2011 Eighth International Conference on Information Technology: New Generations

Predicting the Reliability of Software Systems Using Fuzzy Logic

Sultan Aljahdali
Department of Computer Science
Taif University
Taif, Saudi Arabia
Aljahdali@tu.edu.sa

Alaa F. Sheta
Computers and Systems Department
Electronics Research Institute (ERI)
Cairo, Egypt
asheta66@gmail.com

***Abstract*—Software industry suffer many challenges in developing a high quality reliable software. Many factors affect their development such as the schedule, limited resources, uncertainty in the developing environment and inaccurate requirement specification. Software Reliability Growth Models (SRGM) were significantly used to help in solving these problems by accurately predicting the number of faults in the software during both development and testing processes. The issue of building growth models was the subject of many research work. In this paper, we explore the use of fuzzy logic to build a SRGM. The proposed fuzzy model consists of a collection of linear sub-models joined together smoothly using fuzzy membership functions to represent the fuzzy model. Results and analysis based data set developed by John Musa of Bell Telephone Laboratories [1] are provided to show the potential advantages of using fuzzy logic in solving this problem.**

I. Introduction

Machine Learning (ML) and Soft Computing techniques, such as Genetic Algorithms (GAs), Genetic Programming (GP), Evolutionary strategies (ESs), Artificial Neural Networks (ANN), Fuzzy Logic (FL), and Particle Swarm Optimization (POS), to solve software engineering problems expanded in the recent years.

Estimation of the COCOMO model parameters using Genetic Algorithms (GAs) for NASA Software Projects were provided in [2]. Parameter Estimation of Hyper-Geometric Distribution Software Reliability Growth Model by Genetic Algorithms was presented in [3]. Predicting accumulated faults during the software testing process using parametric and non-parametric models were explored in many articles [4]–[6]. In [7], author provided a strategic solution for estimating software defect fix effort using self-organizing neural network. Genetic programming (GP) was successfully used to find a model that fits the given data points without making any assumptions about the model structure [8], [9]. GP found to be a powerful technique in developing software reliability growth modeling.

In this paper, we explore the use of fuzzy logic to predict faults during the software testing process using software historical faults data. In section II, we provide an overview of various SRGM. An introduction on fuzzy modeling technique is presented in section III. The proposed fuzzy model structure is presented in section IV. Detailed information about the data set and the experiments developed are provided in sections V, VII.

II. Software Reliability Growth Models

In the past three decades, hundreds of models were introduced to estimate the reliability of software systems [10]–[12]. The issue of building growth models was the subject of many research work [13], [14] which helps in estimating the reliability of a software system before its release to the market. Serious application such as weapon systems and NASA space shuttle applications were explored [15]–[17].

Faults may be encounter in market released software. This is a challenge for software companies. It might affect their reputation and finance. Software reliability growth models were significantly used to help in computing the number of faults which is still resides in the software [18]. Thus, specifying the effort required to fix faults, the time required before software can be released and the cost of repair. Software reliability growth models employ system experimental data for testing to predict the number of defects remaining in the software.

Software reliability models can be classified to two types of models according to prediction style either from:

- the design parameters thus called ”defect density” models
- the test data thus ”software reliability growth” models.

Some known SRGM are Logarithmic, Exponential, Power, S-Shaped and Inverse Polynomial models [19], [20]. They are typical analytical models. They normally describe the fault process as a function of execution time (or calendar time) and a set of unknown parameters. The model parameters normally estimated using least-square estimation or maximum likelihood techniques [13].

III. Fuzzy Modeling

Fuzzy logic have been successfully used to solve variety of problems in system identification, signal processing and control [21]–[24]. Fuzzy modeling has been regarded as one of the key problems in fuzzy systems research [25], [26]. In the past years, research focused on the development of fuzzy systems from both theoretical and applications oriented prospective were presented in [27]–[29].

A fuzzy model structure can be represented by a set of fuzzy If-Then rules [30]. A fuzzy rule has two parts the *antecedent* and the *consequence*. The antecedent variables reflect information about the process operating conditions.

