Proceedings of the World Congress on Engineering 2007 Vol I
WCE 2007, July 2 - 4, 2007, London, U.K.

# Modified Memory Convergence with Fuzzy PSO

Subhash Chandra Pandey*, P.K Mishra**
* United College of Engineering & Research, Allahabad, India, subh63@yahoo.co.in
** Motilal Nehru National Institute of Technology, Allahabad, India, pkm@mnnit.ac.in

***Abstract*—Associative neural memories are models of biological phenomena that allow for the storage of pattern associations and the retrieval of the desired output pattern upon presentation of a possibly noisy or incomplete version of an input pattern. In this paper, we introduce fuzzy swarm particle optimization technique for convergence of associative neural memories based on fuzzy set theory. An FPSO consists of clustering of swarm's particle by applying fuzzy c-mean algorithm to attain the neighborhood best. We present a singular value decomposition method for the selection of efficient rule from a given rule base required to attain the global best. Finally, we illustrate the proposed method by virtue of some examples.**

***Index Terms*—Artificial neural network, Convergence, , Fuzzy c-mean, Particle swarm optimization, Singular value decomposition.**

## I. Introduction

An artificial neural network (ANN) is an analysis paradigm that is a simple model of the brain and the back-propagation algorithm is the one of the most popular method to train the artificial neural network. Recently there have been significant research efforts to apply evolutionary computation techniques for the purposes of evolving one or more aspects of artificial neural networks.

The efficient supervised training of feedforward neural networks (FNNs) is a subject of considerable ongoing research and numerous algorithms proposed to this end. The back propagation (BP) algorithm [1] is one of the most common supervised training methods. Although BP training has proved to be efficient in many applications, its convergence tends to be slow, and yields to suboptimal solutions [2].

Attempts to speed up training and reduce convergence to local minima have been made in the context of gradient descent [3, 4, 5 ]. However, these methods are based on variable weight, learning rate, step size and bias to dynamically adapt BP algorithm, and use a constant gain for any sigmoid function during its training cycle.

Evolutionary computation methodologies have been applied to three main attributes of neural networks: network connection weights, network architecture (network topology, transfer function), and network learning algorithm.

Particle swarm optimization (PSO) is a population based stochastic optimization technique [6,7] inspired by social behavior of bird flocking or fish schooling. This is modeled by particles in multidimensional space that have a position and a velocity. These particles are flying through a hyperspace and have two essential reasoning capabilities: the memory of their own best position and knowledge of the swarm's best, best simply meaning the position with the smallest objective function value. Members of a swarm communicate good positions to each other and adjust their own position and velocity based on good positions. There are two main ways this communication is done:

- A global best that is known to all and immediately updated when a new best position is found by any particle in the swarm.
- A "Neighborhood" best where each particle only immediately communicates with a subset of the swarm about best positions.

The remainder of this paper is organized as follows. In Section II, preliminaries of particle swarm optimization are presented. Fuzzy c-mean algorithm is developed for swarm's clustering to attain the neighborhood best in Section III. Singular value decomposition has been proposed for approaching the global best in Section IV. An illustrated example has shown in Section V. Finally, we make concluding remarks in Section VI.

## II. The Particle swarm Optimization

PSO's precursor was a simulator of the social behavior that was used to visualize the movement of a birds' flock. Several version of the simulation model were developed, incorporating concepts such as nearest neighbor velocity matching and acceleration by distance [6,8]. Two variants of the PSO algorithm were developed. One with a global neighborhood and another with local neighborhood [9].

Suppose that the search space is $D$-dimensional, and then the $i^{th}$ particle of the swarm can be represented by a $D$-dimensional vector $X_i = (x_i1, x_i2, ......x_iD)$. The velocity (position change) of this particle can be represented by another $D$-dimensional vector $V_i = (v_i1, v_i2, ......v_iD)$. The best previously visited position of the $i^{th}$ particle is denoted as $P_i = (p_i1, p_i2, ......p_iD)$. Defining $(C_1, C_2, ......, C_k)$

as the index of the best particle in the swarm (i.e. $g^{th}$ particle is the best), and let the superscript denote the iterative number, then the swarm is manipulated according to the following two equations [6]:

