Optimizing Fuzzy Flip-Flop Based Neural Networks by Bacterial Memetic Algorithm

Rita Lovassy[1,2] László T. Kóczy[1,3] László Gál[1,4]

1 Faculty of Engineering Sciences, Széchenyi István University Győr, Hungary
2 Inst. of Microelectronics and Technology, Kandó Kálmán Faculty of Electrical Engineering, Budapest Tech, Budapest, Hungary
3 Dept. of Telecommunication and Media Informatics, Budapest University of Technology and Economics, Hungary
4 Dept. of Technology, Informatics and Economy
Szombathely University of West Hungary
Email: lovassy.rita@kvk.bmf.hu, koczy@sze.hu, gallaci@ttmk.nyme.hu

Abstract—In our previous work we proposed a Multilayer Perceptron Neural Networks (MLP NN) consisting of fuzzy flip-flops (F^3) based on various operations. We showed that such kind of fuzzy-neural network had good learning properties. In this paper we propose an evolutionary approach for optimizing fuzzy flip-flop networks (FNN). Various popular fuzzy operation and three different fuzzy flip-flop types will be compared from the point of view of the respective fuzzy-neural networks' approximation capability.

Keywords— Bacterial Memetic Algorithm, feedbacked fuzzy J-K and fuzzy D flip-flops, Multilayer Perceptron Neural Networks

1 Introduction

Fuzzy set theory, artificial neural networks, bacterial evolutionary algorithms and their hybrid combination have been the subject of intense study and application, especially in the last decades. These were developed with the aim to deal with problems which were hard to solve using traditional techniques. Fuzzy systems are transparent and interpretable, neural networks possess the property of auto-adaptability, while evolutionary, especially bacterial algorithms have been used for the approximation of the optimal structure. To approximate various test functions we use a combination of the above mentioned three main branches of Computational Intelligence. Although some paper report about Support Vector Machines (SVM) outperforming traditional Multilayer Perceptron (MLP) in classification task, but in prediction and regression problems the MLP gives smaller errors with lower network complexity [12]. The paper is organized as follows. After the Introduction Section 2 defines triangular norms and co-norms; and then highlight the five well known fuzzy operations (algebraic, Yager, Dombi, Hamacher and Frank). Section 3 presents the concept of a single fuzzy J-K and D flip-flop, using the fundamental equation as it was proposed in [13]. A comparative study of several types of fuzzy J-K flip-flops based on the five fuzzy operations is made. Comparisons of fuzzy J-K flip-flops with feedback and of two different interpretations of fuzzy D flip-flops are presented. Section 4 is devoted to the investigation of the F^3 based neurons and the MLP [11] constructed from them. We proposed the Fuzzy Flip-Flop Neural Network (FNN) architecture that can be used for learning and approximating various simple transcendental functions. In Section 5 we present Bacterial Memetic Algorithm with Modified Operator Execution Order (BMAM) [5] for FNN variables optimization. Experimental results and comparison between different types of FNNs trained with Levenberg-Marquardt (LM) method and the BMAM are discussed in Section 6, followed by a brief conclusion in the last Section.

2 Fuzzy operations

Klement, Mesiar and Pap enumerate and give the basic definitions and properties of the most general triangular norms in [8], including also graphical illustrations, comparisons. In general a triangular norm [8] (t-norm) is a binary operation T on the unit interval [0, 1], i.e., a function $T:[0,1]^2 \rightarrow [0,1]$, such that for all $a, b \in [0,1]$ the commutativity, associativity, monotonicity and boundary conditions are satisfied. Triangular co-norms (s-norm) were introduced [8] as dual operations of t-norms.

Table 1 shows various t-norms and s-norms selected by us, (as possible bases of the FNN), all five can be used as basic operators in the definition of the fundamental formulas of the fuzzy flip-flop introduced in the next Section. (Many other did not produce good enough neural network properties, cf. [10].) For simplicity we denote the t-norm with i, and the s-norm with u, where the subscripts refer to the initial of the name of the norm: e.g., in case of Yager norms: $i_Y(a,b) = a\ i_Y\ b$ and $u_Y(a,b) = a\ u_Y\ b$.

