A Hybrid Genetic Algorithm and Particle Swarm Optimization based Fuzzy Times Series Model for TAIFEX and KSE-100 Forecasting

TAHSEEN A. JILANI, USMAN AMJAD, NIKOS MASTORAKIS
Department of Computer Science
University of Karachi
Karachi-75270
PAKISTAN.
Faculty of Engineering
Technical University of Sofia
BALGARIA.
tahseenjilani@uok.edu.pk, usmanamjad87@gmail.com, mastor@wseas.org

Abstract: - In this paper we proposed a new evolutionary fuzzy time series forecasting model for Taiwan Futures Exchange (TIAFEX) and Karachi Stock Exchange (KSE-100) forecasting. Our proposed method is based on two-factor high order fuzzy logical relation groups. A hybrid algorithm composed of genetic algorithm (GA) and Particle swarm optimization (PSO) is used to adjust interval length in universe of discourse for TAIFEX and KSE-100 forecasting with the objective of increasing forecasting accuracy and minimizing forecasting error rate.

Keywords: - Fuzzy time series; two-factors high-order fuzzy logical relationships; Genetic Algorithm, Particle Swarm Optimization; TAIFEX; KSE-100 index.

1 Introduction

Forecasting for the unseen future events to be able to cope with the forthcoming situations is one of the needs of human being. Scientists and engineers have interest in this field so that they become able to reduce or overcome the unexpected losses or delays in their projects. Importance of forecasting in financial sector can't be neglected too. Application areas of modelling and forecasting include stock market forecasts, the weather forecasts, fund management, commodities price prediction etc.

Forecasting as a research topic attracted many researchers over the past few decades. Zadeh [1] proposed the fuzzy set theory first and then got fruitful achievements both in theory and applications. Lee et al. [2] presented method for forecasting the temperature and the TAIFEX based on fuzzy logic relation groups and genetic algorithm. They also used the genetic algorithm and simulated annealing in it. Jilani and Burney [3], [4] and Jilani, Burney and Ardil [5], [6] presented new fuzzy metrics for high-order multivariate fuzzy time series forecasting for car road accident casualties in Belgium. Huang et al. [7] proposed a new forecasting model based on two computational methods, fuzzy time series and particle swarm optimization, is presented for academic enrolments. Kuo et al. [8] presented a new hybrid forecast method to solve the TAIFEX forecasting problem based on fuzzy time series and particle swarm optimization.

Rest of the paper is organized as follows. In Section 2, we have presented a brief overview for fuzzy time series, genetic algorithms and particle swarm optimization along-with the variants. In Section 3, we have presented a hybrid method based on genetic algorithm and particle swarm optimization based on two-factor high-order fuzzy time series and experimental results are given in Section 4. Finally, some concluding remarks are given in Section 5.

2. Related works

2.1. Review of Fuzzy time series

The concept of fuzzy logic and fuzzy set theory was introduced to cope with the ambiguity and uncertainty of most of the real-world problems. Thus a time series introduced with fuzziness is termed as fuzzy time series. Song and Chissom [11], [12] introduced the concept of fuzzy time series and since then a number of variants were published by many authors. Some of the basic concepts in fuzzy time series are fuzzy operations, fuzzy relation and fuzzy logical relationship groups [4], [6].

2.2 Review of Genetic Algorithms and Particle Swarm Optimization Techniques

2.2.1 Genetic Algorithm (GA)

Among existing evolutionary algorithms, the most well-known branch is genetic algorithm (GA). GA is a population-based probabilistic search and optimization technique, which works based on the mechanism of natural genetics and Darwin's principle of natural selection. GAs are stochastic search procedures based on the mechanics of natural selection, genetics, and evolution ([11]).

In GA a population consists of chromosomes composed of genes, where the number of chromosomes in a population is called the population size. Evolution in genetic algorithms is achieved by three fundamental genetic operators: **selection, crossover** and **mutation** ([11]). Based on a ranking scheme that promotes the best individuals of the population, i.e. those with lowest fitness function values (for minimization problem), a number of parents are selected from the current population. The parents are stochastically combined and produce offspring that carry parts of their chromosomes. Then, a stochastic mutation procedure alters one or more digits in each chromosome, and the mutated offspring are evaluated. Finally the best among all individuals (parents and offspring) are selected to comprise the population of the next generation. The algorithm terminated as soon as a solution of desirable quality is found or the maximum available computational budget is exhausted. Figure 1 shows flow chart representing working of genetic algorithm.

