,:}=lr_base_une

eta_center1{i,:}=lr_center_gdp
eta_center2{i,:}=lr_center_une


end
eta_r{:,:}=lr_r
end
if memb==3
eta_base=[eta_base1;eta_base2 ]

eta_center=[eta_center1;eta_center2 ]

eta_beta=[eta_beta1;eta_beta2 ]

else
eta_base=[eta_base1;eta_base2 ]

eta_center=[eta_center1;eta_center2 ]

end
if memb==1
    for i=1:n_rules
    for j=1:ni



    ind_rule{:,i}=find((x(:,j)-centers(i,:))<=(bases(i,:)/2));

    end
end


x.field=x
```

```
 y1.field=y1
params_train.field=params_train
X_train.field=X_train
for kk=1:n_rules

delta_center{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})'*...
    (((2*sign(x.field(ind_rule{:,kk}))-centers(kk,:)))/bases(kk,:)))

delta_base{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})'*...
    (((1-W.field(ind_rule{:,kk})))))/centers(kk,:))';


delta_r{:,:}=(e.field(ind_rule{1,kk}))'*(W.field(ind_rule{:,kk}));

del_center{:,kk}=-((eta_center{kk,:}/(2*nk))*sum(delta_center{:,kk}));


del_base{:,kk}=-((eta_base{kk,:}/(2*nk))*sum(delta_base{:,kk}));

del_r{:,:}=-((eta_r{:,:}/(2*nk))*sum(delta_r{:,:}))

del_center_num(kk,:)=(del_center{:,kk})'

del_base_num(kk,:)=(del_base{:,kk})'

del_r_num(:,:)=(del_r{:,:})'

        end



elseif memb==2
for i=1:n_rules
    for j=1:ni



    ind_rule{:,i}=find((x(:,j)-centers(i,:))<=(bases(i,:)^2/2));

    end
end

x.field=x
y1.field=y1





for kk=1:n_rules

delta_center{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'*...
    ((((x.field(ind_rule{1,kk})-centers(kk,:)))/bases(kk,:)^2))

delta_base{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'*...
    ((((x.field(ind_rule{1,kk})-centers(kk,:))).^2/bases(kk,:)^3))
```

```
delta_r{:,:}=(e.field(ind_rule{1,kk}))'*(W.field(ind_rule{:,kk}));


del_center{:,kk}=(-((eta_center{kk,:})/(2*nk))*sum(delta_center{:,kk}));


del_base{:,kk}=(-((2*eta_base{kk,:})/(2*nk))*sum(delta_base{:,kk}));

del_center_num(kk,:)=(del_center{:,kk})'

del_base_num(kk,:)=(del_base{:,kk})'

del_r{:,:}=-((eta_r{:,:}/(2*nk))*sum(delta_r{:,:}))


del_r_num(:,:)=(del_r{:,:})'
  end

elseif memb==3
for i=1:n_rules
    for j=1:ni



    ind_rule{:,i}=find(exp(-(x(:,j)-centers(i,:)))<=(exp(bases(i,:))));

    end
end

x.field=x
y1.field=y1





for kk=1:n_rules

delta_center{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'*...
    ((2*((x.field(ind_rule{1,kk})-centers(kk,:))/...
    bases(kk,:))'*be(kk,:)))'


delta_base{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'*...
    (2*((x.field(ind_rule{1,kk})-centers(kk,:))/...
    bases(kk,:))'*be(kk,:))'

delta_be{:,kk}=(y1.field(ind_rule{1,kk})'*e.field(ind_rule{1,kk}))*...
    (-2*((x.field(ind_rule{1,kk})-centers(kk,:))/...
    bases(kk,:))'.^2*be(kk,:))

delta_r{:,:}=(e.field(ind_rule{1,kk}))'*(W.field(ind_rule{:,kk}));

del_center{:,kk}=(-((eta_center{kk,:})/(2*nk))*sum(delta_center{:,kk}));


del_base{:,kk}=(-((eta_base{kk,:})/(2*nk))*sum(delta_base{:,kk}));
```

```
del_be{:,kk}=(-((eta_beta{kk,:})/(2*nk))*sum(delta_be{:,kk}));

del_r{:,:}=-((eta_r{:,:}/(2*nk))*sum(delta_r{:,:}))

del_center_num=(del_center{:,kk})'
del_base_num=(del_base{:,kk})'
del_be_num=(del_be{:,kk})'

