East African Scholars Journal of Engineering and Computer Sciences
Abbreviated Key Title: East African Scholars J Eng Comput Sci
ISSN 2617-4480 (Print) | ISSN 2663-0346 (Online) |
Published By East African Scholars Publisher, Kenya

OPEN ACCESS

Volume-2 | Issue-4 | Apr -2019 |

Review Article

Approximating the bell-shaped function based on combining hedge algebras and Particle swarm optimization

Nguyen Tien Duy[1] and Nghiem Van Tinh[1]

1Thai Nguyen University of Technology – Thai Nguyen University, ThaiNguyen, Vietnam

*Corresponding Author
Nghiem Van Tinh

Abstract: In the recent years, many researchers proposed various approximating functional methods by using fuzzy logic and hedge algebras (HA) but their results are not satisfied. To overcome these drawbacks, a novel method is proposed to increase the approximation accuracy of the function by using hedge algebra combined with particle swarm optimization (PSO) algorithm in this paper. First, the hedge algebras is applied to deal with the problems of approximate inference for a specific fuzzy model based on rule base for resolving approximate function problem. After that, combining the nonlinear interpolation with PSO algorithm to find optimal parameters of hedge algebras by minimizing objective function value. To verify the effectiveness of the proposed model, the bell-shaped function is selected to illustrate, and the simulation result shows that the proposed model gets approximation bell surface that is close to original bell-shaped function. This result is very encouraging for future work on the development of hedge algebras and PSO algorithm in the other approximating function problems.

Keywords: Approximation inference, identification, function approximation, hedge algebras, PSO.

1. INTRODUCTION

Nowadays, computer science can be regarded as an important science in the development of information technology. Beside theoretical mathematics, applied mathematics flourishes with the appearance and development of digital computers. In particular, the digital method is the science in the field of applied mathematics researching the approximate solutions of the equations, the problem of function approximation and optimization problems. Solving a problem of function approximation aims at changing a complex function such as an expression form or function as a table with simpler functions.

Interpolation is one of the methods to restore the continuous characteristics of a function $y = f(x)$ from the discrete data set by measurement or observation. When $f(x)$ is a complex function and difficult to compute, it also needs to be approximated by a polynomial. The simplest interpolation is the one by a polynomial.

In many fields of science and technology, as well in cybernetics, it is very difficult to determine the input-output relation of an element or a system with a function form. Searching the model approximation method based on a template data set is often carried out in order to build an approximation relationship with the acceptable error rate which its relationship is simpler. Therefore, analyzing and synthesizing the system are able to be more efficient (Nhu Lan Vu, & Tien Duy Nguyen 2007).

In the theory of function approximation, the problems of interpolation, even approximation and square approximation (known as the least square method) are often studied.