3 Fuzzy flip-flops

3.1 Fuzzy J-K Flip-Flops

The fuzzy J-K flip-flop is an extended form of the binary J-K flip-flop. In this approach the truth table for the J-K flip-flop is fuzzified, where the binary NOT, AND and OR operations are substituted by their respective fuzzy counterparts, i.e. fuzzy negation, t-norm, and co-norm respectively.

ISBN: 978-989-95079-6-8

Table 1: Some t-norms and co-norms.

Fuzzy operation	t-norm; i (a, b)	s-norm; u (a, b)
Algebraic (A)	ab	$a+b-ab$
Yager (Y)	$1-\min\left[1,((1-a)^w+(1-b)^w)^{1/w}\right]$	$\min\left[1,(a^w+b^w)^{1/w}\right]$
Dombi (D)	$\frac{1}{1+\left[(1/a-1)^\alpha+(1/b-1)^\alpha\right]^{1/\alpha}}$	$\frac{1}{1+\left[(1/a-1)^{-\alpha}+(1/b-1)^{-\alpha}\right]^{-1/\alpha}}$
Hamacher (H)	$\frac{ab}{\gamma+(1-\gamma)(a+b-ab)}$	$\frac{a+b-(2-\gamma)ab}{1-(1-\gamma)ab}$
Frank (F)	$\log_s\left[1+\frac{(s^a-1)(s^b-1)}{s-1}\right]$	$1-\log_s\left[1+\frac{(s^{1-a}-1)(s^{1-b}-1)}{s-1}\right]$

Parameters w, α, γ and s lie within the open interval $(0, \infty)$. The next state $Q(t+1)$ of a J-K flip-flop is characterized as a function of both the present state $Q(t)$ and the two present inputs $J(t)$ and $K(t)$. (For simplicity (t) will be omitted in the next.) The so called fundamental equation of the J-K type fuzzy flip-flop [13] is

$$Q(t+1) = (J \vee \neg K) \wedge (J \vee Q) \wedge (\neg K \vee \neg Q) \quad (1)$$

where $\neg, \wedge, \vee$ denote fuzzy operations (e.g. $\neg K = 1-K$). As a matter of course, it is possible to substitute the standard operations by any other reasonable fuzzy operation triplet (e.g. De-Morgan triplet), thus obtaining a multitude of various fuzzy flip-flop pairs.

In [9] we studied the behavior of fuzzy J-K flip-flops based on various fuzzy operations, and illustrated their behavior by the graphs belonging to the next states of fuzzy flip-flops for typical values of Q, J and K.

3.2 Fuzzy D Flip-Flops

Connecting the inputs of the fuzzy J-K flip-flop in a particular way, namely, by applying an inverter in the connection of the input J to K, case of $K = 1-J$, a fuzzy D flip-flop is obtained. Substituting $D = \neg K = J$ in equation (1), the fundamental equation of fuzzy D flip-flop will be

$$Q(t+1) = (D \vee D) \wedge (D \vee Q) \wedge (D \vee \neg Q) \quad (2)$$

As an alternative approach, Choi and Tipnis [4] proposed an equation which also exhibits the characteristics of a fuzzy D flip-flop, as follows

$$Q(t+1) = D \wedge (D \vee Q) \wedge (\neg Q \vee D) \quad (3)$$

We will refer to this type of fuzzy D flip-flop as Choi (type fuzzy) D flip-flop (because of the first author). Comparing the characteristical equation of the fuzzy D flip-flop (2), with expression (3), there is essential difference between the two fuzzy flip-flops.

As $D = D \wedge D$, and $D = D \vee D$ i.e. the t-norm (T) and co-norm (S) are idempotent [1], hold only in the exceptional case ($T(x, x) = x$ and $S(x, x) = x$ for all $x \in [0,1]$); when

T = min, and S = max.

For example, using the algebraic norm

$$u_A(a,a) = a + a - a \cdot a = 2 \cdot a - a^2 = a \quad (4)$$

is satisfied only in the two borderline cases, when $a = 0$, or $a = 1$. It is surprising how much the satisfaction of idempotence influences the behavior of the fuzzy D flip-flops, as it will be shown.

In the next Section we will give an overview of the different type fuzzy J-K, D and Choi D F^{3}s, based on familiar norms listed in Table 1.