978-0-7695-4367-3/11 $26.00 © 2011 IEEE
DOI 10.1109/ITNG.2011.14

The consequent of the rule is usually a linear regression model which is valid around the given operating condition [31]–[35].

Consider a dynamical system with a set on inputs x_1, x_2, $\ldots$, x_n variables and y single output variable. The relationship between these variables can be represented as:

$$y = f(x_1, x_2, \ldots, x_n) \tag{1}$$

Our objective is to develop a fuzzy model structure which describe the function f between the inputs $x_i, i = 1, \ldots, n$ and the output y. Knowing that the inputs and outputs are measured at the sample time t; $t = 1, 2, \ldots, N$. A fuzzy model of a dynamic system for a multi-input single-output (MISO) system has a set of rules of the following format:

$$R_k : \quad \textbf{If } x_1 \text{ is } A_{k1} \textbf{ and } \ldots \textbf{ and } x_n \text{ is } A_{kn} \textbf{ then}$$
$$y_k = a_{k1}x_1 + a_{k2}x_2 + \cdots + a_{kn}x_n + a_{k0} \tag{2}$$

where k is the rule number. For example, if we want to model a three input $n = 3$ single output system using two rules $i = 1, 2$. The system can be represented as follows:

$$R_1 : \quad \textbf{If } x_1 \text{ is } A_{11} \textbf{ and } x_2 \text{ is } A_{12} \textbf{ and } x_3 \text{ is } A_{13} \textbf{ then}$$
$$y_1 = a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{10}$$
$$R_2 : \quad \textbf{If } x_1 \text{ is } A_{21} \textbf{ and } x_2 \text{ is } A_{22} \textbf{ and } x_3 \text{ is } A_{23} \textbf{ then}$$
$$y_2 = a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{20}$$

where y_1 is the output corresponding to the region of values, in the input domain, of the variables x_1, x_2, x_3 based a set of membership function A_{11}, A_{12} and A_{13}.

The inner-product space for the fuzzy set A_i, which defines the fuzzy region in the antecedent space, can be described as:

$$A_i = \prod_{j=1}^{n} A_{ij} = A_{i1} \times A_{i2} \times \ldots A_{in} \tag{3}$$

Thus, the degree of fulfillment is given by:

$$\beta_i = \mu_{A_{i1}(x_1)} \wedge \mu_{A_{i2}(x_2)} \wedge \ldots \mu_{A_{i1}(x_n)} \tag{4}$$

Other operator can be used to develop β_i. The antecedent part of the rule divide the input domain of x into a set of fuzzy sub-domains. Each domain is corresponding to a fuzzy set. The number of rules M can be computed as:

$$M = \prod_{i=1}^{n} T_i \tag{5}$$

where n is the dimension of the input space (i.e. the number of inputs) as described earlier and T_i is the number of terms in the ith antecedent variables.

IV. Proposed Fuzzy Model Structure

Our objective is to approximation the dynamics of the fault measurements during the testing process and instead of representing it in a single nonlinear model we can extend it by a set of local linear models. Each local model should be able to represents the relationship between the historical faults $y(k-1), y(k-2), y(k-3), y(k-4)$ and the current fault $y(k)$ in a certain range of operating conditions. Such a proposed fuzzy model structure can be successfully represented by means of fuzzy If-Then rules. The proposed model equation is given as follows:

$$y(k) = FM(y(k-1), y(k-2), y(k-3), y(k-4)) \tag{6}$$

Using membership functions and the antecedent of the rule we can define the fuzzy region in the product space. The *antecedent* variables gives the condition of the process status now. The consequent of the rule is typically a local linear regression model which relates $y(k)$ with $y(k-1), y(k-2), y(k-3), y(k-4)$.

A rule-based fuzzy model requires the identification of the following: 1) the $antecedent$ 2) the $consequent$ structure, 3) the type of the membership functions for different operating regions and 4) the estimation of the consequent parameters. The developed fuzzy models implemented based the Takagi-Sugeno technique [31], [32]. The proposed technique does not require any *a prior* knowledge about the operating regimes. If a sufficiently number of measurements are collected which reflects the operating ranges of interest, the developed fuzzy model will be an efficient one.