Quick Response Code

Journal homepage:
http://www.easpublisher.com/easjecs/

Article History
Received: 15.04.2019
Accepted: 28.04.2019
Published: 23.05.2019

DOI: 10.36349/easjecs.2019.v02i04.003

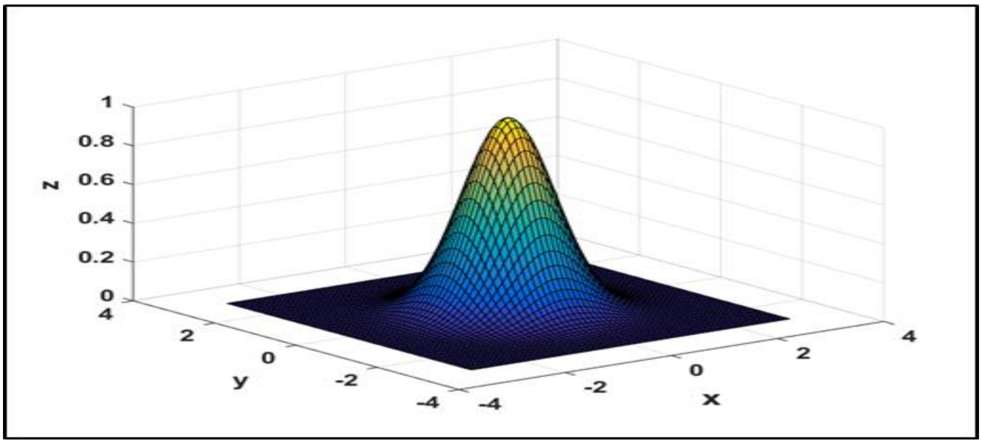

Figure 1. The bell "original" $z(x,y) = e^{-(x^2+y^2)}$

Many control problems under fuzzy approach have been solved quite efficiently since Prof. Lotfi Zadeh's fuzzy logic theory (California University, Berkeley) first introduced in 1965. Fuzzy logic allows performing linguistic values based on fuzzy set (Cao, Z., & Kandel A., 1989; Nguyen, C. H. *et al.*, 2014; Cat, H. N. *et al.*, 2014). In addition, a different computing approach on the language terms which are hedge algebras. The theory of hedge algebras has been introduced by N.C. Ho and Wechler on a strictly algebraic structure on linguistic variables (Ho N. C., & Wechler, W. 1990) since 1990. Whereas each linguistic value of the linguistic variable is quantified by a real value in the range [0, 1], semantically quantifying function formula was constructed based on parameters such as the fuzzy measurement of generating elements and hedges. In this study, we apply the hedge algebras to deal with the problems of approximate inference for a specific fuzzy model based on rule base for solving function approximation. By combining with nonlinear interpolation and PSO to optimize the fuzzy parameters, approximate computing values brings the more accurate results. To be able to evaluate the effectiveness of the proposed model, the bell - shaped function (Satish, K. 1999) are utilized as an "original" to illustrate the proposed method, give as follows:

$$z(x,y) = e^{-(x^2+y^2)} \qquad (1)$$

Where; x, y denote varies in the range = [-3, 3], step = 0.1, we can map the bell surface of the function z(x, y) as shown in Figure 1.

This paper includes four sections in addition to the introduction. In Sec. 2, a brief review of the basic concepts of HA and PSO algorithms are introduced. In Sec. 3, the approximation inference system based on HA is presented. Section 4 evaluates the performance of the proposed method. The last section summarizes the conclusions derived from the empirical results are given in Sec. 5.

Preliminaries

In this section, we first briefly introduce some concepts associated with HA (Ho N. C., & Wechler, W. 1990), PSO algorithm and then explain normalization and denormalization process of approximation inference system.

2.1. Hedge algebras

Suppose that there is a linguistic value set that is the language domain of speed language variable and includes these following words: X = *dom*(SPEED) = {*Very Very small* < *Very small* < *small* < *Little small* < *Very Little small* < *medium* < *Very Little big* < *Little big* < *big* < *Very big* < *Very Very big*, …}. The linguistic values are used in the approximation inference problems based on knowledge by rules. Obviously, it is necessary to have a structure strong enough which is based on the inherent order of the linguistic values in the domain of linguistic variable. Therefore, we can compute the semantic per linguistic value of linguistic variables in the problems of approximation inference.

Definition of hedge algebras:

Each linguistic variable X can be expressed as an algebraic structure AX= (X, G, C, H, ≤), called a hedge algebras, in which X is a set of terms in X; ≤ denotes the naturally semantic order relationships of the terms in X; $G = \{c^-, c^+\}$, $c^- \le c^+$, called the generating elements (For example, *G = {small, big})*; $C = \{0, W, 1\}$ is the set of constants, with $0 \le c^- \le W \le c^+ \le 1$, to indicate the elements that has the smallest, largest and neutral elements (For example, W = *medium*); $H = H^- \cup H^+$, with $H^- = \{h_j: -1 \le j \le -q\}$ is the set of negative hedge, $\forall h \in H^-$ then $hc^+ \le c^+$ and $H^+ = \{h_j: 1 \le j \le p\}$ is the positive hedge, $\forall h \in H^+$ then $hc^+ \ge c^+$. For example, H^-= {*Rather*, *Little*}, H^+= {*Very*, *More*}. With $x \in X$, $x = h_n h_{n-1} \dots h_1 c$, $c \in G$. Moreover, it can be seen that there is a comparable relationship existed among hedges and they can be shown by their signs (Cao, Z., & Kandel A., 1989; Cat, H. N. *et al.*, 2014) as follows:

Sign function: *Sgn*: X → {–1, 0, 1} is recursively defined:

With k, $h \in$ H, $c \in \{c^-, c^+\}$

$$Sgn(c+) = +1 \text{ and } sgn(c-) = -1 \quad (2)$$

$$\{h \in H+ \mid sgn(h) = +1\} \text{ and}$$

$$\{h \in H- \mid sgn(h) = -1\} \quad (3)$$

$$sgn(hc) = +sgn(c) \text{ if } hc > c \text{ and}$$

$$sgn(hc) = -sgn(c) \text{ if } hc < c \quad (4)$$

Or $$sgn(hc) = sgn(h) \times sgn(c) \quad (5)$$

sgn(*khx*) = +*sgn*(*hx*) if *k is* positive to *h* (*sgn*(*k*,*h*) = +1) and *sgn*(*khx*) = -*sgn*(*hx*) if *k* is negative to *h* (*sgn*(*k*,*h*) = +1).

Generally:

$\forall x \in H(G)$, it can be written as follow:

$x = h_m \dots h_1 c$, with $c \in G$ and $h_1, \dots, h_m \in H$.

Then: $$sgn(x) = sgn(h_m, h_{m-1}) \times \dots \times sgn(h_2, h_1) \times sgn(h_1) \times sgn(c) \quad (6)$$

$(sgn(hx) = +1) \Rightarrow (hx \geq x)$ and $(sgn(hx) = -1) \Rightarrow (hx \leq x)$

Fuzzy measurement:

The concept of "fuzzy" of a fuzzy language information is very important in computing the semantic value of the term (Ho N. C., & Wechler, W. 1990; Cat, H. N. *et al.*, 2014). The semantics of the language value in AX is built from the sets $H(x)=\{x=h_n h_{n-1} \dots h_1 c, c \in G, h_j \in H\} \cup \{x\}, x \in X$, it can be regarded as a fuzzy model of x. The set $H(x)$, $x \in X$ determines the fuzzy measurement *fm* of X and is equal to the "radius" of $H(x)$. It can be recursively computed from the fuzzy measurement of the generating elements, $fm(c^-)$, $fm(c^+)$ and the fuzzy measurement of hedges $\mu(h)$, $h \in H$. They are called the fuzzy parameters of X.

fm: X → [0, 1] is called the fuzzy measurement if it meets this following condition:

$$fm(c-) + fm(c+) = 1 \text{ and } \quad \sum_{h \in H} fm(hx) = fm(x), \text{ with } \forall x \in X \quad (7)$$

With the elements 0, W and 1,

$$fm(0) = fm(W) = fm(1) = 0 \quad (8)$$

And with $\forall x, y \in X, \ \forall h \in H$,

$$\frac{fm(hx)}{fm(x)} = \frac{fm(hy)}{fm(y)} \quad (9)$$

The equation (8) does not depend on the elements x, y, it characterizes as the hedge h, called the fuzzy measurement of h, denoted as $\mu(h)$. The characteristics of $fm(x)$ and $\mu(h)$ are as follows:

We have $x \in X$, $x = h_n h_{n-1} \dots h_1 c$,

$$fm(hx) = \mu(h) fm(x), \forall x \in X \quad (10)$$

$$fm(hnhn - 1 \dots h1c) = \mu(hn) \dots \mu(h1) fm(c), c \in G \ (11)$$

$$\sum_{i=-1}^{-q} \mu(h_i) = \alpha \text{ And} \sum_{i=1}^{p} \mu(h_i) = \beta, \text{ with } \alpha, \beta > 0 \text{ and } \alpha + \beta = 1 \quad (12)$$

Semantically Quantifying Mapping*: With* the fuzzy parameter set, semantically quantifying value is determined by semantically quantifying mapping function *(SQM - Semantically quantifying Mapping)* v recursively as follows:

$$\upsilon(W) = \theta = fm(c-) \quad (13)$$

$$\upsilon(c-) = \theta - \alpha.fm(c-) = \beta.fm(c-) \quad (14)$$

$$\upsilon(c+) = \theta + \alpha.fm(c+) \quad (15)$$

$$v(h_j x) = v(x) + Sgn(h_j x) \left\{ \left[\sum_{i=Sgn(j)}^{j} fm(h_i x) \right] - \omega(h_j x) fm(h_j x) \right\} (16)$$

With: $\omega(h_j x) = \frac{1}{2}[1 + Sgn(h_j x) Sgn(h_p h_j x)(\beta - \alpha)], j \in [-q \wedge p] = [-q, p] \backslash \{0\}$

Semantically quantifying mapping function (*SQMs*) can be directly mapped from linguistic values to semantically quantifying value of its semantics. Therefore, based on SQMs, we can simulate approximation inference methods of human which always ensures the semantics order of the language.

2.2. The Pso Algorithm

Kennedy and Eberhart (1995) proposed traditional particle swarm optimization (PSO) techniques for dealing with optimization problems, where a set of potential solutions is represented by a swarm of particles and each particle is move through the search space for search the optimal solution. When particles moving, all particles (i.e, N particles) have fitness values which are evaluated by fitness functions and the position of the best particle among all particles found so far is recorded and each particle keeps its personal best position which has passed previously. At each time of moving, each j^{th} element in the velocity vector $V_{id} = [v_{id,1}, v_{id,2}, \dots, v_{id,n}]$ and each element $x_{id,j}$ in the position vector $X_{id} = [x_{id,1}, x_{id,2}, \dots, x_{id,n}]$ of particle id are calculated as follows:

$$V_{id}^{k+1} = \omega^k * V_{id}^k + C_1 * Rand() * (P_{best\_id} - x_{id}^k) + C_2 * Rand() * (G_{best} - x_{id}^k) \quad (17)$$

$$x_{id}^{k+1} = x_{id}^k + x_{id}^{k+1} \quad (18)$$

$$\omega^k = \omega_{max} - \frac{k*(\omega_{max}-\omega_{min})}{iter\_max} \quad (19)$$

The P_{best_id} for j^{th} particle is presented as $P_{best\_id} = [p_{id,1}, p_{id,2}, \dots, p_{id,n}]$ and calculated as:

$$P_{best\_id}^{k+1} f(x) = \begin{cases} P_{best_{id}}^{k+1}, & if\ fitness(x_{id}^{k+1}) > P_{best_{id}}^k \\ fitness(x_{id}^{k+1}), & if\ fitness(x_{id}^{k+1}) \le P_{best_{id}}^k \end{cases} \quad (20)$$

The G_{best} at k^{th} iteration is computed as:

$$G_{best} = \min(P_{best_{id}}^k) \quad (21)$$

where V_{id}^k is the velocity of the particle id in k^{th} iteration, and is limited to $[-V_{max}, V_{max}]$, V_{max} is a constant pre-defined by user. X_{id}^k is the current position of a particle id in k^{th} iteration. The symbol ω denotes the inertial weight coefficient. The symbols C1 and C2 denote the self-confidence coefficient and the social confidence coefficient, respectively. In a standard PSO, the value of ω decreases linearly during the whole running procedure, and C1 = C2 = 2. The symbol Rand () denotes a function can generate a random real number between 0 and 1 under normal distribution. The symbols P_{best_id} denotes the personal best position of the particle id, respectively. The symbol Gbest denotes the best one of all personal best positions of all particles within the swarm. To be clearly visualized, Figure 2 depicts the trends of searching point of each particle based on by updating according to the personal best position of the particle and best positions of all particles within the swarm. The whole running procedure of the standard PSO is described in Algorithm 1.