2.2.2 Particle swarm optimization (PSO)

Particle swarm optimization is a member of wide category of swarm intelligence methods ([12]). It is a population base stochastic optimization technique developed by Eberhart and Kennedy [13], [14] inspired by social behaviour of bird flocking or fish schooling. Each particle in PSO flies through the search space with an adaptable velocity that is dynamically modified according to its own flying experience and also communicates together utilizing flying experience of the other particles. In the PSO, each particle remember the best position from its own flying experience is called Pbest, and then the overall best out of all the particles in the population is called Gbest. Many real optimization problems can be formulated as the following functional optimization problem:

$$\min f(x), \quad X = [x_1, x_2, \cdots, x_n], \quad (1)$$
$$such\ that \quad x_i \in [a_j, b_j], \quad j = 1,2, \dots \dots \dots n$$

<u>Algorithm [Pseudo code]: Particle Swarm Optimization</u>

Step 1: (Initialization) For each particle j in the population.
Step 2: initialize X_j and V_j randomly.
Step 3: evaluate f_j.
Step 4: initialize G_{best} with the best function value among the population.
Step 5: initialize $P_{best,j}$ with a copy of X_j, $\forall j \leq N$.
Step 6: Repeat until a stopping criterion is satisfied:
Step 7: find G_{best} such that $[G_{best}] \leq f_j$, $\forall j \leq N$.
Step 8: for each particle j, $P_{best.j} = X_j$ if $f_j < f_{best}[j]$, $\forall j \leq N$
Step 9: for each particle j, update V_j and X_j according to equation 2 and 3.
Step 10: evaluate f_j for all particles.

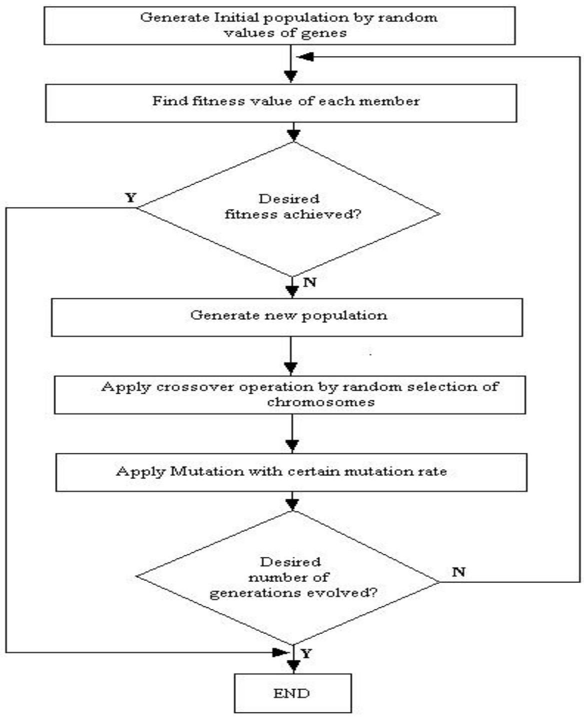

Fig.1 Flow chart of Genetic Algorithm

where f is objective function, and X is the decision vector consisting of n variables. Each individual within the swarm (i.e. population of particles) is represented by a vector in multidimensional search space. This vector has also one assigned vector which determines the next movement of the particle and is called the velocity vector. The PSO algorithm also determines how to update the velocity of a particle. Each particle updates its velocity based on current velocity and the best position it has explored so far; and also based on the global best position explored by swarm Engelbrecht [15], J. Sadri, and Ching Y. Suen [16]. The PSO process then is iterated a fixed number of times or until a minimum error based on desired performance index is achieved.

A particle's status on the search space is characterized by two factors: its position and velocity, which are updated by following equations.

$$V_{i,j}(t+1) = V_{i,j}(t) + c_1 R_1 [\, Pbest_{i,j}(t) - x_{i,j}(t)] + c_2 R_2 [\, Gbest_{i,j}(t) - x_{i,j}(t)] \quad (2)$$

$$x_{i,j}(t+1) = x_{i,j}(t) + V_{i,j}(t+1) \quad (3)$$

where $V_j = [v_{j,1}, v_{j,2}, \cdots v_{j,n}]$ called the velocity for particle j, which represents the distance to be travelled by this particle from its current position; $X_j = [x_{j,1}, x_{j,2}, \cdots, x_{j,n}]$ represents the position of particle j; Pbest represents best previous position of the particle j (i.e. local best position or its experience); Gbest represents the best position among all particles in the population $X = [X_1, X_2, \cdots, X_n]$ (i.e. global best position); R_1 and R_2 are two independently uniformly distributed random variables with range [0,1]; c_1 and c_2 are positive constant parameters called acceleration coefficients which control the maximum step size. Generally the value of each component in V_j can be clamped to the range $[-v_{max}, v_{max}]$ to control excessive roaming of particles outside the search space. Then the particle flies toward a new position according to eq. (3). This process is repeated until a user-defined stopping criterion is reached (i.e. positions of all particles converge to the same set of values.