$$Z_{id}^{n+1} = Z_{id}^{n} + C_a r_1^n (p_{id}^n - x_{id}^n) + C_a r_2^n (p_{gd}^n - x_{id}^n) \quad (1)$$

$$x_{id}^{n+1} = x_{id}^{n} + Z_{id}^{n+1} \quad (2)$$

Where $d = 1,2,....D$ ; $i = 1,2,....N$ and $N$ is the size of the swarm; $C_a$ is a positive constant called, *acceleration constant* $r1, r2$ are the random numbers, uniformly distributed in [0,1]; and $n = 1,2,....$ determines the iteration numbers.

Equations (1) and (2) define the initial version of the PSO algorithm. Since there was no actual mechanism for controlling the velocity of a particle, it was necessary to impose a maximum value $V_{\max}$ on it. If the velocity exceeded this threshold, it was set equal to $V_{\max}$. This parameter is proved to be crucial, because large values could results in the particles moving past good solutions, while small values could result in insufficient exploration of the search space. This lack of control mechanism for the velocity resulted in low efficiency for PSO.

Various attempts have been made to improve the performance of the base line PSO with varying success. Eberhart and Shi focus on optimizing the update equations for the particles [9]. Angeline used a selection mechanism in an attempt to improve the general quality of the particles in swarm. Kennedy uses cluster analysis to modify the update equation, so that particles attempt to confirm to the centre of their clusters rather than attempting to conform to a global best.

The aforementioned problem was addressed by incorporating a weight parameter for the previous velocity of the particle. Thus in the latest version of the PSO, Equations (1) and (2) are changed to the following ones [10]:

$$Z_{id}^{n+1} = x(W Z_{id}^n + C_{a1} r_1^n (p_{id}^n - x_{id}^n) + C_{a2} r_2^n (p_{id}^n - x_{id}^n))$$

$$x_{id}^{n+1} = x_{id}^{n} + Z_{id}^{n+1}$$

In our proposed model both the approaches have been consider together. First, we clustered the swarm by applying fuzzy c-mean algorithm to attain the neighborhood best and then we reduce the number of rules required to attain the global best by virtue of singular value decomposition method.

## III. NEIGHBORHOOD BEST USING FCM

The Fuzzy c – Means algorithm generalizes the hard c – means algorithm to allow a particle of swarm to partially belong to a multiple clusters. Therefore, it produces a soft partition for a given swarm. To do this, the objective function $J$ of hard c-means has been extended in two ways.

- The fuzzy membership degrees in clusters were incorporated into the formula, and
- An additional parameter $p$ is introduced as a weight exponent in the fuzzy membership.

The extended objective function, denoted J, is

$$J(P,V) = \sum_{i=1}^{k} \sum_{x_k \in X} (\mu_{C_i}(x_k))^p \left\| x_k - v_i \right\|^2$$

Where $p$ is a fuzzy partition of the swarm $X$ formed by $C_1, C_2, ......, C_k$. The parameter $p$ is a weight that determines the degree to which partial members of a cluster affect the clustering result.

Theorem3.1 A constrained fuzzy partition $(C_1, C_2, ......, C_k)$ can be a local minimum of the objective function J only if the following conditions are satisfied:

$$\mu_{C_i}(x) = 1 \Big/ \sum_{j=1}^{k} \left( \left\| x - v_i \right\|^2 \Big/ \left\| x - v_j \right\|^2 \right)^{\frac{1}{p-1}}$$

$$1 \le i \le k, x \in X \quad (3)$$

$$v_i = \sum_{x \in X} (\mu_{C_i}(x))^p \times x \Big/ \sum_{x \in X}^{n} (\mu_{C_i}(x))^p \qquad 1 \le i \le k \quad (4)$$

Based on this theorem, FCM updates the prototypes and the membership function iteratively using (3) and (4) until a convergence criterion is reached.