3.3 Fuzzy Flip-Flops Based on Various Fuzzy Operations

The behavior of fuzzy flip-flops based on algebraic, Yager, Dombi, Hamacher and Frank t-norms, combined with the standard negation $c(a)$=1-a in every case has been analyzed and compared [9] , in order to investigate, whether and to what degree they present more or less sigmoidal (S-shaped) $J \to Q(t+1)$ transfer characteristics in the particular cases, when K=1-Q, K=1-J, with a fixed value of Q. F^3 based algebraic type t-norm presents non-sigmoidal behavior, with piecewise linear characteristics and several breakpoints, but having the advantage of the hardware implementation of F^3 [14], this type was also studied for comparison.

3.4 Fuzzy J-K, D and Choi D Flip-Flops Based on Some Classes of Fuzzy Set Unions and Intersections

Using a triplet consisting of the standard negation, and a dual pair of fuzzy operations, i.e., using Yager t-norm and s-norm, the maxterm form in the unified equation (1) can be rewritten as:

$$Q_Y(t+1) = \left(J \; u_Y \left(1-K\right)\right) \; i_Y \; \left(J \; u_Y \; Q\right) \; i_Y \; \left(\left(1-K\right) \; u_Y \left(1-Q\right)\right) \quad (5)$$

This is the fundamental equation of the Yager type fuzzy J-K flip-flop.

By substituting the above mentioned triplet into equations (2) and (3) we defined the fundamental equations of the Yager type fuzzy D (6) and Yager type Choi D (7) flip-flops.

$$Q_Y(t+1) = (D \; u_Y \; D) \; i_Y (D \; u_Y \; Q) \; i_Y \; (D \; u_Y \; (1-Q)) \quad (6)$$

$$Q_Y(t+1) = D \; i_Y (D \; u_Y \; Q) \; i_Y \; ((1-Q) \; u_Y \; D) \quad (7)$$

In a very similar way, in the cases of algebraic, Dombi, Hamacher and Frank operation triplets we defined the corresponding fundamental equations of the respective fuzzy J-K, D and Choi D flip-flops.

ISBN: 978-989-95079-6-8

The parameter values in the Yager, Dombi, Hamacher and Frank norms strongly influence the $J \to Q(t+1)$ characteristics curvature. We have compared the characteristics for various typical parameter values and choose w=2, α=2, γ=10 and s=100, where we obtained more or less S-shaped $J \to Q(t+1)$ characteristics. A change of the t-norms in the characteristical equations of fuzzy J-K, D and Choi D flip-flops leads to the modification of the slope of the transfer function, which will affect the learning rate in the implementation of neural networks.

4 Fuzzy flip-flop based neurons

In this section we propose the use of the fuzzy flip-flops discussed above as neurons in a MLP. In the next, we study the effect of applying the mentioned five t-norms and co-norms in the investigation of the F^3 based neurons and the MLPs constructed from them. An important aspect of these F^3s is that they all have a certain convergent behavior when their input J is excited repeatedly. This convergent behavior guarantees the learning property of the networks constructed this way.

4.1 Function Approximation by Fuzzy Flip-Flop Network Trained with Levenberg – Marquardt Algorithm

A fuzzy flip-flop based supervised feedfordward backpropagation network is applied in order to approximate test functions. This function is represented by a set of 100 input/output data sets. All the input and output signals are distributed in the unit interval. In general, two trainable layer networks with sigmoid transfer functions in the hidden layer and linear transfer functions in the output layer have good approximation and interpolation properties [7], analogously the neural system model proposed is based on two hidden layers constituted from fuzzy flip-flop neurons.

The nonlinear characteristics exhibited by fuzzy neurons are represented by quasi sigmoid transfer functions given by fuzzy J-K, D and Choi D flip-flops based on algebraic, Yager, Dombi, Hamacher and Frank operations. The proposed network activation function is the same at each hidden layer, from unit to unit. For simplicity we did not apply activation function or threshold to the output layer, and considered our model with only one output. The number of neurons was chosen after experimenting with different size hidden layers. Smaller neuron numbers in the hidden layer result in worse approximation properties, while increasing the neuron number results in better performance, but longer simulation time. In our approach the weighted input values are connected to input J of the fuzzy flip-flop based on a pair of t-norm and t-conorm, having quasi sigmoid transfer characteristics. The output signal is then computed as the weighted sum of the input signals, transformed by the transfer function.