V. The Software Reliability Data

John Musa of Bell Telephone Laboratories compiled a software reliability database [1]. His objective was to collect fault interval data to assist software managers in monitoring test status, predicting schedules and to assist software researchers in validating software reliability models. These models are applied in the discipline of software reliability engineering. The dataset consists of software fault data on 16 projects. Careful controls were employed during data collection to ensure that the data would be of high quality. The data was collected throughout the mid 1970s. It represents projects from a variety of applications including real time command and control, word processing, commercial, and military applications.

In our case, we used data from three different projects. They are Real Time Control, Military and Operating System. A MATLAB toolbox for modeling of fuzzy systems [36] was used to implement the following results. The routines of the toolbox contain the Gustanfson-Kessel (GK) clustering algorithm, whose implementation is given in [37].

VI. Validation Criteria

In order to check the performance of the developed model, we compute the Variance-Accounted-For (VAF) performance criterion to measure how close the measured values to the values developed using the fuzzy models. The VAF is computed as:

$$VAF = \left[1 - \frac{var(y - \hat{y})}{var(y)}\right] \times 100\% \tag{7}$$

where $y, \hat{y}$ are the real actual output and the fuzzy model estimated output, respectively.

VII. Experimental Results

We run the Fuzzy Model Identification Toolbox [36] along with three membership functions. The data set was split into two parts: 1) 70% of the collected data for training and 2 30% for testing (i.e. validation). The set of rules which describe the three software projects (i.e. Real Time Control, Military and Operating Systems [1]) are presented in Tables I, II and III.

In Figure 1, we show the membership function for the real time control application. We used three clusters to build the fuzzy model. Figure 2 show the actual and predicted faults over the training and testing data for the real time control applications. The fuzzy membership functions for the military application and operating systems are shown in Figures 3 and 5, respectively. Figure 4 and 6 show the actual and predicted faults over the training and testing data for the military and operating systems applications.

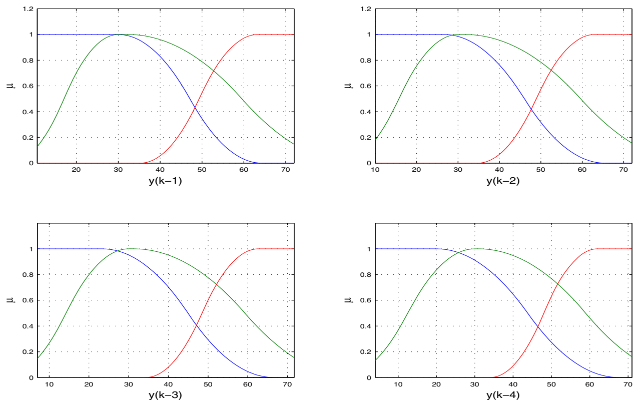

Fig. 1. Membership functions for the Real-Time Control Applications

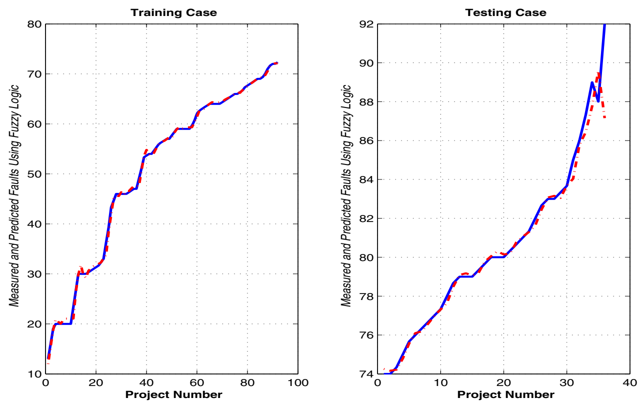

Fig. 2. Actual and estimated responses in Real-Time Control Applications

The developed model's performance were computed using the VAF criteria and reported in Table IV. It can be seen that the performance of the developed fuzzy model based historical data were achieving significant modeling capabilities.