To enhance the performance of standard PSO, many variants has been proposed and implemented by many researchers in many different research problems. One of the variant of SPSO with improved formula for velocity computation is

$$V_{i,j}(t+1) = wV_{i,j}(t) + c_1 R_1 [\, Pbest_{i,j}(t) - x_{i,j}(t)] + c_2 R_2 [\, Gbest_{i,j}(t) - x_{i,j}(t)] \quad (4)$$

An inertia term, w, is added to reduce the exponential or free fall of the velocities in successive iterations. A larger value of w promotes global exploration and a smaller value promotes a local search. To achieve a balance between global and local exploration, an inertia weight whose value decreases linearly with the iteration number has been used:

$$w(t) = w_{up} - \left(\frac{w_{up} - w_{low}}{T_{max}}\right) * t \quad (5)$$

where w_{up} and w_{low} are the initial and final values of the inertia weight, respectively, and T_{max} is the maximum number of iterations used in PSO. The values of w_{up}1.4 and w_{low}= 0.4 are commonly used.

3. Hybrid GA and PSO based two-factor high-order fuzzy time series forecasting model

Based on the two-factors high-order fuzzy time series, the proposed forecasting algorithm can be described as follows:

Step 1: Define the universe of discourse of the main-factor (X) U = [D_{min}-D_1, D_{max}+D_2], where D_{min} and D_{max} are the minimum and maximum values from the known historical data, respectively, and D_1 and D_2 are two proper positive real numbers. Similarly, define the universe of discourse V of the second-factor(Y) V=[E_{min}-E_1, E_{max}+E_2], where E_{min} and E_{max} are the minimum and maximum values from the known historical data, respectively, and E_1 and E_2 are two selected positive numbers. In the proposed algorithm, we partition the universe of discourse U of the main-factor into sixteen intervals and V of the second-factor into eight intervals. So, each chromosome consists of fifteen X genes and seven Y genes. We have set a population size of 50 chromosomes and the system randomly generates 50 chromosomes as the initial population, as shown in Table 1.

TABLE 1
INITIAL POPULATION OF CHROMOSOMES

C. No	X1	X2	.	.	.	X15	Y1	Y2	.	.	Y8
1	1079	1118	.	.	.	1644	306	348	.	.	778
2	959	982	.	.	.	1551	92	203	.	.	737
3	1083	1150	.	.	.	1576	182	193	.	.	773
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
49	1000	1132	.	.	.	1588	94	322	.	.	660
50	1018	1080	.	.	.	1637	301	362	.	.	807

Step 2: Define the linguistic term Ai, for i=1, 2,... n, which represented by fuzzy sets of the main-factor.

$$A_1 = {}^{1}/_{u_1} + {}^{0.5}/_{u_2} + \cdots + {}^{0}/_{u_{n-1}} + {}^{0}/_{u_n}$$
$$A_2 = {}^{0.5}/_{u_1} + {}^{1}/_{u_2} + {}^{0.5}/_{u_3} + \cdots + {}^{0}/_{u_n}$$
$$\vdots \quad (6)$$

$$A_n = {}^0/_{u_1} + \cdots + {}^0/_{u_{n-2}} + {}^{0.5}/_{u_{n-1}} + {}^1/_{u_n}$$

where $A_1, A_2, \ldots A_n$ are linguistic terms to describe the values of the main-factor which are divided into equal length intervals $u_1, u_2, \ldots, u_n$. And also, define the linguistic term B_j, for j = 1, 2,…, m, which is represented by fuzzy sets of the second-factor as follows:

$$\begin{aligned} B_1 &= {}^1/_{v_1} + {}^{0.5}/_{v_2} + \cdots + {}^0/_{v_{n-1}} + {}^0/_{v_n} \\ B_2 &= {}^{0.5}/_{v_1} + {}^1/_{v_2} + {}^{0.5}/_{v_3} + \cdots + {}^0/_{v_n} \\ &\vdots \\ B_n &= {}^0/_{v_1} + \cdots + {}^0/_{v_{n-2}} + {}^{0.5}/_{v_{n-1}} + {}^1/_{v_n} \end{aligned} \tag{7}$$

where $B_1, B_2, \ldots B_m$ are linguistic terms to describe the values of the second-factor which are divided into equal length intervals $v_1, v_2, \ldots, v_n$.