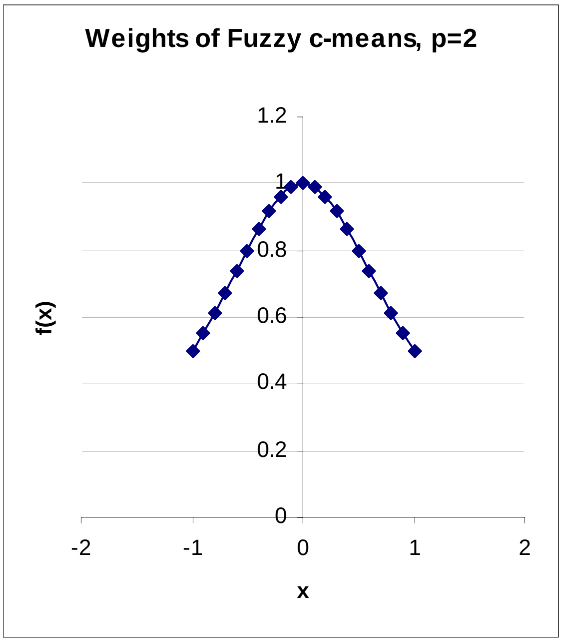
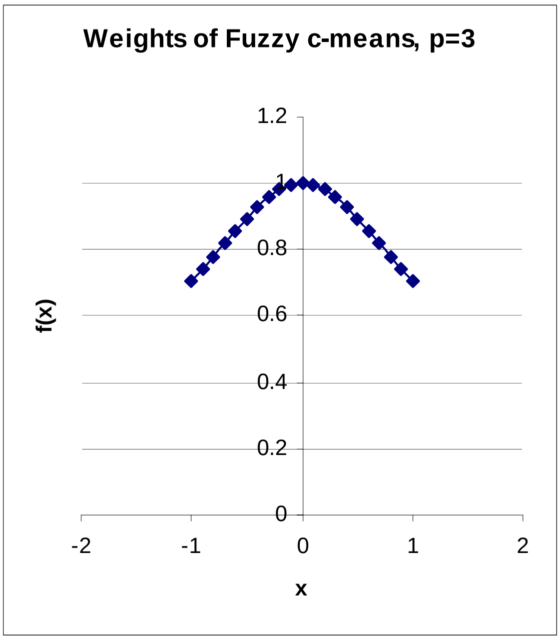



Fig.1 (a) Weight of the FCM algorithm for p = 2




F

Fig1 (b) Weight of the FCM algorithm for p = 3

The algorithm of FCM can be described as:
FCM ($X, c, m, \varepsilon$)

$X$ : an unlabeled swarm size
$C$ : the number of clusters to form
$p$ : the parameter in objective function
$\varepsilon$ : a threshold for the convergence criteria.
Initial prototype $V = \{v_{1,} v_2, ......., v_c\}$

Repeat
$V^{previous} \leftarrow V$
Compute membership function using (4)
Update the prototype, $v_i$ in $V$ using (3)
Until $\sum_{i=1}^{c} \left\| v_i^{previous} - v_i \right\| \leq \varepsilon$

## IV. GLOBAL BEST USING SVD

In our proposed model, after clustering the particles of swarm, orthogonal transformation method is used for selecting important fuzzy rules from a given rule base [11, 12, 13, 14 ]. Unlike conventional methods where multiple iterations are usually required to find "optimal" number of fuzzy rules, orthogonal transformation methods are a non iterative procedure. Therefore, orthogonal transformation methods are computationally less expensive compared to the conventional methods especially when the numbers of particles in the swarm are too large. In this section we introduce how to use singular value decomposition (SVD) to select the most important fuzzy rules from a given rule base and construct compact fuzzy models with better generalization ability.

Singular value decomposition takes a rectangular $n$-by-$p$ matrix $A$, in which the n rows represents the genes and the columns represents the experimental condition [15]. The SVD theorem states:

$$A_{n\times p} = U_{n\times n} S_{n\times p} V^T{}_{p\times p}$$

Where $U^T U = I_{n\times n}$

$V^T V = I_{p\times p}$ (i.e. $U$ and $V$ are orthogonal)

$S$ = diag $(\sigma_{1,} \sigma_2, ....., \sigma_m) \in R^{n\times p} (m = \min\{n, p\})$ is a diagonal matrix with $\sigma_1 \geq \sigma_2 \geq ..... \geq \sigma_m \geq 0$. The columns of $U$ are the left singular vectors has singular values and is diagonal (mode amplitudes); and $V^T$ has rows that are the right singular vectors (expression level vectors).