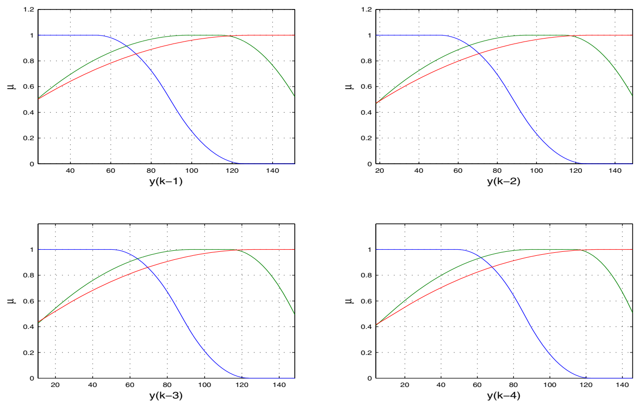

Fig. 3. Membership functions for the Military Applications

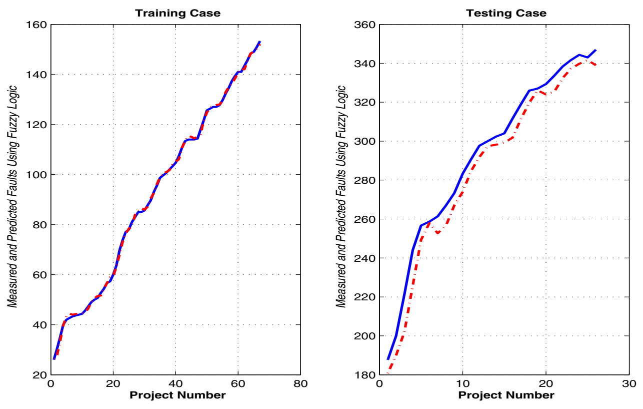

Fig. 4. Actual and estimated responses in Military Applications

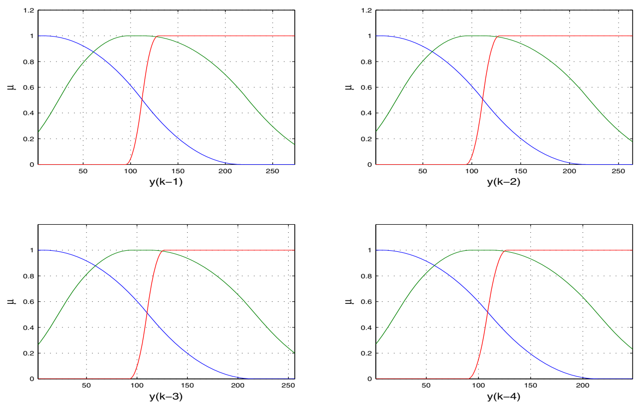

Fig. 5. Membership functions for the Operating System Applications

TABLE IV
VAF FOR THE FUZZY MODELS

Project Name	Training VAF	Testing VAF
Military	99.9383	99.0606
Real Time Control	99.8693	95.2851
Operating System	99.9890	99.9044

TABLE I
FUZZY RULES FOR REAL TIME AND CONTROL APPLICATIONS

1. **If** $y(k-1)$ is A_{11} **and** $y(k-2)$ is A_{12} **and** $y(k-3)$ is A_{13} **and** $y(k-4)$ is A_{14} **then**
$\hat{y}(k) = 1.56 \cdot 10^0 y(k-1) - 8.89 \cdot 10^{-1} y(k-2) - 6.43 \cdot 10^{-2} y(k-3) + 2.10 \cdot 10^{-1} y(k-4) + 3.82 \cdot 10^0$
2. **If** $y(k-1)$ is A_{21} **and** $y(k-2)$ is A_{22} **and** $y(k-3)$ is A_{23} **and** $y(k-4)$ is A_{24} **then**
$\hat{y}(k) = 1.56 \cdot 10^0 y(k-1) - 8.19 \cdot 10^{-2} y(k-2) - 1.16 \cdot 10^0 y(k-3) + 5.87 \cdot 10^{-1} y(k-4) + 5.38 \cdot 10^0$
3. **If** $y(k-1)$ is A_{31} **and** $y(k-2)$ is A_{32} **and** $y(k-3)$ is A_{33} **and** $y(k-4)$ is A_{34} **then**
$\hat{y}(k) = 1.49 \cdot 10^0 y(k-1) - 6.63 \cdot 10^{-1} y(k-2) - 6.74 \cdot 10^{-2} y(k-3) + 2.35 \cdot 10^{-1} y(k-4) + 6.12 \cdot 10^{-1}$