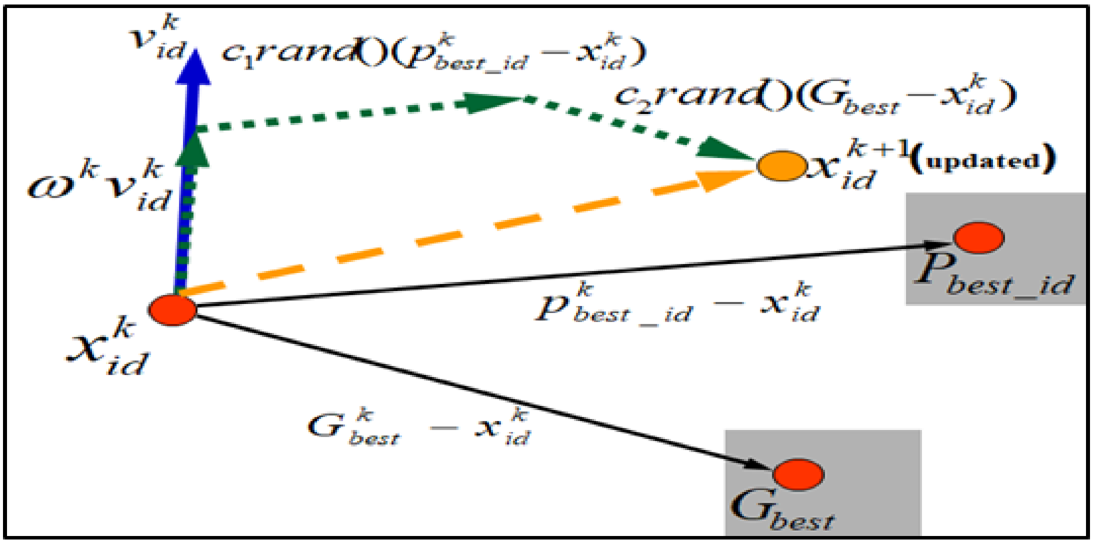

Figure 2: Depicting one searching point of each particle in PSO

Algorithm 1*: Standard PSO algorithm*

1. Initialize the N particles' positions X_{id} and velocities V_{id}
2. **While** the stop condition (the maximum moving step is reached) is not satisfied **do**
2.1**. For** each particle id in [1: N] **do**
✓ Calculate the fitness value of particle id
✓ if fitness value better than previous P_{best_id} then
▪ Set fitness value is new P_{best_id} according to (20)
end if
end **for**
2.2. Update the global best position of all particles G_{best} according to (21).
2.3. F**or** particle i, (1≤ id ≤ N) **do**
✓ Move particle id to another position according to (17) and (18)
end f**or**
✓ update ω according to (19)
end **while**

2.3. Normalization and denormalization

In this section, we refer to a common process during numerical computations, where one shrinks the allowed input to to semantic value domain of linguistic variables. This process are called Normalization from $(x, y) \rightarrow (x_s, y_s)$. The component Quantified Rule Base Composition & HA-IRMd has the task of computing the value of semantically quantifying mapping in the input and output linguistic variables for fuzzy model. Based on the rule system of the approximation inference problem, building the input – output relationship surface Sreal. When the inference approximation set works, with each set of input values (xs, ys), this component will find out interpolation value at the output. The component Denormalization will convert the value zs on the semantic value domain to the real value at the output.