Step 3: For fuzzification of historical data find out the interval u_i, where $1 \leq i \leq n$, to which the value of the main-factor belongs to u_i.

Case (1) If the value of the main-factor belongs to u_1, then the value of the main-factor is fuzzified into $1/A_1$+$0.5/A_2$, denoted by X1.

Case (2) If the value of the main-factor belongs to u_i, where $2 \leq i \leq n-1$, then the value of the main-factor is fuzzified into $0.5/A_{i-1}$+$1/A_i$ + $0.5/A_{i+1}$ denoted by Xi.

Case (3) If the value of the main-factor belongs to u_n, then the value of the main-factor is fuzzified into $0.5/A_{n-1}$+$1/A_n$, denoted by Xn.

Find out the interval v_j, where $1 \leq j \leq m$, to which the value of the second-factor belongs to v_j.

Case (1) If the value of the second-factor belongs to v_1, then the value of the second-factor is fuzzified into $1/B_1$+$0.5/B_2$, denoted by Y1.

Case (2) If the value of the second-factor belongs to v_j, where $2 \leq j \leq m-1$, then the value of the second-factor is fuzzified into $0.5/B_{i-1}$+$1/B_i$ + $0.5/B_{i+1}$, denoted by Yj.

Case (3) If the value of the second-factor belongs to v_m, then the value of the second-factor is fuzzified into $0.5/B_{n-1}$+$1/B_n$, denoted by Ym.

Step 4: Construct the two-factors nth-order fuzzy logical relationships based on the main and secondary factors from the fuzzified historical data obtained in Step 3. If the fuzzified historical data of the main-factor of day i is X_i, then construct the two-factors kth-order fuzzy logical relationships $((X_{ik}, Y_{ik}), \ldots, (X_{i2}, Y_{i2}), (X_{i1}, Y_{i1})) \rightarrow X_i$ from day $i-k$ to day i, where $2 \leq k \leq n$ and $X_{ik}, \cdots\cdots X_{i2}, X_{i1}$ denote the fuzzified values of the main-factor of days i-k,…, i-2, i-1 respectively; $Y_{ik}, \cdots\cdots Y_{i2}, Y_{i1}$ denote the fuzzified value of the second-factor of days i-k,…, i-2, i-1, respectively. Then, divide the derived fuzzy logical relationships into fuzzy logical relationship groups based on the current states of the fuzzy logical relationships ([3]).

Primary Factor	Sec. Factor	Fuzzy Logic Relation Groups (FLRs)
A_{15}	B8	--
A_{15}	B8	--
A_{15}	B8	--
A_{15}	B8	Group 1: ((A15, B8), (A15, B8), (A15, B8)) → A15
.	.	.
.	.	.
A6	B5	Group 44: ((A8, B6), (A7, B6), (A6, B6)) → A6

Table 2: Two –Factor Third Order Fuzzy Logical Relationship Groups

Step 5: For two-factor k-order fuzzy logical relationship in Table 2, the forecasted value is calculated using:

$$t_j = \frac{\sum_{i=j-k}^{j+k} w_i}{\sum_{i=j-k}^{j+k} \left(\frac{w_i}{a_i}\right)} \tag{8}$$

Based on above defuzzification formula, we define a two-factor third-order defuzzifier as ([3])

$$t_j = \begin{cases} \dfrac{1+0.5}{\frac{1}{a_1}+\frac{0.5}{a_2}} & \text{if } j = 1, \\ \dfrac{0.5+1+0.5}{\frac{0.5}{a_{j-1}}+\frac{1}{a_j}+\dfrac{0.5}{a_{j-1}}} & \text{if } 2 \leq j \leq n-2 \\ \dfrac{0.5+1}{\frac{0.5}{a_{n-1}}+\frac{1}{a_n}} & \text{if } j = n \end{cases} \tag{9}$$

Here a_{j-1}, a_j *and* a_{j+1} are the midpoints of the intervals u_{j-1}, u_j *and* u_{j+1} respectively.

Step 6: Calculate the fitness value of each chromosome in the population. In the proposed method we used average forecasting error rate (AFER) and mean square error (MSE) as fitness values of each chromosome for TAIFEX and KSE index forecasting.