TABLE II
FUZZY RULES FOR MILITARY APPLICATIONS

1. **If** $y(k-1)$ is A_{11} **and** $y(k-2)$ is A_{12} **and** $y(k-3)$ is A_{13} **and** $y(k-4)$ is A_{14} **then**
$\hat{y}(k) = 1.53 \cdot 10^0 y(k-1) - 6.76 \cdot 10^{-1} y(k-2) + 6.87 \cdot 10^{-1} y(k-3) - 4.92 \cdot 10^{-1} y(k-4) - 1.96 \cdot 10^0$
2. **If** $y(k-1)$ is A_{21} **and** $y(k-2)$ is A_{22} **and** $y(k-3)$ is A_{23} **and** $y(k-4)$ is A_{24} **then**
$\hat{y}(k) = 2.36 \cdot 10^0 y(k-1) - 5.90 \cdot 10^{-1} y(k-2) - 1.84 \cdot 10^0 y(k-3) + 1.07 \cdot 10^0 y(k-4) - 1.27 \cdot 10^0$
3. **If** $y(k-1)$ is A_{31} **and** $y(k-2)$ is A_{32} **and** $y(k-3)$ is A_{33} **and** $y(k-4)$ is A_{34} **then**
$\hat{y}(k) = 4.10 \cdot 10^{-1} y(k-1) + 7.93 \cdot 10^{-2} y(k-2) + 1.38 \cdot 10^{-1} y(k-3) + 3.47 \cdot 10^{-1} y(k-4) + 8.41 \cdot 10^0$

TABLE III
FUZZY RULES FOR OPERATING SYSTEM APPLICATIONS

1. **If** $y(k-1)$ is A_{11} **and** $y(k-2)$ is A_{12} **and** $y(k-3)$ is A_{13} **and** $y(k-4)$ is A_{14} **then**
$\hat{y}(k) = 1.78 \cdot 10^0 y(k-1) - 4.96 \cdot 10^{-1} y(k-2) - 6.19 \cdot 10^{-1} y(k-3) + 3.22 \cdot 10^{-1} y(k-4) + 2.83 \cdot 10^{-1}$
2. **If** $y(k-1)$ is A_{21} **and** $y(k-2)$ is A_{22} **and** $y(k-3)$ is A_{23} **and** $y(k-4)$ is A_{24} **then**
$\hat{y}(k) = 1.53 \cdot 10^0 y(k-1) - 7.57 \cdot 10^{-1} y(k-2) + 9.33 \cdot 10^{-2} y(k-3) + 1.23 \cdot 10^{-1} y(k-4) + 2.52 \cdot 10^0$
3. **If** $y(k-1)$ is A_{31} **and** $y(k-2)$ is A_{32} **and** $y(k-3)$ is A_{33} **and** $y(k-4)$ is A_{34} **then**
$\hat{y}(k) = 1.93 \cdot 10^0 y(k-1) - 1.11 \cdot 10^0 y(k-2) - 8.67 \cdot 10^{-3} y(k-3) + 2.13 \cdot 10^{-1} y(k-4) - 4.25 \cdot 10^0$

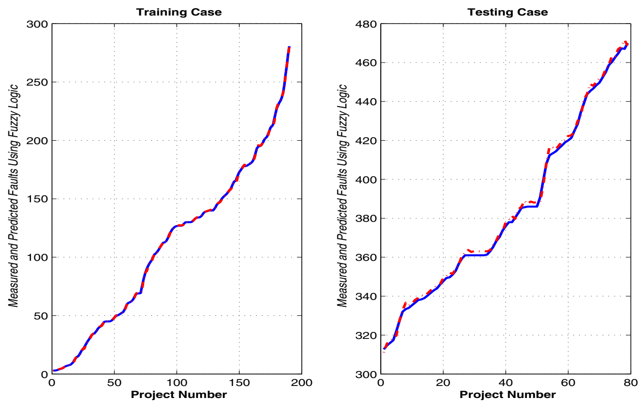

Fig. 6. Actual and estimated responses in Operating System Applications

VIII. CONCLUSIONS AND FUTURE WORK

In this paper we developed a set of fuzzy models for predicting the reliability of software projects in various applications. A fuzzy nonlinear regression models were developed for predicting the accumulated faults of software engineering applications. The developed fuzzy models implemented based the Takagi-Sugeno technique. The developed fuzzy models were tested using three types of applications. They are real-time control, military and operating systems applications. The data set was developed by John Musa of Bell Telephone Laboratories [1]. The results were very promising.