Through component Quantified Rule Base Composition & HA-IRMd, the system will convert the value zs on the semantic value domain to the real value at the output and which is called Denormalization. The normalization and Denormalization process are shown in Figure 2.

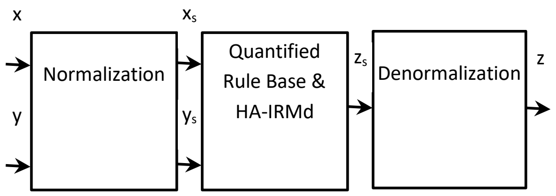

Figure 3. The diagram of approximation inference system

The range of any input - output linguistic variables to the approximation inference system is often any domain [a, b]. Semantic value in the hedge algebras is computed in the domain [0, 1]. Many problems using hedge algebras before used to use linear transformations from domain [a, b] to the domain [0, 1] and vice versa (Cao, Z., & Kandel A., 1989; Nguyen, C. H. *et al.*, 2014).The advantage of this linear transformation is simple and fast. In additions, another question is given whether the quality of the approximation inference would be better if we used the nonlinear transformations. For each specific approximation inference system, if we use nonlinear transformations, the optimal value can be easily achieved. The nonlinear transformations can use the polynomial order 2, 3 ... In this paper, we create nonlinearity the interval [a, b] and [0, 1] into three linear segments. Thus, there are four values on each variable domain:

[a, b]	a	y_1	y_2	b
[0, 1]	0	x_1	x_2	1

Using interpolation methods *Lagrange* on value set to perform the **normalization** and **denormalization**.

Designing the approximation inference system based on HA and PSO

As can be seen in the *Figure 2*, in order to make the approximation inference set give good results, designing an approximation inference based on the hedge algebras must consider computations of the components. In this paper, interpolation on the semantic relations input - output surface used is linear interpolation. The approximation inference set was built by 3 hedge algebras for 3 linguistic variables *Lx*, *Ly* and *Lz* corresponds to the variables x, y and z.

Select the parameter set and the rule system

First, determine the components of the hedge algebras for input - output variables. Hedge algebras for linguistic variables *Lx*, *Ly*, *Lz* of the variables *x*, *y*, *z* are:

1) The set of generating element G = {*N*, *P*}, with $c^{-}=N$ (*Negative*) and $c^{+} = P$ (*Positive*).

2) The set of hedges chosen: H^{-} = {*L* (*Little*)} and H^{+} = {*V* (*Very*)}.

The sign relationship of the hedges to the others and the generating elements are defined in the following sign table:

Table 1. The sign relationship of the hedges and the generating elements

	V	*L*	*N*	*P*
V	+	+	–	+
L	–	–	+	–

Rule base system is quantatively built on the variability of the bell-shaped function (1) (Pham Thanh Ha. 2010) in the following *Table 2*.

Table 2. Rule table for approximation inference Lz

Ly \ ***Lx***	***0***	***VN***	***LN***	***W***	***LP***	***VP***	***1***
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*
VN	*0*	*VVN*	*VN*	*LVN*	*VN*	*VVN*	*0*
LN	*0*	*VN*	*P*	*VP*	*P*	*VN*	*0*
W	*0*	*LVN*	*VP*	*1*	*VP*	*LVN*	*0*
LP	*0*	*VN*	*P*	*VP*	*P*	*VN*	*0*
VP	*0*	*VVN*	*VN*	*LVN*	*VN*	*VVN*	*0*
1	*0*	*0*	*0*	*0*	*0*	*0*	*0*

Rule table for approximation inference in *Table 2* can be clarified as follows:

If Lx = VN and Ly = VN then Lz = VVN
If Lx = VN and Ly = LN then Lz = VN
If Lx = LN and Ly = VN then Lz = VN
If Lx = LN and Ly = LN then Lz = P

Optimize the parameters using PSO

In this section, we apply PSO (Eberhart, R., & Kennedy, J. 1995) to optimize the fuzzy parameters of the hedge algebras and the nonlinear points in the variation domain of the language value and semantic value domain by minimizing objective function value. The objective function is calculated as follows:

$$MSE = \frac{1}{n\times n}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(z_{gt}(i,j) - z_{ha}(i,j)\right)^2 \quad (22)$$

To improve approximation accuracy of the proposed method, the effective of optimization techniques which are one of main issues presented in this paper. A novel method for approximating bell – shaped function is developed by PSO algorithm to find out the fuzzy parameters of the hedge algebras without increasing the number of rules.