During network training, the weights and thresholds are first initialized to small, random values and the network was trained with Levenberg-Marquardt algorithm with 100 maximum numbers of epochs as more or less sufficient. First we fixed the activation function, the number of layers and the number of units in each layer. The chosen target activation function was the *tansig* (hyperbolic tangent sigmoid transfer function). This function is well suited to the demands of backpropagation.

5 Bacterial Memetic Algorithm with Modified Operator Execution Order (BMAM)

The Bacterial Memetic Algorithm with Modified Operator Execution Order (BMAM) [5] is a very recent soft computing tool. It combines a special kind of Genetic Algorithm [6] - Pseudo-Bacterial Genetic Algorithm (PBGA) - and the Levenberg-Marquardt (LM) method. We use this algorithm for training fuzzy flip-flop based neural network (FNN) to improve function approximation performance. The Bacterial Memetic Algorithm (BMA) resulted in better approximation properties in fuzzy modeling problems than the Bacterial Evolutionary Algorithm (BEA) (which outperformed the traditional Genetic Algorithms [2], [3]).

The core of BMAM contains the bacterial mutation, which is inspired by the biological bacterial cell model. Its basic idea is to improve the parts of chromosomes contained in each bacterium. The bacterial mutation mechanism is used by the bacteria which can transfer genes to other bacteria. To find the optimal approximation for our network we encoded our FNN weighs, biases, Q and fuzzy operation parameter values in a bacterium (chromosome). Therefore a procedure is working on changing the variables, testing the model obtained in this way and selecting the best models.

5.1 Function Approximation by Fuzzy Flip-Flop Network Trained with BMAM

The learning of the FNN is formulated as a parameter optimization problem, using the mean square error as the fitness evaluation set-up. The basic steps followed by the algorithm embrace the bacterial mutation operation and the LM method.

In the initial population a number of individuals are randomly created and evaluated. Next, an evolutionary cycle is started by applying the bacterial mutation operation for each individual. A number of copies (clones) of the bacterium are generated. Then the same part or parts of the chromosome is choose and mutate randomly, except one single clone that remains unchanged during this mutation cycle. The LM method nested into evolutionary algorithm is applied for a few times for each individual. The selection of the best clone is made and transfers its parts to the other clones. The part choosing-mutation-LM method-selection-transfer cycle is repeated until all the parts are mutated, improved and tested. The best individual is remaining in the population, and all other clones are deleted. This process is repeated until all the individuals have gone through the modified bacterial mutation.

6 Simulation results

The combination of two sine wave forms with different period lengths as test function was,

$y = \sin(c_1 * x) * \sin(c_2 * x)/2+0.5$,

ISBN: 978-989-95079-6-8

where the input vector x generated a sinusoidal output y. The values of constants c_1 and c_2 were selected to produce a frequency proportion of the two components 1:0.35.
The function approximation goodness depended on the flexible Q and fuzzy operation parameter values, on the fuzzy flip-flop types, hidden layer neuron numbers and in addition on the fuzzy operations themselves. To check the goodness of training, we used the Mean Squared Error (MSE) as a measure of the error made by the FNN. The simulation results are summarized in the next two tests.

6.1 Test 1

In one of our earlier paper [9] we optimized the value of Q, for every combination of J-K, D and Choi D type F^3s with all five fuzzy operation pairs, the parameters of the norms chosen for suitable values, solving the problem of the selection of the value of Q with the fine tuning of FNN.
The fuzzy flip-flop based FNNs are based on algebraic, further parametric Yager, Dombi, Hamacher and Frank norms. Table 2 shows the experimental results obtained with LM optimization [9].

Table 2: Q optimums obtained with LM algorithm.