In the basic principle of using SVD for fuzzy rule selection, we can use fuzzy model with constant consequent constituents as an example. This type of fuzzy model, which is usually referred to as the zero order TSK model, has the following form [16].

: $R_i$ : If $x_1$ is $A_{i1}$ and $x_2$ is $A_{i2}$ and ........$x_m$ is $A_{im}$

*Then* $y$ is $C_i$; $i = 1, 2, ....M$ .

Where $C_i$ is the constant constituents. The total output of the model is computed by.

$$Y = \sum_{i=1}^{M} w_i c_i \Big/ \sum_{i=1}^{M} w_i$$

Where $w_i$ is the matching degree.

The SVD starts with an oversized rule base and then remove redundant or less important fuzzy rules through a "one pass" operation. Finally the efficient rule obtained is obeyed by all the swarm's cluster to approach the global best. In next section we will illustrate the method by taking few examples.

## V. ILLUSTRATIVE EXAMPLES

Suppose we are given a swarm of size six particle, each of which has two features $F_1$ and $F_2$.We list the particle in given table. Assuming that we want to use FCM to partition the swarm in two clusters [16], suppose we set the parameter $p$ in FCM at 2, and the initial prototypes to $v_1$ = (5,5) $v_2$ = ( 10,10 ).

Table1. A swarm to be partitioned

| | $F_1$ | $F_2$ |
|---|---|---|
| X1 | 2 | 12 |
| X2 | 4 | 9 |
| X3 | 7 | 13 |
| X4 | 11 | 5 |
| X5 | 12 | 7 |
| X6 | 14 | 4 |

The initial membership functions of the two clusters are calculated using (3):

$$\mu_{c_1}(x_1) = 1 \Big/ \sum_{j=1}^{2} \left( \|x_1 - v_1\| / \|x_1 - v_j\| \right)^2$$

$$\|x_1 - v_1\|^2 = 3^2 + 7^2 = 9 + 49 = 58$$

$\|x_1 - v_2\|^2$ = $8^2 + 2^2$ = 64 + 4 = 68

$\mu_{C_1}(x_1)$ = 1/ [(58/58) + (58/68)]

Similarly, we obtain the following:

$\mu_{C_2}(x_1)$ = 1/[(68/58)+(68/68)] = 0.4603

$\mu_{C_1}(x_2)$ = 1/[(17/17)+(17/37)] = 0.6852

$\mu_{C_2}(x_2)$ = 1/ [(37/17) + (37/37)] = 0.3148

$\mu_{C_1}(x_3)$ = 1/[(68/68)+(68/18)] = 0.2093

$\mu_{C_2}(x_3)$ = 1/ [(18/68) + ((18/18)] = 0.7907

$\mu_{C_1}(x_4)$ = 1/ [(36/36) + (36/26)] = 0.4194

$\mu_{C_2}(x_4)$ = 1/ [(26/36) + (26/26)] = 0.5806

$\mu_{C_1}(x_5)$ = 1/ [(53/53) + (53/13)] = 0.197

$\mu_{C_2}(x_5)$ = 1/ [(13/53) + (13/13)] = 0.803

$\mu_{C_1}(x_6)$ = 1/ [(82/82) + (82/52)] = 0.3881

$\mu_{C_2}(x_6)$ = 1/ [(52/82) + (52/52)] = 0.6119

Therefore, using these initial prototypes of the two clusters, the membership function indicates that $x_1$ and $x_2$ are more in the first cluster, while the remaining particles in the swarm are more in the second cluster.

