Gradus Vol 5, No 1 (2018) 134-139
ISSN 2064-8014

SOFTWARE IMPLEMENTATION OF AUTOMATIC FUZZY SYSTEM GENERATION AND OPTIMIZATION

Ádám Bors[*]*, Zsolt Csaba Johanyák*

Department of Information Technology, GAMF Faculty of Engineering and Computer Science, John von Neumann University, Hungary

Keywords:
fuzzy logic
software
optimization
fuzzy system

Article history:
Received 30 March 2018
Revised 25 April 2018
Accepted 29 April 2018

Abstract
Automatic fuzzy system generation from sample data is a common task in fuzzy modeling. Here usually first an initial system is created using clustering, grid partitioning or other approaches and next, the parameters of the system are optimized based on the difference between the sample output and the output of the fuzzy system.
The software being presented in this paper supports the whole process providing a wide range of parameterization opportunities. It also includes an optimization toolbox that offers five optimization algorithms, from which one represents a novel approach. The proposed new algorithm was compared with four well-known methods using several benchmark functions and it ensured better results in case of many functions.

1. Introduction

The original goal of our project was to create a software capable creating a fuzzy system from sample data. The fuzzy system should work as similar as possible to the modelled system described by sample data. The creation of the fuzzy system would be done in two steps, first one generates an initial system using the sample input and output data, then the parameters of the systems are optimized to make it even more accurate. Multiple types of fuzzy set based solutions were meant to be implemented, as well as different optimization methods to choose from.

Basically optimization methods are used to find the combination of parameters that result in the most favorable output. In most of the cases the problem can be formulated in such a way that the goal becomes finding a minimum point. This is why the implementation the chosen methods targeted only the search for the minimum.

A test application was also developed for the optimization methods, which included several test-functions to see how well each method performed on various test functions. Measurements were made of their effectiveness and the tool can help with finding proper values for the optimization method parameters.

Having an optimization toolbox developed our research work focused more on the specific field of fuzzy model generation. Fuzzy model based solutions are widely used to calculate an output value from given input values according to a set of rules that specify the connection between antecedent and consequent fuzzy sets. Fuzzy systems can be used in a wide variety of cases and require fairly low computational power to process even in complex cases. Being able to acquire an appropriate system usually poses a challenge, but our tool could present a solution.

The rest of this paper is organized as follows. Section 2 gives a short introduction in the optimization methods included into the optimization toolbox presenting the main ideas of the mentioned methods. Section 3 presents the results of the experimental investigation of the optimization techniques using five well-known benchmark functions. Section 4 focuses on fuzzy system generation and optimization and the conclusions are drawn in Section 5.

* Corresponding author: adam.bors4@gmail.com

2. Optimization Methods

The goal of an optimization process is finding a combination of values within given bounds that result in the most favorable output. Since the connection between the input and output sides is not always given with such a mathematical function for which one could determine its minimum point easily by analytical methods, we use a guided search that aims finding the global minimum of a so called fitness function. Most methods keep repeating a sequence of given steps until a certain condition is met, which can be one of the following.

- A solution has been found that ensures a fitness value below a threshold value.
- A given number of iterations have been reached.
- The number of fitness evaluations has exceeded a limit.

Further on the five optimization approaches that are included in our toolbox are presented shortly focusing on their main ideas only.

2.1. Firework Algorithm

The firework algorithm [13] unlike most other methods it was inspired by something man-made, rather than something natural. It selects several particles in each generation, then places new particles in their vicinity. This gives them a similar appearance to fireworks, where multiple sparks appear around a central location. The selected particles are chosen in such a way, that ones with good and bad fitness values alike will make up the centers of the next generation's explosions. This behavior lets it search even for value combinations that are not close to the currently known best particle, but often at the cost of slower convergence.

2.2. Particle Swarm Optimization Algorithm

The particle swarm optimization (PSO) algorithm [5] is a nature inspired method. The particles have their own knowledge of the search space, but also share some of it with each other. This way they move towards not just the globally known best position, but their own previously found best positions as well. They also maintain a certain amount of their velocity, which lets them move to previously unexplored regions of the search space. In each generation they take a small step based on their current velocity, which is then recalculated. This algorithm finds minimum points quickly and accurately, but tends to get stuck in local minima.

2.3. Genetic algorithm

The genetic algorithm (GA) [3] resembles how genes are inherited, with some added randomness. Well-performing particles are crossed over and some of the newly created particles have random mutations applied to them. This way the algorithm always has a chance to find a better solution even when it is seemingly trapped in a local minimum.

2.4. Clonal selection algorithm

The fourth optimization method in the toolbox is the clonal selection algorithm [1], which belongs to the family of artificial immune system algorithms. It makes copies of the best performing antibodies and makes randomized changes to them. Since it only uses the available information to a small degree, it rarely gets stuck in local minima, but it also does not give us the best possible solution always. It is efficient at finding a point with great fitness, but has trouble when it comes to finding an even better one, even when it would be close to the previous one.