ACKNOWLEDGMENTS

Authors would like to thank Prof. Babuška for providing the FMID Matlab Toolbox.

REFERENCES

[1] J. Musa, “Data analysis center for software: An information analysis center,” *Western Michigan University Library, Kalamazoo, Michigan*, 1980.
[2] A. Sheta, “Estimation of the COCOMO model parameters using genetic algorithms for NASA software projects,” *Journal of Computer Science, USA*, vol. 2, no. 2, pp. 118–123, 2006.

[3] T. Minohara and Y. Tohma, “Parameter estimation of hyper-geometric distribution software reliability growth model by genetic algorithms,” in *Proceedings of the 6th International Symposium on Software Reliability Engineering*, pp. 324–329, 1995.

[4] S. Aljahdali, D. Rine, and A. Sheta, “Prediction of software reliability: A comparison between regression and neural network non-parametric models,” in *ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2001), Beirut, Lebanon*, pp. 470–473, 2001.

[5] S. Aljahdali, A. Sheta, and D. Rine, “Predicting accumulated faults in software testing process using radial basis function network models,” in *17th International Conference on Computers and Their Applications (CATA), Special Session on Intelligent Software Reliability, San Francisco, California, USA*, 2002.

[6] A. Sheta, “Reliability growth modeling for software fault detection using particle swarm optimization,” in *2006 IEEE Congress on Evolutionary Computation, Sheraton, Vancouver Wall Centre, Vancouver, BC, Canada, July 16-21, 2006.*, pp. 10428–10435, 2006.

[7] H. Zeng and D. Rine, “A neural network approach for software defects fix effort estimation,” in *Proceedings of the Eighth IASTED International Conference Software Engineering and Applications*, pp. 513–517, 2004.

[8] W. Afzal and R. Torkar, “Suitability of genetic programming for software reliability growth modeling,” in *Proceedings of the International Symposium on Computer Science and its Applications*, pp. 114–117, 2008.

[9] S. Paramasivam and K. M., “Evaluation of GP model for software reliability,” *Signal Processing Systems, International Conference on*, vol. 0, pp. 758–761, 2009.

[10] J. Musa, “A theory of software reliability and its application,” *IEEE Trans. Software Engineering*, vol. 1, pp. 312–327, 1975.

[11] M. Xie, “Software reliability models - past, present and future,” *In N. Limnios and M. Nikulin (Eds). Recent Advances in Reliability Theory: Methodology, Practice and Inference*, pp. 323–340, 2002.

[12] S. Yamada, “Software reliability models and their applications: A survey,” in *International Seminar on Software Reliability of Man-Machine Systems - Theories Methods and Information Systems Applications - August 17-18, Kyoto University, Kyoto, Japan*, 2000.

[13] M. R. Lyu, *Handbook of Software Reliability Engineering*. IEEE Computer Society Press, McGraw Hill, 1996.

[14] J. Musa, *Software Reliability Engineering: More Reliable Software, Faster and Cheaper*. ISBN 0079132715: Published AuthorHouse, 2004.

[15] P. Carnes, “Software reliability in weapon systems,” in *Eighth International Symposium on Software Reliability Engineering (ISSRE ’97)*, 1997.

[16] T. Keller and N. Schneidewind, “Successful application of software reliability engineering for the NASA space shuttle,” in *Eighth International Symposium on Software Reliability Engineering (ISSRE ’97)*, 1997.

[17] N. F. Schneidewind and T. W. Keller, “Applying reliability models to the space shuttle,” *IEEE Transactions on Software Engineering*, pp. 28–33, 1992.