del_r_num(:,:)=(del_r{:,:})'
    end
  end

if memb==3
centers=centers+del_center_num
bases=bases+del_base_num
be=be+del_be_num

y1.field=y1.field+del_r_num

else
    centers=centers+del_center_num
bases=bases+del_base_num
y1.field=y1.field+del_r_num
end
if memb==3
center_gdp=centers(1:3,1)
center_une=centers(4:6,1)
bases_gdp=bases(1:3,1)
bases_une=bases(4:6,1)
beta_gdp=bases(1:3,1)
beta_une=bases(4:6,1)
centers=[center_gdp;center_une ]

bases=[bases_gdp;bases_une ]
be=[beta_gdp;beta_une]
else
center_gdp=centers(1:3,1)
center_une=centers(4:6,1)
bases_gdp=bases(1:3,1)
bases_une=bases(4:6,1)

centers=[center_gdp;center_une ]

bases=[bases_gdp;bases_une ]
end


if memb==1
mf1=trimf(gdp,[-bases_gdp(1,1)/2+center_gdp(1,1) center_gdp(1,1)...
    bases_gdp(1,1)/2+center_gdp(1,1)  ]);  % LOW
mf2=trimf(gdp,[-bases_gdp(2,1)/2+center_gdp(2,1) center_gdp(2,1)...
    bases_gdp(2,1)/2+center_gdp(2,1) ]);  % LOW
mf3=trimf(gdp,[-bases_gdp(3,1)/2+center_gdp(3,1) center_gdp(3,1)...
    bases_gdp(3,1)/2+center_gdp(3,1)]);   % LOW
```

```
mf4=trimf(une,[-bases_une(1,1)/2+center_une(1,1) center_une(1,1)...
    bases_une(1,1)/2+center_une(1,1)]); % LOW
mf5=trimf(une,[-bases_une(2,1)/2+center_une(2,1) center_une(2,1)...
    bases_une(2,1)/2+center_une(2,1)]); % MEDIUM
mf6=trimf(une,[-bases_une(3,1)/2+center_une(3,1) center_une(3,1)...
    bases_une(3,1)/2+center_une(3,1)]); % HIGH


elseif memb==2
mf1=gaussmf(gdp,[bases_gdp(1,1) center_gdp(1,1)  ]);      % LOW
mf2=gaussmf(gdp,[bases_gdp(2,1) center_gdp(2,1)  ]);   % MEDIUM
mf3=gaussmf(gdp,[bases_gdp(3,1) center_gdp(3,1) ]);      % HIGH


mf4=gaussmf(une,[bases_une(1,1) center_une(1,1)  ]);      % LOW
mf5=gaussmf(une,[bases_une(2,1) center_une(2,1)  ]);   % MEDIUM
mf6=gaussmf(une,[bases_une(3,1) center_une(3,1)  ]);     % HIGH




elseif memb==3
mf1=gbell_mf(gdp,[bases_gdp(1,1) beta_gdp(1,1) center_gdp(1,1)  ]);      %
LOW
mf2=gbell_mf(gdp,[bases_gdp(2,1) beta_gdp(2,1) center_gdp(2,1)  ]);   %
MEDIUM
mf3=gbell_mf(gdp,[bases_gdp(3,1) beta_gdp(3,1) center_gdp(3,1) ]);      %
HIGH


mf4=gbell_mf(une,[bases_une(1,1) beta_une(1,1) center_une(1,1)  ]);      %
LOW
mf5=gbell_mf(une,[bases_une(2,1) beta_une(2,1) center_une(2,1)  ]);   %
MEDIUM
mf6=gbell_mf(une,[bases_une(3,1) beta_une(3,1) center_une(3,1)  ]);     %
HIGH

end

if em==1             % Min operator for AND rule
w1=min([mf1 mf4]')  % LOW-LOW
w2=min([mf1 mf5]')  % LOW-MEDIUM
w3=min([mf1 mf6]')  % LOW-HIGH
w4=min([mf2 mf4]')  % MEDIUM-LOW
w5=min([mf2 mf5]')  % MEDIUM-MEDIUM
w6=min([mf2 mf6]')  % MEDIUM-HIGH
w7=min([mf3 mf4]')  % HIGH-LOW
w8=min([mf3 mf5]')  % HIGH-MEDIUM
w9=min([mf3 mf6]')  % HIGH-HIGH


elseif em==2      % product operator for AND rule
w1=(mf1.*mf4)'  % LOW-LOW
w2=(mf1.*mf5)'  % LOW-MEDIUM
w3=(mf1.*mf6)'  % LOW-HIGH
w4=(mf2.*mf4)'  % MEDIUM-LOW
w5=(mf2.*mf5)'  % MEDIUM-MEDIUM
w6=(mf2.*mf6)'  % MEDIUM-HIGH
w7=(mf3.*mf4)'  % HIGH-LOW
```