Step 7: Select the top 10 chromosomes from the population whose fitness values are smaller than the other chromosomes. Use the best chromosome from these 10 at eleventh position. To form new population consisting of 50 chromosomes, generate another 39 chromosomes randomly generated by the system and apply crossover operation on chromosome at eleventh position with any other randomly selected chromosome by system.

Step 8: Now initialize particles for PSO from the first 15 chromosomes of newly generated population

in Step 7. We have used PSO to tune up particles after regenerating population.
Step 9: Initialize weight, C1 and C2 parameters and find forecast using our 15 particles one by one. Based on the standard PSO algorithm-1, calculate the pbest and gbest values based on objective values for each particle.
Step 10: For each particle, calculate particle velocity according to eq. (4), which is based on the gbest and the pbest values and then update particle positions according to eq. (3). This step will lead the each particle to a more promising solution for optimal tuning of two-factor nth-order fuzzy time series. Update the value of the weight factor w according to eq. (5).
Step 11: If the PSO iterations have executed for a predefined number of times then stop, otherwise repeat from step 9. We have run PSO for 100 iterations in proposed method. After stopping iterations of PSO, set back the particles into starting 15 positions of our chromosomes for GA. In this way we got our chromosomes tuned up through PSO.
Step 12: Randomly select two chromosomes from the population to perform the crossover operation. When performing the crossover operation, the system randomly selects one crossover point both from X and Y genes, where the crossover point of X genes is an integer between 1 and $n-1$, and the crossover point of Y genes is an integer between 1 and $m-1$, n and m are the number of X and Y genes respectively. Sort the genes in chromosomes.
Step 13: Select a chromosome randomly from population for mutation operation. Find out a mutation point both for X and Y genes randomly. Assume that the system randomly selects gene x_3 and gene y_4 of a chromosome to perform the mutations, then the gene x_3 will be replaced by a value between x_2 and x_4, and gene y_4 will be replaced by a value between y_3 and y_5. In this paper, we have used the mutation rate of 0.5.
The overall computational flow of the proposed PSO based two-factors high-order fuzzy time series forecasting is described in Fig. 2.

4. Experimental results

We have used two stock market datasets. In the first experiment, we have applied the proposed method to the TAIFEX as shown in Table 3. For TAIFEX forecasting the MSE is 163.597. We have compared results of proposed method with many other existing method's results in Table 4. Then we applied same method for KSE-100 index forecasting and obtained MSE value 7822.805.

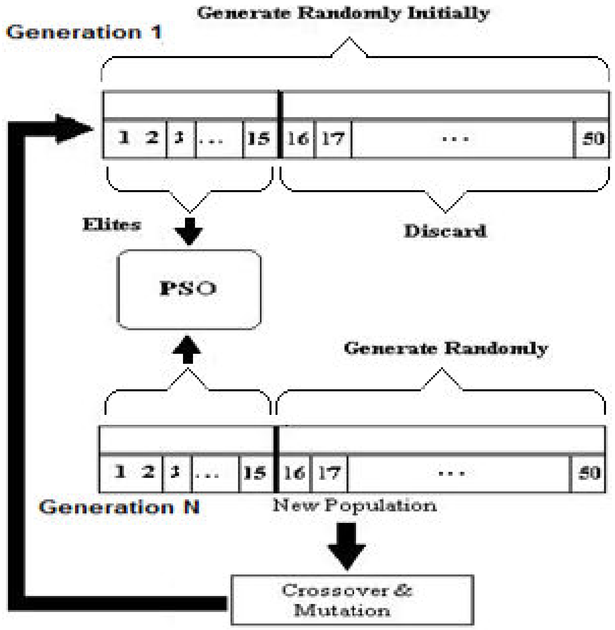

Fig. 2. PSO based two-factor high-order fuzzy time series forecasting algorithm.

Methods	MSE
Chen's method [17]	9668.94
Quantile based method (Jilani and Burney) [6]	3736.64
Lee, Wang and Chen (2007) [2]	249.61
Wang and Chen [18]	252.47
Proposed Method	**163.597**

TABLE 4: A Comparison of MSE of different methods with proposed method for TAIFEX forecasting

5. Conclusion and Future studies

During the last few years, stock market shows very volatile time series in nature and it has difficult to make the potential relationship as a mathematical model. So, fuzzy time series has shown good performances for these real world problems. Our proposed forecasting method for TAIFEX and KSE-100 has shown better forecasting accuracy than many previous researches.
In future work, we aim to develop hybrid multivariate fuzzy time series forecasting techniques by using evolutionary and nature inspired computing methods.