2.5. Modified clonal selection algorithm

Seeing how the previously mentioned method performs well, bit has a significant downside, we decided to improve it by adding a local search step to the end of each generation, which supplements the existing global search. This involves trying to change the position of the best antibody by a small randomized number along each axis and overwriting the original when a better solution is found. Due to this modification, the algorithm continuously tries to find points around the

best performing particle that have even better fitness, instead of trying to find these points across the whole search space randomly. The modified algorithm was found to give better results than the original one.

3. Experimental Evaluation of the Optimization Methods

In order to compare the implemented well-known methods and the proposed new one we tested them against five widely used benchmark functions introduced in [10]. The tests ran until the processes reached a predefined number of evaluations, which gave results showing how good of a solution they found in approximately the same time interval. The algorithms were put through different challenges by the test functions. The results can be seen in *Table 1*.

Table 1. The results of the optimization tests on various functions

	Firework	*Particle swarm*	*Genetic algorithm*	*Clonal generation*	*Modified clonal generation*
Parabola	0.003891	0	0.000482	0.001047	0.000237
	0.000229	0	0.000506	0.000218	0.000011
	0.004353	0	0.001721	0.000875	0.000084
Average	0.002824333	**0**	0.000903	0.000713333	0.000110667
Tripod	2.000011	0	1.056429	0.016366	0.008281
	1.219455	2	0.165951	0.014777	0.002298
	1.000234	0	0.021329	0.02639	0.018436
Average	1.406566667	0.666666667	0.414569667	0.019177667	**0.009671667**
Alpine	0.000635	0	0.000091	0.000907	0.000252
	0.174246	0	0.002578	0.000364	0.000529
	0.001375	0	0.001287	0.000736	0.000241
Average	0.058752	**0**	0.001318667	0.000669	0.000340667
Griewank	0.04685	0.007396	0.000113	0.007429	0.007398
	0.004828	0	0.000853	0.007402	0.000011
	0.018313	0.007396	0.007433	0.007454	0.007402
Average	0.023330333	0.004930667	**0.002799667**	0.007428333	0.004937
Rosenbrock	0.000264	0	5.048517	0.425764	0.002665
	27.47576	0	24.149024	0.399099	0.009095
	9.861422	4.031709	1.579632	0.048878	0.03909
Average	12.44581533	1.343903	10.25905767	0.291247	**0.01695**
Overall average	2.787457733	0.403100067	2.135729733	0.063847067	**0.006402**
Number of generations	501	996	452	123	118

Each algorithm had their own strengths and weaknesses, but their performance depends significantly on their parameters and in part on the randomly generated initial particles/antibodies. The modified clonal selection performed better than its original counterpart in all of the cases. The local search means more evaluations per generation, so the number of generations is slightly lower. This trade-off is greatly compensated by the much lower fitness value found.

4. Fuzzy System Generation and Optimization

The concept of fuzzy sets and logic was originally developed by Lotfi A. Zadeh [15]. Since then intensive research work have been done in this field. Fuzzy set based solutions have many

practical applications ([8][9][2][11][14]) and several different methods were proposed for inference and different calculations. We implemented three different inference types, i.e. the Mamdani type [7], the Larsen type [6] and the Takagi-Sugeno type [12]. They have their characteristics, but each serve the general purpose of taking input values and returning output values using fuzzy sets and rules. Nature inspired and other heuristic optimization methods are widely used in fuzzy model identification (see e.g. [8][9])

The software creates the fuzzy system in two steps. First, an initial system is generated using grid partitioning using the same method as presented in [4]. Next, the parameters of the initial system are optimized to obtain the most possible similarity between the sample output values and the output values created by the fuzzy system. During the fine-tuning the shape and position of the fuzzy sets are modified. This is done by minimizing the difference between the original output values and the ones calculated using the generated system. The mean squared error (MSE) or its root (RMSE) can be used as performance indicators in this process.

The correct generation of fuzzy control systems was verified using corresponding input and output data. First the systems were generated, then the mean square root (PI_{MSE}) was calculated between the original output values and the ones calculated by the new system. This can be seen in *Table 2*.