```
w8=(mf3.*mf5)'   % HIGH-MEDIUM
w9=(mf3.*mf6)'   % HIGH-HIGH


elseif em==3       % max operator for OR rule
w1=max([mf1 mf4]')   % LOW-LOW
w2=max([mf1 mf5]')   % LOW-MEDIUM
w3=max([mf1 mf6]')   % LOW-HIGH
w4=max([mf2 mf4]')   % MEDIUM-LOW
w5=max([mf2 mf5]')   % MEDIUM-MEDIUM
w6=max([mf2 mf6]')   % MEDIUM-HIGH
w7=max([mf3 mf4]')   % HIGH-LOW
w8=max([mf3 mf5]')   % HIGH-MEDIUM
w9=max([mf3 mf6]')   % HIGH-HIGH
end



for j=1:t
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& w8(:,j)==0& w9(:,j)==0)
nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;nw9(:,j)=0;
else
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));

end

end

X_train=[nw1.*gdp';nw2.*gdp';nw3.*gdp';nw4.*gdp';nw5.*gdp';nw6.*gdp';nw7.*g
dp';nw8.*gdp';nw9.*gdp';...
nw1.*une';nw2.*une';nw3.*une';nw4.*une';nw5.*une';
nw6.*une';nw7.*une';nw8.*une';nw9.*une';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9];
x=x.field

params_train=pinv(X_train')*y % [p1 p2 q1 q2 s1 s2 r1 r2....pn qn sn rn]
y1=X_train'*params_train;
e=y1-y;
error=sum(sum(e.^2))
e.field=e
epochs=epochs+1
```

```
array_y ( epochs )= error;
index_epochs ( epochs ) = epochs;
end

antecedent_par=[nw1',nw2',nw3',nw4',nw5',nw6',nw7',nw8',nw9']


consequent_par=y1

output_2=sum(antecedent_par'*consequent_par)/sum(sum(antecedent_par))


out=output_2


for kkk=1:nk
    if y1(kkk,:)>out
        S_in_sample(kkk,:)=1

    elseif y1(kkk,:)<out
        S_in_sample(kkk,:)=0
    end
end

% Compuation for clasiffication table preparation
Actual_positive_in_sample=find(y==1)
Actual_negative_in_sample=find(y==0)

Predicted_positive_in_sample=find(S_in_sample==1)
Predicted_negative_in_sample=find(S_in_sample==0)

Sum_actual_positive_in_sample=length(Actual_positive_in_sample)
Sum_actual_negative_in_sample=length(Actual_negative_in_sample)

Sum_predicted_positive_in_sample=length(Predicted_positive_in_sample)
Sum_predicted_negative_in_sample=length(Predicted_negative_in_sample)

Total_predicted_in_sample=find(y==S_in_sample)

Perc=(length(Total_predicted_in_sample)/nk)*100

W_in_sample=y(Total_predicted_in_sample)

Positive_in_sample=find(W_in_sample==1)
Negative_in_sample=find(W_in_sample==0)


Final_predicted_Positive_in_sample=length(Positive_in_sample)/Sum_actual_po
sitive_in_sample

Final_predicted_Negative_in_sample=length(Negative_in_sample)/Sum_actual_ne
gative_in_sample

Final_predicted_Positive_in_sample=Final_predicted_Positive_in_sample*100

Final_predicted_Negative_in_sample=Final_predicted_Negative_in_sample*100
```

```
Total_performance_in_sample=((length(Positive_in_sample) +
length(Negative_in_sample))/nk)*100


clear nw1
clear nw2
clear nw3
clear nw4
clear nw5
clear nw6
clear nw7
clear nw8
clear nw9


load file.mat
y_tes=data(end-15:end,end)
x_tes=data(end-15:end,1:2)
gdp=x_tes(:,1)
une=x_tes(:,2)
t1=length(y_tes)