The FCM algorithm then updates the prototypes according to (4).

$$v_1 = \sum_{k=1}^{6} (\mu_{C_1}(x_k))^2 \times x_k \Big/ \sum_{k=1}^{6} (\mu_{C_i}(x_k))^2$$

= [$0.5397^2$×(2,12)+$0.6852^2$×(4,9)+$0.2093^2$×(7,13)+$0.4194^2$×(11,5)+$0.197^2$×(12,7)+$0.3881^2$×(14,4)] / [$0.5397^2$+$0.6852^2$+$0.2093^2$+$0.4194^2$+$0.197^2$+$0.3881^2$]

= [(7.2761/1.0979),(10.044/1.0979)]
= (6.6273, 9.1484)

$$v_2 = \sum_{k=1}^{6} (\mu_{C_2}(x_k))^2 \times x_k \Big/ \sum_{k=1}^{6} (\mu_{C_2}(x_k))^2$$

= [$0.4603^2$×(2,12)+$0.3148^2$×(4,.9)+$0.7909^2$×(7,13)+$0.5806^2$×(11,5)+$0.803^2$×(12,7)+$0.6119^2$×(14,4)] / [$0.4603^2$+$0.3148^2$+$0.7909^2$+$0.5806^2$+$0.803^2$+$0.6119^2$]

= [(22.326/2.2928),(19.4629/2.2928)]
= (9.7374, 8.4887)

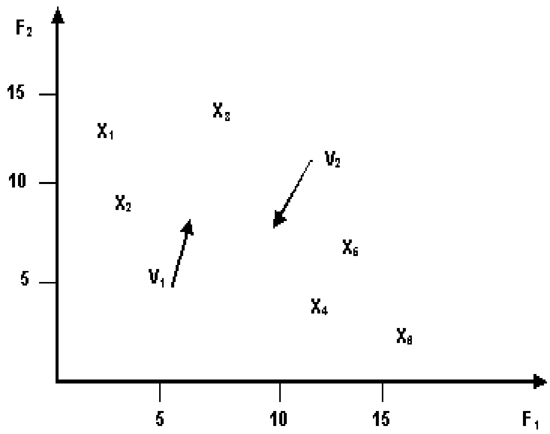



Fig.2 An example of Fuzzy c-mean Algorithm

The update prototype $v_1$ as shown in Fig2, is moved closer to the center of the cluster formed by $x_1, x_2$ and $x_3$; while the updated prototype $v_2$ is moved closer to the cluster formed by $x_4$, $x_5$ and $x_6$.

Now we illustrate how to solve for SVD to obtain efficient rule for approaching the global best, let's take the example of the matrix.

$$A = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

For a $n \times n$ matrix $W$, the nonzero vector $X$ is the eigenvector of $W$ W if: $W\ X = \lambda\ X$, $\lambda$ is the eigenvalue of $A$ and $X$ is the eigenvector of $A$ corresponding to $\lambda$. So to find the eigenvalues of the entity we compute matrices $AA^T$ and $A^T A$. The eigenvectors of $AA^T$ make up the columns of $U$ so we can do the following analysis to find $U$.

$$AA^T = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 4 & 0 & 0 \\ 1 & 3 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 20 & 14 & 0 & 0 \\ 14 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = W$$

Since $WX = \lambda X$ then $(W - \lambda I)X = 0$.

Hence

$$\begin{bmatrix} 20-\lambda & 14 & 0 & 0 \\ 14 & 10-\lambda & 0 & 0 \\ 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & -\lambda \end{bmatrix} X = (W - \lambda I)X = 0$$

Thus, from the solution of characteristic equation, we obtain $\lambda$=0, $\lambda$=0, $\lambda$ = 15+14.81, $\lambda$ = 15-14.81. This value can be used to determine the eigenvector that can be placed in the columns of $U$. Thus, we obtain the following equations.

19.883 $X1$ +14 $X2$ = 0; 14 $X1$ + 9.883 $X2$ = 0; $X3$=0; $X4$=0

Upon simplifying the first two equations we obtain a ratio which relates the value of $X1$ and $X2$. The values of $X1$ and $X2$ are chosen such that the elements of S are the square roots of the eigenvalues. Thus a solution that satisfies the above equation $X1$=-0.58 and $X2$=0.82 and $X3 = X4$=0 (this is the second column of the $U$ matrix). Substituting the other eigenvalues we obtain:

-9.883 $X1$ + 14 $X2$ = 0; 14 $X1$ -19.883 $X2$ = 0; $X3$ = 0; $X4$=0 .Thus a solution that satisfies this set of equations is $X1$ = 0.82 and $X2$ = -0.58 and $X3$ = $X4$ = 0 (this is the first column of the $U$ matrix ). Combining these we obtain:

$$U = \begin{bmatrix} 0.82 & -0.58 & 0 & 0 \\ 0.58 & 0.82 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Similarly, we can find the value of $V$

$$A^T A = \begin{bmatrix} 2 & 4 & 0 & 0 \\ 1 & 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Similarly, we can obtain the expression

$$V = \begin{bmatrix} 0.40 & -0.91 \\ 0.91 & 0.40 \end{bmatrix}$$

Finally, the $S$ is square root of the eigenvalues from $AA^T$ or $A^T A$ and can be obtained directly giving us:

$$S = \begin{bmatrix} 5.47 & 0 \\ 0 & 0.37 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

It is obvious that $\sigma_1 \triangleright \sigma_2 \triangleright \sigma_3$..... .. This is what the paper was indicating.

## VI. Conclusion

In this research, a new approach is proposed for the convergence of associative neural memories by using the Fuzzy Particle Swarm Optimization technique (FPSO). The approach focuses on the neighborhood best and global best to increase the speed of convergence. In addition, this proposed model overcomes the local minima problem which is major issue with the PSO technique.

The illustrated examples suggest that our new approach can be used successfully as real time memory convergence technique for the artificial neural network.

## References

[1] Armij, L, “Minimization of functions having Lischitz continuous first partial derivatives,”Pacific journal of mathematics,Vol16,pp.1-3,1966.

[2] Baladi, and Hornik, K, “Neural networks and principal component analysis: learning from examples and local minima,” Neural networks, Vol2, pp.53-58,1989.

[3] Bello., M.G, “Enhanced training algorithms and intwgrated training / architecture selection for multi layer perceptron networks,” IEEE Transaction on Neural Networks, Vol3, pp.864-874,1992.

[4] Antsaklis, P.J, and Moody, J.O, “The Dependence identification Neural Network Construction Algorithm,” IEEE Transaction on Neural Networks, Vol1, pp3-15, January 1996.

[5] Baron Robert, H, and Coughlin, James P, “ Neural Computation in Hopfield Networks and Boltzmann Machines,” Newark: University of Delaware Press,1994.

[6] Kennedy, J, and Eberhart, R. C, “Particle swarm optimization,” Proc. IEEE int’l conf. On neural networks, Vol.4, pp.1942-1948, 1995.

[7] Eberhart, R.C, and Shi, Y, “Particle swarm optimization: Development, applications and resources, “Proc. Congress on evolutionary computation, IEEE service centre, Korea, 2001.

[8] Kennedy, J, and Eberhart, R, “ Swarm Intelligence,” Morgan KaufmannPublisers, 2001.

[9] Shi, Y, and Eberhart, R, “Parameter selection in Particle Swarm Optimization,” IEEE International Conference on Evolutionary Computations,” Lecture netes in computer science, Springer, pp.591-600,1998.

[10] Shi, Y, and Eberhart, R, “Amodified Particle Swarm Optimizer,” IEEE International conference on evolutionary computation, Anchorage, Alaska, 1998.

[11] Mouzouris, G. C, and Mendel, J.M, “Designing fuzzy logic systems for uncertain environments using a singular-value- QR decomposition method,” Proceedings of the fifth IEEE International conference on fuzzy systems, New Orleans, pp295-301, 1996.

[12] Yen, J, and Wang, L, “An SVD based fuzzy modelreduction strategy, ” Proceeding of the fifth IEEE International conference on fuzzy system, New Orleans, pp. 835-841, 1996.

[13] Yen, J, and Wang, L, “Granule-based Models,” Handbook of fuzzy computation,” IOP Publishing, 1998.

[14] Yen, J,and Wang, L, “Simplifying fuzzy rule-based models using orthogonal transformation methods,” IEEE Transaction on Systems, man, and Cybernatics, Vol-29, 1999.

[15] Alter, O, Brown, O.O, and Botstein, D, “Singular value decompositin for genome-wide expression data processing and modeling,” Proc. Natl Acad Sci USA, 2000.

[16] Yen, J, and Langari, R, “Fuzzy logic: Intelligence,control, and information,” Pearson edition, 2004.