	Fuzzy neuron type		
Fuzzy operation	J-K	D	Choi D
Algebraic (A)	**0.21**	**0.15; 0.91**	**0.09; 0.81**
Yager (Y)	0.06	0.20	0.17
Dombi (D)	**0.11**	0.92	0.06
Hamacher (H)	0.32	0.57	0.38
Frank (F)	0.31	0.45	0.55

The $J \to Q(t+1)$ fuzzy D and Choi D flip-flop characteristics based on algebraic norm present about the same sigmoidal character in multiple points of the domain [7], which fact can leads to rather different optimum results, corresponding to minimal median MSE values. In Test 1 we propose a new method to find the optimal Q – fuzzy operation parameter pair for every combination of J-K, D and Choi D type F^3s by training a 1-8-8-1 FNN with bacterial memetic algorithm. The advantage of this network size lies in good learning capability. In our application a population with 29 parameters - according to the network size - was initialized. During simulations 20 generations of 5 individuals with 5 clones were chosen to obtain the best fitting variables. Table 3 shows the unique optimal Q values found by BMAM algorithm, which are approximately equal to the respective optimum values listed in Table 2. We obtained almost the same results after every run, although in short term the training is not always successful.

Table 3: Q optimums obtained with BMAM algorithm.

	J-K	D	Choi D
Algebraic	0.25	0.12	0.13

The experimental results in case of J-K F^3 type FNN based on Dombi norm further indicated that the proposed network had multiple optimal Q – α parameter pairs. The test revealed, for example, the Q = 0.12, α = 2.9 values as a result that was close to the optimum solution mentioned above.

6.2 Test 2

In the next we will compare the function approximation performance of FNN trained in two different ways. In the first one the network training function updates weight and bias values according to Levenberg-Marquardt optimization (section 4.1). In the second approach the FNN variables - weights, bias, for fixed Q and fuzzy operation parameter values - optimization was made by the bacterial memetic algorithm with the modified operation execution order (section 5.1). During the simulations we covered all 15 possible combinations of fuzzy J-K, D and Choi D type FNNs with all five fuzzy operation pairs to approximate the above mentioned test function.
By changing the number of the layers, we used a 1-4-3-1 FNN in order to have a network that approximated accurately and also fast enough the sine waves, to emphasize the difference between various FNNs.
The parameter of Dombi, Yager, Hamacher and Frank norms were fixed α=2, w=2, γ=10 and s=100, which values provided good learning and convergence properties.
In our approach we fixed the value of Q, the present state value belonging to each fuzzy flip-flop, according to the values in Table 2. Tables 4, 5 and 6 present the 30 runs average approximation goodness, by indicating the minimum, median, mean and maximum mean squared error of the training values for each of the *tansig*, algebraic, Yager, Dombi, Hamacher and Frank types of FNNs.
Figures 1 and 2 present the graphs of the simulations in case of fuzzy J-K flip-flop with feedback based neural network trained with LM and respectively BMAM algorithms. In both of cases the algebraic F^3 provides a fuzzy neuron with rather bad learning ability. Comparing the median MSE values, considering them as the most important indicators of trainability, the Yager and Dombi types FNNs performed best, they can be considered as rather good function approximators. Our simulations have shown that BMAM performs better in every case than the original LM technique.
Figures 3, 4 and 5, 6 compare the behavior of fuzzy D flip-flop and Choi type fuzzy D flip-flop based NNs. It is interesting that according to the numerical illustrations (Tables 5 and 6) the average of 30 run mean squared errors in these cases the best results after the idealistic *tansig* function are given by the Hamacher and Yager F^3, which are followed by the Dombi and finally by the algebraic one. The Hamacher and Yager types FNNs have excellent approximation properties. It is surprising how much the satisfaction of idempotence influences the behavior of the fuzzy D flip-flop based NN. Comparing the simulation results belonging to the two types of fuzzy D flip-flop with the same norms, it can be seen that, for the same value of Q, the value of the MSE differs, which fact leads to a rather different behavior in the applications.
Using BMAM algorithm in the FNN variable optimization task, we obtained better function approximation capability with lower error than in case of FNNs trained with LM method.