if memb==1
mf1=trimf(gdp,[-bases_gdp(1,1)/2+center_gdp(1,1) center_gdp(1,1)...
    bases_gdp(1,1)/2+center_gdp(1,1)  ]);  % LOW
mf2=trimf(gdp,[-bases_gdp(2,1)/2+center_gdp(2,1) center_gdp(2,1)...
    bases_gdp(2,1)/2+center_gdp(2,1) ]);  % LOW
mf3=trimf(gdp,[-bases_gdp(3,1)/2+center_gdp(3,1) center_gdp(3,1)...
    bases_gdp(3,1)/2+center_gdp(3,1)]);   % LOW



mf4=trimf(une,[-bases_une(1,1)/2+center_une(1,1) center_une(1,1)...
    bases_une(1,1)/2+center_une(1,1)]); % LOW
mf5=trimf(une,[-bases_une(2,1)/2+center_une(2,1) center_une(2,1)...
    bases_une(2,1)/2+center_une(2,1)]); % MEDIUM
mf6=trimf(une,[-bases_une(3,1)/2+center_une(3,1) center_une(3,1)...
    bases_une(3,1)/2+center_une(3,1)]); % HIGH





elseif memb==2
mf1=gaussmf(gdp,[bases_gdp(1,1) center_gdp(1,1)  ]);      % LOW
mf2=gaussmf(gdp,[bases_gdp(2,1) center_gdp(2,1)  ]);   % MEDIUM
mf3=gaussmf(gdp,[bases_gdp(3,1) center_gdp(3,1) ]);      % HIGH


mf4=gaussmf(une,[bases_une(1,1) center_une(1,1)  ]);      % LOW
mf5=gaussmf(une,[bases_une(2,1) center_une(2,1)  ]);   % MEDIUM
mf6=gaussmf(une,[bases_une(3,1) center_une(3,1)  ]);     % HIGH




elseif memb==3
```

```
mf1=gbell_mf(gdp,[bases_gdp(1,1) beta_gdp(1,1) center_gdp(1,1)  ]);      %
LOW
mf2=gbell_mf(gdp,[bases_gdp(2,1) beta_gdp(2,1) center_gdp(2,1)  ]);   %
MEDIUM
mf3=gbell_mf(gdp,[bases_gdp(3,1) beta_gdp(3,1) center_gdp(3,1) ]);      %
HIGH


mf4=gbell_mf(une,[bases_une(1,1) beta_une(1,1) center_une(1,1)  ]);      %
LOW
mf5=gbell_mf(une,[bases_une(2,1) beta_une(2,1) center_une(2,1)  ]);   %
MEDIUM
mf6=gbell_mf(une,[bases_une(3,1) beta_une(3,1) center_une(3,1)  ]);     %
HIGH


end

if em==1              % Min operator for AND rule
w1=min([mf1 mf4]')  % LOW-LOW
w2=min([mf1 mf5]')  % LOW-MEDIUM
w3=min([mf1 mf6]')  % LOW-HIGH
w4=min([mf2 mf4]')  % MEDIUM-LOW
w5=min([mf2 mf5]')  % MEDIUM-MEDIUM
w6=min([mf2 mf6]')  % MEDIUM-HIGH
w7=min([mf3 mf4]')  % HIGH-LOW
w8=min([mf3 mf5]')  % HIGH-MEDIUM
w9=min([mf3 mf6]')  % HIGH-HIGH


elseif em==2      % product operator for AND rule
w1=(mf1.*mf4)'  % LOW-LOW
w2=(mf1.*mf5)'  % LOW-MEDIUM
w3=(mf1.*mf6)'  % LOW-HIGH
w4=(mf2.*mf4)'  % MEDIUM-LOW
w5=(mf2.*mf5)'  % MEDIUM-MEDIUM
w6=(mf2.*mf6)'  % MEDIUM-HIGH
w7=(mf3.*mf4)'  % HIGH-LOW
w8=(mf3.*mf5)'  % HIGH-MEDIUM
w9=(mf3.*mf6)'  % HIGH-HIGH


elseif em==3      % max operator for OR rule
w1=max([mf1 mf4]')  % LOW-LOW
w2=max([mf1 mf5]')  % LOW-MEDIUM
w3=max([mf1 mf6]')  % LOW-HIGH
w4=max([mf2 mf4]')  % MEDIUM-LOW
w5=max([mf2 mf5]')  % MEDIUM-MEDIUM
w6=max([mf2 mf6]')  % MEDIUM-HIGH
w7=max([mf3 mf4]')  % HIGH-LOW
w8=max([mf3 mf5]')  % HIGH-MEDIUM
w9=max([mf3 mf6]')  % HIGH-HIGH
end

for j=1:t1
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& w8(:,j)==0& w9(:,j)==0)
nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;nw9(:,j)=0;
```