Let the number of the variables in hedge algebras be m. Each particle id is a vector consisting of n elements p_k where 1≤ k ≤ m . Based on these m-1 variables, define the m optimal parameters. In this work, we use three variables which to determine the output variables of the hedge algebras based on PSO algorithm. The optimal parameters are received as in *Table 3*:

Table 3: The optimal fuzzy parameters

	L_x, L_y	L_z
fm(N)	0.5	0.371479
$\alpha = \mu(L)$	0.450874	0.342747

With the optimal parameter set found by PSO, we can compute the semantically quantifying value of linguistic terms based on applying the formula (13) to (16) in rule system as shown in *Table 4*.

Table 4: Semantically quantifying value ofrule system

L_{ys} \ L_{xs}	**0**	**0.1508**	**0.3762**	**0.5000**	**0.6238**	**0.8492**	**1**
0	0	0	0	0	0	0	0
0.1508	0	0.1055	0.1605	0.1892	0.1605	0.1055	0
0.3762	0	0.1605	0.5869	0.7285	0.5869	0.1605	0
0.5000	0	0.1892	0.7285	1.0000	0.7285	0.1892	0
0.6238	0	0.1605	0.5869	0.7285	0.5869	0.1605	0
0.8492	0	0.1055	0.1605	0.1892	0.1605	0.1055	0
1	0	0	0	0	0	0	0

Nonlinear conversion curve using interpolation method *Lagrange* to the *normalization* and *denormalization* as shown in the *Figuge 4.*

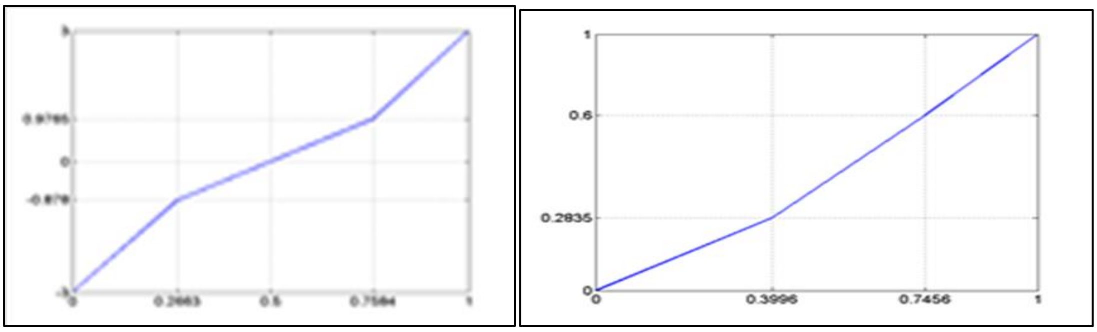

Figure 4. Normalizationcurve and denormalization

Computing results

Suppose *x*, *y* varies in the range [-3, 3], with *step* = 0.1, the total of models is $n \times n$ = 61×61 = 3721. The approximation inference set computed according to the hedge algebras gives the approximation curved surface as shown in the *Figure 5.*