[18] Y. Tohman, K. Tokunaga, S. Nagase, and M. Y., “Structural approach to the estimation of the number of residual software faults based on the hyper-geometric distribution model,” *IEEE Trans. on Software Engineering*, pp. 345–355, 1989.

[19] W. Farr, “Software reliability modeling survey,” pp. 71–117, 1996.

[20] E. Jones, L. Jones, and A. J. Rembert, “A simulation based trainer for software reliability modeling,” *International Symposium on Software Reliability Engineering*, p. 160, 2001.

[21] D. A. White and D. A. Sofge, *Handbook of Intelligent Control, Neural, Fuzzy, and Adaptive Approaches*. Van Nostrand Reinhold, New York, 1992.

[22] L. X. Wang and J. M. Mendel, “Fuzzy adaptive filter: with application to nonlinear channel equalization,” in *Proc. of 2nd Int. Conf. Fuzzy Syst.*, pp. 895–900, 1993.

[23] L. X. Wang and J. M. Mendel, “Fuzzy adaptive filter: with application to nonlinear channel equalization,” *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 3, pp. 161–170, 1993.

[24] M. Brown and C. J. Harris, *Neurofuzzy Adaptive Modeling and Control*. Hemel Hempstead: Prentice Hall, 1994.

[25] L. X. Wang, “Fuzzy systems are universal approximators,” in *Proceedings of IEE Int. Conf. on Fuzzy Systems, San Diego, USA*, pp. 1163–1170, 1992.

[26] D. Dubois and H. Prade, “Fuzzy sets in approximate reasoning: part 1,” *Fuzzy Sets and Systems*, vol. 40, pp. 143–202, 1992.

[27] R. R. Yager and D. P. Filev, *Essentials of Fuzzy Modeling and Control*. John Wiley, New York, 1994.

[28] A. Lotfi, *Application of Learning Fuzzy Inference Systems in Electricity Load Forecast*. In report of the EUNITE worldwide competition on Electricity Load Forecast using Intelligent Techniques (ISBN: 80-89066-41-0), 2002.

[29] A. Lotfi and J. M. Garibaldi, *Applications and Science in Soft Computing*. Springer, ISBN: 3-540-40856-8, 2004.

[30] B. Kosko, “Fuzzy systems as universal approximators,” in *Proceedings of Int. Conf. Fuzzy Syst.*, pp. 1153–1162, 1998.

[31] R. Babuška, *Fuzzy Modeling and Identification*. PhD thesis, Delft University of Technology, 1996.

[32] H. A. Babuška, R. Braake, A. J. Krijgsman, and H. B. Verbruggen, “Comparison of intelligent control schemes for real-time pressure control,” *Control Engineering Practice*, vol. 4, pp. 1585–1592, 1996.

[33] A. Sheta, *Modeling the Tennessee Eastman Chemical Reactor Using Fuzzy Logic*. New York, USA: Book Chapter. The ISE Book Series on Fuzzy System Engineering-Theory and Practice, Nova Science, ISBN: 3-540-25322-X, 2005.

[34] A. Sheta, E. Oznergiz, M. Abdelrahman, and R. Babuska, “Modeling of hot rolling industrial process using fuzzy logic,” in *Proceedings of the ISCA 22nd International Conference on Computer Applications in Industry and Engineering (CAINE-2009), San Francisco, November 4 - 6, 2009, CA, USA*, pp. 81–86, 2009.

[35] M. Abdelrahman, A. Sheta, and W. Deabes, “Fuzzy mathematical modeling for reconstructing images in ECT of manufacturing processes,” in *Proceedings of the International Conference on Computer Engineering and Systems (ICCES2009), Ain Shams University, December 14-16, 2009, Cairo, Egypt*.

[36] R. Babuška, *Fuzzy Modeling and Identification Toolbox*. Delft University of Technology, The Netherland, http://lcewww.et.tudelft.nl/bubuska, 1998.

[37] D. E. Gustafson and W. C. Kessel, “Fuzzy clustering with a fuzzy covariance matrix,” in *Proceedings of the IEEE CDC, San Diego, CA, USA*, pp. 761–766, 1979.