```
else
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));
nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7(:,j)+
w8(:,j)+w9(:,j));

end

end





X_test=[nw1.*gdp';nw2.*gdp';nw3.*gdp';nw4.*gdp';nw5.*gdp';nw6.*gdp';nw7.*gd
p';nw8.*gdp';nw9.*gdp';...
nw1.*une';nw2.*une';nw3.*une';nw4.*une';nw5.*une';
nw6.*une';nw7.*une';nw8.*une';nw9.*une';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9];

yf1=X_test'*params_train;
for kkk=1:t1
    if yf1(kkk,:)>out
        S_out_sample(kkk,:)=1

    elseif yf1(kkk,:)<out
        S_out_sample(kkk,:)=0
    end
end

Actual_positive=find(y_tes==1)
Actual_negative=find(y_tes==0)

Predicted_positive=find(S_out_sample==1)
Predicted_negative=find(S_out_sample==0)

Sum_actual_positive=length(Actual_positive)
Sum_actual_negative=length(Actual_negative)

Sum_predicted_positive=length(Predicted_positive)
Sum_predicted_negative=length(Predicted_negative)

Total_predicted=find(y_tes==S_out_sample)
```

```
WWW=y_tes(Total_predicted)

Positive=find(WWW==1)
Negative=find(WWW==0)
Final_predicted_Positive=length(Positive)/Sum_actual_positive

Final_predicted_Negative=length(Negative)/Sum_actual_negative

Final_predicted_Positive=Final_predicted_Positive*100

Final_predicted_Negative=Final_predicted_Negative*100

Total_performance=((length(Positive) + length(Negative))/t1)*100
% Classification Table
re = '=========================';
li= '-------------------------';
sp = '   ';
disp([re ' Classification Table ' re])
disp([li '         True         ' li])
disp([ '                   1                                0
Total ' ])
disp([li ' ---------------------' li])
disp(sprintf('               %f:                        %f    %f',
length(Positive_in_sample),...
    Sum_actual_positive_in_sample-length(Positive_in_sample) ))
disp(sprintf('               %f:                       %f    %f',
Sum_actual_negative_in_sample-...
    length(Negative_in_sample) , length(Negative_in_sample)))
disp([li ' ---------------------' li])
disp(sprintf('Sum            %f:                       %f    %f',
Sum_actual_positive_in_sample,...
    Sum_actual_negative_in_sample, nk))
disp([li ' ---------------------' li])

disp(sprintf('Correct Predict Financial Stage 1 (Crisis)          %f',...
    Final_predicted_Positive_in_sample))

disp(sprintf('Correct Predict Financial Stage 2 (No Crisis)       %f',...
    Final_predicted_Negative_in_sample))

disp(sprintf('Overall Prediction Performance                      %f',...
    Total_performance_in_sample))

disp(blanks(1)')

disp([re ' Classification Table ' re])
disp([li '         True         ' li])
disp([ '                   1                                0
Total ' ])
disp([li ' ---------------------' li])
disp(sprintf('               %f:                        %f    %f',
length(Positive),...
    Sum_actual_positive-length(Positive)))
disp(sprintf('               %f:                       %f    %f',  ...
    Sum_actual_negative-length(Negative), length(Negative)))
disp([li ' ---------------------' li])
disp(sprintf('Sum            %f:                       %f    %f',
Sum_actual_positive ,...
```

```
    Sum_actual_negative, t1))
disp([li ' ----------------------' li])

disp(sprintf('Correct Predict Financial Stage 1 (Crisis)          %f',
Final_predicted_Positive))

disp(sprintf('Correct Predict Financial Stage 2 (No Crisis)       %f',
Final_predicted_Negative))

disp(sprintf('Overall Prediction Performance                      %f',
Total_performance))

figure, plot(y,'-r'); hold on; plot(y1,'-b');
xlabel('Periods')
ylabel('Values')
%title('In_sample forecasts')
h1 = legend('Actual','forecasts',1);
figure, plot(y_tes,'-r'); hold on; plot(yf1,'-b');
xlabel('Periods')
ylabel('Values')
%title('Out_of_sample forecasts')
h = legend('Actual','forecasts',1);
figure, plot (index_epochs , array_y );
xlabel('epochs')
ylabel('Error')
title('Number of Epochs')



function y = gbell_mf(x, params)

a = params(1); b = params(2); c = params(3);


y = exp(-((x - c)/a).^2*b);
```