ISBN: 978-989-95079-6-8

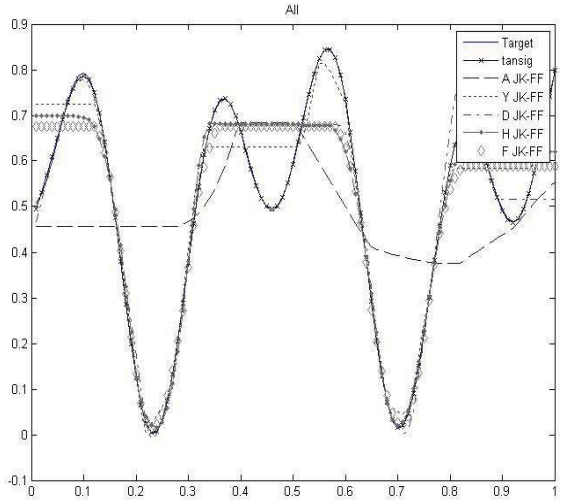

Figure 1: Simulation results of J-K FNN trained with LM algorithm.

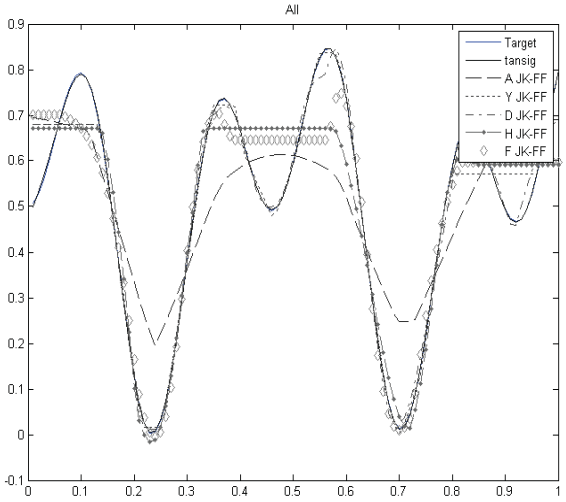

Figure 2: Simulation results of J-K FNN trained with BMAM algorithm.

Table 4: Mean squared error values case of JK FNN.

	LM				BMAM			
	min	median	mean	max	min	median	mean	max
tansig	6.6912×10^{-6}	**0.0056865**	0.0185562	0.0600578	4.8485×10^{-6}	**7.5218×10^{-5}**	0.0006223	0.0041448
A	0.04402231	**0.0548533**	0.0555179	0.0818562	0.00888158	**0.0170071**	0.0195847	0.0368526
Y	0.00286463	**0.0534354**	0.0474332	0.0615071	0.00133805	**0.0031207**	0.0310288	0.0064944
D	0.00420610	**0.0512862**	0.0439112	0.0691943	9.2216×10^{-5}	**0.0022534**	0.0024855	0.0052513
H	0.00551096	**0.0385112**	0.0400807	0.0720822	0.00351781	**0.0071140**	0.0081815	0.0256592
F	0.05552121	**0.0485408**	0.0425236	0.0660629	0.00228814	**0.0066725**	0.0076724	0.0312439

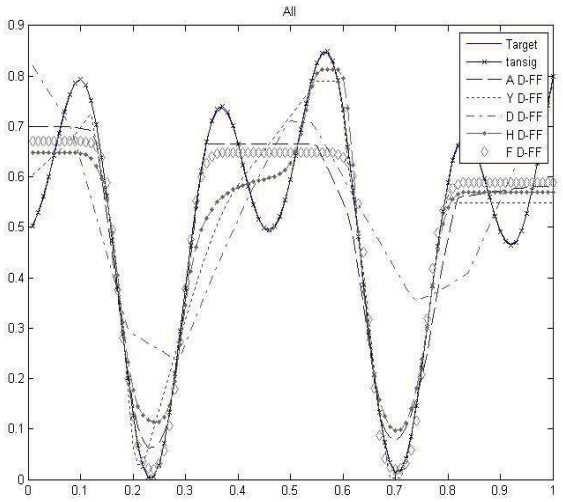

Figure 3: Simulation results of D FNN trained with LM algorithm.