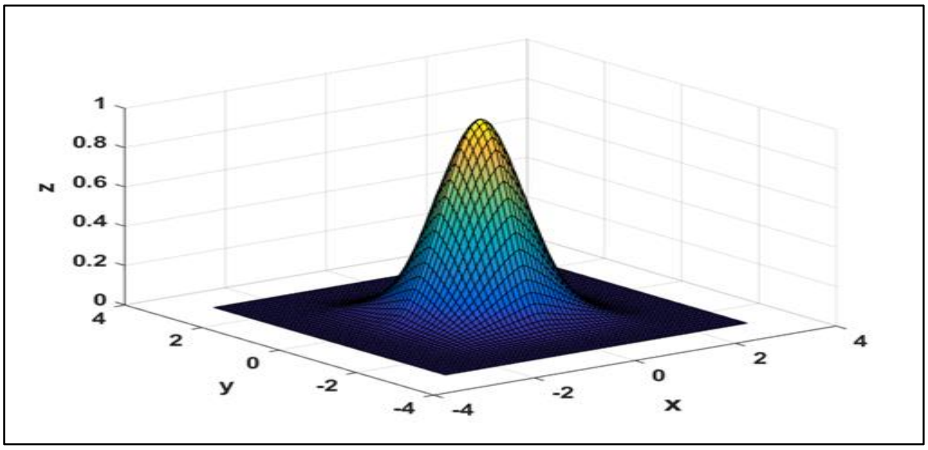

***Figure 5.* Approximation bell-shaped**

According to Eq.(22), the computing error is: MSE = $3.2134*10^{-5}$.

CONCLUSIONS

In this paper, we propose a method to improve the accuracy of the approximation function using hedge algebras by executing the normalization and denormalization by nonlinear interpolation. We can draw out the computing result of approximation inference set to the optimal parameters found by PSO as follows:

- Hedge algebras has the preeminent because of its simple computing method, it doesn't take a lot of computing time to calculate one function value when the fuzzy inferring is minor. Furthermore, this method brings a more accurate result than the one in (Pham Thanh Ha. 2010; Satish, K. 1999).
- After having built the inference set, it is easier to optimize the parameters. The accuracy of inference just depends on the fuzzy parameters of hedge algebras, *normalization*, *denormalization* and the interpolation on the input - output relationship.
- To optimize the fuzzy parameters of the hedge algebras, we can easily apply the optimal and efficient algorithms such as GA, ACO, SA, ...
- It is completely possible to widen the application of hedge algebras in order to solve the problems of approximation inference based on the rules. Due to the statement of rule system, the semantic order of the language must be assured.
- Apply the efficient and optimal methods to optimize the parameters of hedge algebras in the fuzzy.

Acknowledge

This paper was implemented by the "Knowledge technologies and soft computing – TNUT" research group.

REFERENCES

1. Cao, Z., & Kandel A., (1989). "Applicability of some fuzzy implication operators", *Fuzzy Sets and Systems* 31, 151-186.
2. Nguyen, C. H., Vu, N. L., Nguyen, T. D., & Van, T. P. (2014). Study the ability of replacing fuzzy and PI controllers with the Hedge-Algebras-Based controller for DC motor. *Journal of science and technology*, *52*(1), 35-48.
3. Ho N. C., & Wechler, W. (1990). "Hedge algebra: An algebraic approach to structures of sets of linguistic truth values", *Fuzzy Sets and Systems* 35, pp. 281–293.
4. Cat, H. N., Nhu, L. V., & Tien, D. N. (2014). Hedge-Algebra-Based Voltage Controller for a Self-Excited Induction Generator. In *The 7th National Conference on Fundamental and Applied IT Research*.
5. Pham Thanh Ha. (2010). Development of fuzzy reasoning methods using Hedge Algebras and applications, Institute of Information Technology, Vietnam Academy of Science and Technology.
6. Satish, K. (1999). "Managing Uncertainty in the Real World – Part Fuzzy Sets", Resonance, 4 (2), pp.37 – 47.
7. Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. *In Proceedings of the 6th international symposium science*, pp. 39–43.
8. Nhu Lan Vu, & Tien Duy Nguyen. (2007). "Identification of the rule – based systems using Hedge Algebras", *Journal of science and technology*, 45, (1), 1-6.