ISBN: 978-1-61804-089-3

Date	Actual TAIFEX index (primary factor)	Fuzzified TAIFEX	Actual TAIEX index (secondary factor)	Fuzzified TAIEX	Forecasted TAIFEX Index
8/3/1998	7552	A_{15}	7599	B_8	--
8/4/1998	7560	A_{15}	7593	B_8	--
8/5/1998	7487	A_{15}	7500	B_8	--
8/6/1998	7462	A_{15}	7472	B_8	7461.917
8/7/1998	7515	A_{15}	7530	B_8	7541.465
8/10/1998	7365	A_{14}	7372	B_7	7372.554
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
9/28/1998	6840	A_7	6911	B_6	6844.361
9/29/1998	6806	A_6	6885	B_6	6786.851
9/30/1998	6787	A_6	6834	B_5	6786.851
AFER					0.1488%
MSE					163.597

Table 3: Actual and Forecasted TAIFEX Index

References:

[1] L.A. Zadeh, Fuzzy sets, Information and Control 8 (1965), pp. 338–353.

[2] Lee L.-W., Wang L.-H., Chen S.-M., 2007. Temperature prediction and TAIFEX forecasting based on fuzzy logical relationships and genetic algorithms, Expert Systems with Applications 33, 2007, pp. 539- 550.

[3] T.A. Jilani, S.M.A. Burney, Multivariate stochastic fuzzy forecasting models, Expert Systems with Applications 35 (2008a), pp. 691–700.

[4] T.A. Jilani, S.M.A. Burney, A refined fuzzy time series model for stock market forecasting, Physica-A 387 (2008b), pp. 2857-2862.

[5] T.A. Jilani, S.M.A. Burney, C. Ardil, Multivariate High Order Fuzzy Time Series Forecasting for Car Road Accidents, International Journal of Computational Intelligence 4 (1), (2008c), pp. 15-20.

[6] T.A. Jilani, S.M.A. Burney, C. Ardil, A New Quantile Based Fuzzy Time Series Forecasting Model, International Journal of Intelligent Systems and Technologies 3 (4), (2008d), pp. 201-207.

[7] Y.-L. Huang, S.-J. Horng, M. He, P. Fan, T.-W. Kao, M. K. Khan, J.-L. Lai, I-H. Kuo, A hybrid forecasting model for enrollments based on aggregated fuzzy time series and particle swarm optimization, Expert Systems with Applications 7(38), (2011), pp. 8014–8023.

[8] I-H. Kuo, S.-J. Horng, Y.-H. Chen, R.-S. Run, T.-W. Kao, R.-J. C., J.-L. Lai, T.-L. Lin, Forecasting TAIFEX based on fuzzy time series and particle swarm optimization, Expert Systems with Applications 2(37), (2010), pp. 1494–1502

[9] Q. Song and B.S. Chissom, Fuzzy time series and its models, Fuzzy Sets and Systems 54 (3) (1993a), pp. 269–277.

[10] Q. Song and B.S. Chissom, Forecasting enrollments with fuzzy time series – Part II, Fuzzy Sets and Systems 62 (1) (1994a), pp. 1–8.

[11] Goldberg D.E. Genetic algorithm in search, optimization, and machine learning, Addison-Wesley, Massachusetts, 1989.

[12] M. Clerc, Particle Swarm Optimization, ISTE Ltd, USA, 2006.

[13] J. Kennedy, and R Eberhart, Particle Swarm Optimization, IEEE Conference on Neural Networks, (Perth, Australia), Piscataway, NJ, IV, 1995, pp. 1942-1948.

[14] J. Kennedy and R. Eberhart. Swarm Intelligence. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 2001.

[15] A. P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, Wiley, 2005.

[16] J. Sadri, and Ching Y. Suen, A Genetic Binary Particle Swarm Optimization Model, IEEE Congress on Evolutionary Computation, Vancouver, BC, Canada, 2006.

[17] .M. Chen, Forecasting enrollments based on fuzzy time series, Fuzzy Sets and Systems 81 (3) (1996), pp. 311–319.
[18] Wang N.-Y., Chen S.-M., 2009. Temperature prediction and TAIFEX forecasting based on automatic clustering techniques and two-factors high-order fuzzy time series , Expert Systems with Applications, 36(2), pp.2143-2154.