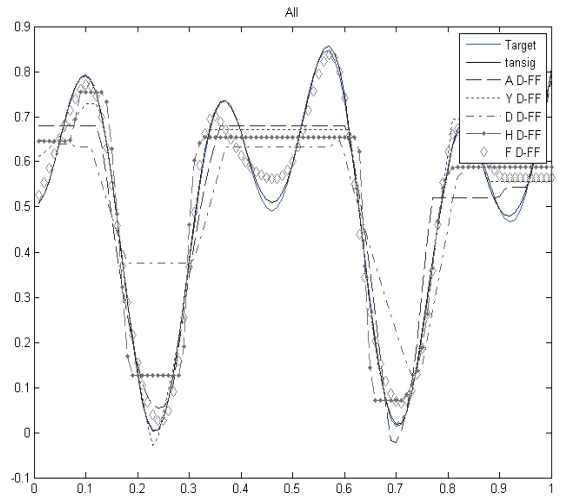

Figure 4: Simulation results of D FNN trained with BMAM algorithm.

Table 5: Mean squared error values case of D FNN.

	LM				BMAM			
	min	median	mean	max	min	median	mean	max
tansig	4.3611×10^{-6}	**0.0068415**	0.0203523	0.0565211	9.6899×10^{-7}	**4.7893×10^{-5}**	0.0006056	0.0033605
A	0.00771386	**0.0531832**	0.0468789	0.0838021	0.00490354	**0.00918654**	0.0114988	0.0330397
Y	0.00562192	**0.0515206**	0.0448304	0.0653909	0.00031811	**0.00335034**	0.0033998	0.0065352
D	0.03089988	**0.0502911**	0.0502276	0.0692478	0.01059031	**0.02699423**	0.0255351	0.0365076
H	0.00511140	**0.0485498**	0.0425834	0.0679154	0.00017240	**0.00586916**	0.0073385	0.0271431
F	0.00694345	**0.0506654**	0.0458813	0.0640876	0.00051204	**0.00681308**	0.0065404	0.0229583

ISBN: 978-989-95079-6-8

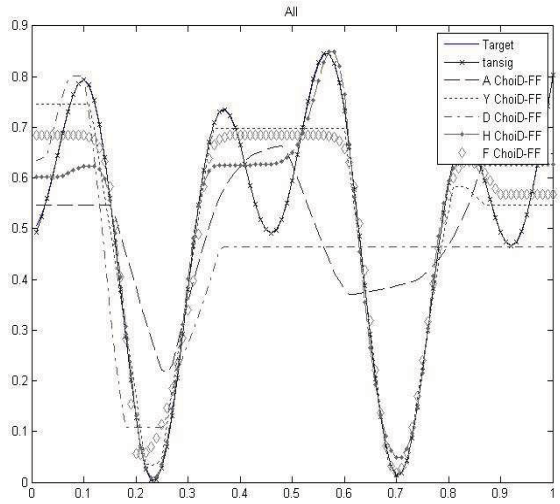

Figure 5: Simulation results of Choi D FNN trained with LM algorithm.

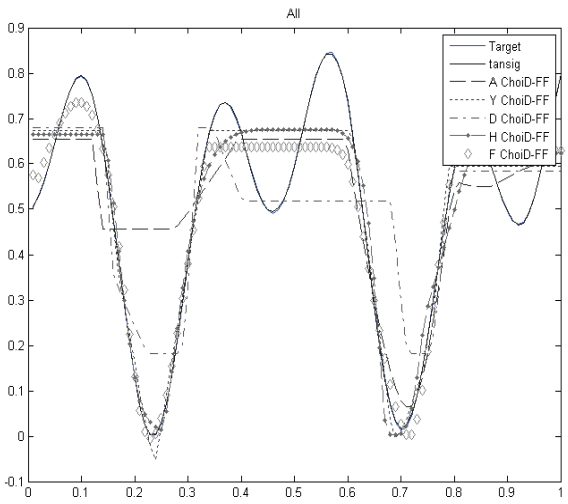

Figure 6: Simulation results of Choi D FNN trained with BMAM algorithm.

Table 6: Mean squared error values case of Choi D FNN.