Table 2. The results of the fuzzy system generation

Membership function count	PI_{MSE}
2	0.385892
3	0.35489
4	0.104177
5	0.17099
6	0.166918
7	0.117259
8	0.021361
9	0.006942
10	0.01412
15	0.001591

Table 3. The results of the fuzzy control system optimization

Membership function count	*Optimized parameter*	*Optimization target*	*Original* PI_{MSE}	*Optimized* PI_{MSE}
3	Base ratio	Input	0.595727	0.58502 0.585112 0.584901
		Output		0.576314 0.576310 0.576307
		Input and output		0.563336 0.563123 0.562823
	Reference point	Input		0.198174 0.209501 0.230194
		Output		0.519039 0.519028 0.519022
		Input and output		0.122615

				0.123118 0.168264
Membership function count	*Optimized parameter*	*Optimization target*	*Original PI_{MSE}*	*Optimized PI_{MSE}*
15	Base ratio	Input	0.03989	0.039797 0.039799 0.039799
		Output		0.037674 0.037674 0.037674
		Input and output		0.037674 0.037674 0.037674
	Reference point	Input		0.009077 0.02785 0.009077
		Output		0.032407 0.021563 0.023256
		Input and output		0.006705 0.007602 0.009077

The values show that the system works appropriately, with generally smaller error when more fuzzy sets are used in a partition. This shows that the first step is successful and even in this stage we have a usable product.

The second step is the optimization of the generated system. Measurements were made on how the PI_{MSE} values changed after an optimization, the results of which can be seen in *Table 3*.

The effect of the optimization varies depending on the characteristics of the data, but it makes a noticeable difference in most of the cases. Optimizing the reference point led to a bigger improvement than that of the base ratio, this is due to the former being able to make a relatively large change to the system compared to the latter.

5. Conclusions and summary

The test results show that the software package can be used to run optimization processes, effectively perform fuzzy set based solutions and generate new systems that can be used in various scenarios. A potential use case would be using the application to generate a system based on existing data, optimizing in and also integrating the fuzzy solver library into an application. That way the generated system can be used for decision making purposes.

Acknowledgment

This research was supported by the ÚNKP-17-A-PAE-39 New National Excellence Program of the Ministry of Human Capacities and by EFOP-3.6.1-16-2016-00006 "The development and enhancement of the research potential at John von Neumann University" project. The Project is supported by the Hungarian Government and co-financed by the European Social Fund.

References

[1] De Castro, L. N., & Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. IEEE transactions on evolutionary computation, 6(3), 239-251.

[2] E.H. Guechi, J. Lauber, M. Dambrine, G. Klančar and S. Blažič (2010): PDC control design for non-holonomic wheeled mobile robots with delayed outputs, Journal of Intelligent and Robotic Systems, vol. 60, no. 3-4, pp. 395-414, Dec. 2010.

[3] J.H. Holland, Genetic Algorithms, Scientific American, July 1992, pp 66-72.
[4] Z.C Johanyák: New Initial Fuzzy System Generation Features in the SFMI Toolbox, 5th IEEE International Symposium on Logistics and Industrial Informatics (LINDI 2013), Wildau, Germany, September 5-7, 2013, pp. 29-34.
[5] Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. in Proceedings of IEEE International Conference on Neural Networks IV., Perth, 1995, 1942–1948.
[6] Larsen, P. M.: Industrial application of fuzzy logic control, in International Journal of Man Machine Studies, Vol. 12(4), 1980, pp. 3-10.
[7] Mamdani, E. H. and Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller, in International Journal of Man Machine Studies, Vol. 7, 1975, pp. 1-13.
[8] R.-E. Precup, M.-C. Sabau, and E. M. Petriu, Nature-inspired optimal tuning of input membership functions of Takagi-Sugeno-Kang fuzzy models for anti-lock braking systems, Applied Soft Computing, vol. 27, pp. 575-589, Feb. 2015.
[9] R.-E. Precup, R.-C. David, and E. M. Petriu, Grey wolf optimizer algorithm-based tuning of fuzzy control systems with reduced parametric sensitivity, IEEE Transactions on Industrial Electronics, vol. 64, no. 1, pp. 527-534, Jan. 2017.
[10] Raj, Ashish, Evolutionary Optimization Algorithms for Nonlinear Systems (2013). All Graduate Theses and Dissertations. Paper 1520. 22-31.
[11] I. Škrjanc and S. Blažič (2005): Predictive functional control based on fuzzy model: design and stability study, Journal of Intelligent and Robotic Systems, vol. 43, no. 2-4, pp. 283-299, Aug. 2005.
[12] Takagi, T. and Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control, in IEEE Transactions on System, Man and Cybernetics, Vol. 15, 1985, pp. 116-132.
[13] Tan, Y., & Zhu, Y. (2010, June). Fireworks algorithm for optimization. In International Conference in Swarm Intelligence. Springer Berlin Heidelberg. 355-364
[14] Ján Vaščák (2012): Adaptation of fuzzy cognitive maps by migration algorithms, In: Kybernetes, Vol. 41, no. 3/4, Mar. 2012, pp. 429-443, ISSN 0368-492X.
[15] Zadeh, L.A.: Outline of a new approach to the analysis of complex systems and decision processes, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 3, No. 1, Jan. 1973, pp. 28-44.