	LM				BMAM			
	min	median	mean	max	min	median	mean	max
tansig	1.1454×10^{-6}	**0.0291153**	0.0251398	0.0624285	1.9737×10^{-7}	$\mathbf{2.7722 \times 10^{-5}}$	0.0005026	0.0024156
A	0.03693712	**0.0567222**	0.0557563	0.0878214	0.00777224	**0.0265908**	0.0251665	0.0363809
Y	0.00660108	**0.0508917**	0.0483636	0.0641307	0.00328527	**0.0077983**	0.0087765	0.0254508
D	0.03222194	**0.0534598**	0.0537578	0.0861916	0.01436651	**0.0235421**	0.0247559	0.0347627
H	0.00459423	**0.0497143**	0.0446552	0.0737876	0.00546122	**0.0077690**	0.0111944	0.0286523
F	0.00668745	**0.0518677**	0.0445716	0.0727711	0.00293771	**0.0092323**	0.0126012	0.0310469

7 Conclusions

In this paper we proposed the use of BMAM to optimizing fuzzy flip-flop based neural network (FNN). We compared the function approximation performance of five different types of FNNs, which depends from the choice of different fuzzy flip-flop types and from the training algorithm. The result of learning as well as its performance might differ quite significant under these two learning modes. The simulations have shown that the bacterial memetic algorithm can improve the function approximation capability by optimizing the FNN variables values.

Acknowledgment

This paper was supported by the Széchenyi University Main Research Direction Grant 2009, National Scientific Research Fund Grant OTKA T048832 and K75711, and National Office for Research and Technology.

References

[1] R. Bellman and M. Giertz, "On the analytic formalism of the theory of fuzzy sets", Information Science, vol. 5, 1973, pp. 149-156.

[2] J. Botzheim, C. Cabrita, L. T. Kóczy, A. E. Ruano, **"**Fuzzy rule extraction by bacterial memetic algorithms" International Journal of Intelligent Systems, 2009, pp. 312-339.

[3] J. Botzheim, C. Cabrita, L. T. Kóczy, A. E. Ruano, "Genetic and bacterial programming for B-spline neural networks design", Journal of Advanced Computational Intelligence and Intelligent Informatics, Vol. 11., No. 2, 2007, pp. 220-231.

[4] B. Choi and K. Tipnis, "New Components for Building Fuzzy Logic Circuits", Proc. of the 4th Int. Conf. on Fuzzy Systems and Knowledge Discovery, vol. 2, 2007, pp. 586-590.

[5] L. Gál, J. Botzheim and L. T. Kóczy, „Improvements to the Bacterial Memetic Algorithm used for Fuzzy Rule Base Extraction", Computational Intelligence for Measurement Systems and Applications, CIMSA 2008, Istanbul, pp.38-43.

[6] J. H. Holland, "Adaptation in Nature and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence", The MIT Press Cambridge, 1992.

[7] K. M. Hornik, M. Stinchcombe and H. White, "Multilayer feedforward networks are universal approximators", Neural Networks, vol.2, no.5, 1989, pp. 359-366.

[8] E. P. Klement, R. Mesiar and E. Pap, "Triangular Norms", Series: Trends in Logic , vol.8, 2000.

[9] R. Lovassy, L. T. Kóczy and L. Gál, "Function Approximation Capability of a Novel Fuzzy Flip-Flop Based Neural Network", IJCNN 2009 Atlanta- accepted.

[10] R. Lovassy, L. T. Kóczy and L. Gál, "Analyzing Fuzzy Flip-Flops Based on Various Fuzzy Operations", Acta Technica Jaurinensis Series Intelligentia Computatorica vol. 1, no. 3, 2008, pp. 447-465.

[11] R. Lovassy, L. T. Kóczy and L. Gál, "Multilayer Perceptron Implemented by Fuzzy Flip-Flops", IEEE World Congress on Computational Intelligence, WCCI 2008, Hong Kong, pp. 1683-1688.

[12] S. Osowski, K. Siwek and T. Markiewicz, "MLP and SVM Networks – a Comparative Study", Proceedings of the 6th Nordic Signal Processing Symposium, Finland, 2004, pp. 37-40.

[13] K. Ozawa, K. Hirota and L. T. Kóczy, "Fuzzy flip-flop", In: M. J. Patyra, D. M. Mlynek, eds., Fuzzy Logic. Implementation and Applications, Wiley, Chichester, 1996, pp. 197-236.

[14] K. Ozawa, K. Hirota, L. T. Kóczy and K. Omori, "Algebraic fuzzy flip-flop circuits", Fuzzy Sets and Systems 39/2, North Holland, 1991, pp.215-226.

ISBN: 978-989-95079-6-8