in Proceedings of EUFIT'97

Naturally Defined Membership Functions for Fuzzy Logic Systems and A Comparison with Conventional Set Functions

Jianwei Zhang, Khac Van Le and Alois Knoll
Faculty of Technology, University of Bielefeld, 33501 Bielefeld, Germany
Phone: ++49-(0)521-106-2951, Fax: ++49-(0)521-106-2962
Email: {zhang, ilkhacva, knoll}@techfak.uni-bielefeld.de

Abstract: We point out that B-spline basis functions are naturally defined membership functions for fuzzy logic systems, i.e. the specification of these functions depends only on the partition points of each linguistic variables, no more necessarily also on the additional parameters if normal set functions are used. Based on B-spline basis functions, a fuzzy controller can be constructed, which works like an adaptive B-spline interpolator. Through comparative examples for function approximation, we show that learning of such a fuzzy controller generally converges faster. The approximation errors are in most cases not larger than the results achieved by using the normal set functions, in some cases even smaller, depending on the type of test functions. The good approximation ability and the fast convergence of learning make this model suitable to supervised and unsupervised learning for a wide range of modelling and control tasks.

1 Introduction

After it is theoretically proven that fuzzy controllers are general function approximators, some constructing methods are reported. Typical systems are ANFIS (Adaptive-Network-based Fuzzy Inference System) [Jan93], NEFCON (NEural Fuzzy CONtroller) [NKK94] and SAM (Standard Additive Model) [MK96]. To contruct a fuzzy controller for such a task, a basic problem is the selection of membership functions (MFs) for modelling linguistic terms.

Recently, *Non-Uniform B-Splines* (NUBS) model is compared with a fuzzy controller, [BH94, ZK96]. In this paper, we also follow the usage of this type of NUBS basis functions (*B-functions* for short). In our previous work, we compared the fuzzy controlllers based on B-functions with ANFIS based TSK model and showed some advantages of our system. In this work, we compare B-functions with a series of parameterised set functions selected by [MK96]. These set functions were used by SAM, which is close to the classical controller type, to approximate non-linear functions.

2 B-Spline Basis Functions as Fuzzy Sets

2.1 Definition

Assume x is a general input variable of a control system which is defined on the universe of discourse $[x_0, x_m]$. Given a sequence of ordered parameters (knots): $(x_0, x_1, x_2, \ldots, x_m)$, the $i-$th normalised B-spline basis function (B-function) $X_{i,k}$ of order k is defined as:

$$X_{i,k}(x) = \begin{cases} \begin{cases} 1 & \text{for } x_i \leq x < x_{i+1} \\ 0 & \text{otherwise} \end{cases} & \text{if } k = 1 \\ \frac{x - x_i}{x_{i+k-1} - x_i} X_{i,k-1}(x) + \frac{x_{i+k} - x}{x_{i+k} - x_{i+1}} X_{i+1,k-1}(x) & \text{if } k > 1 \end{cases} \quad \text{with } i = 0, 1, \ldots, m-k.$$

2.2 Peak Support Points and Knots

In fuzzy set theory, the *support* of a fuzzy set A within a universal set X is the crisp set that contains all the elements of X that have non-zero membership grades in A. If a B-function of order k $(k > 1)$ is used for modelling a fuzzy set, it possesses only one peak which has the largest membership grade. If the order $k = 2$ or the knots are evenly distributed, the support point of this peak, denoted as the *Psupp*-point (*peak support point*), can be defined as $Psupp(A) = \{x | A(x) = maximum\}$, where A is defined by a B-function.

A B-function representing A_i is defined by the *knots*, the boundary points of the *support* of A_i. The complete *knots* consist of two parts, the *interior knots* (noted as *Iknots*) which lie within the universe of discourse and *extended knots* (*Eknots*) which are generated at both ends of the universe for defining the *virtual linguistic terms*. Generally, $m - (k \ mod \ 2)$ interior knots are needed, where m is the number of the real linguistic terms, and k is the order of the B-functions ($k \leq m$).

If k is even, the interior knots coincide with the *Psupp*-points. If k is odd, the $m - 1$ interior knots can be determined by the *Psupp*-points, Fig. 1.

2.3 A B-Spline Interpolator

Since a MIMO (Multiple-Input-Multiple-Output) rule base is normally divided into several MISO (Multiple-Input-Single-Output) rule bases, we consider only the MISO case. Under the following conditions:

- periodical B-spline basis functions as membership functions for inputs;
- fuzzy singletons as membership functions for outputs (control vertices);
- "product" as fuzzy conjunctions;
- "centroid" as defuzzification method;
- addition of "virtual linguistic terms" at both ends of each input variable and
- extension of the rule base for the "virtual linguistic terms" by copying the output values of the "nearest" neighbourhood;

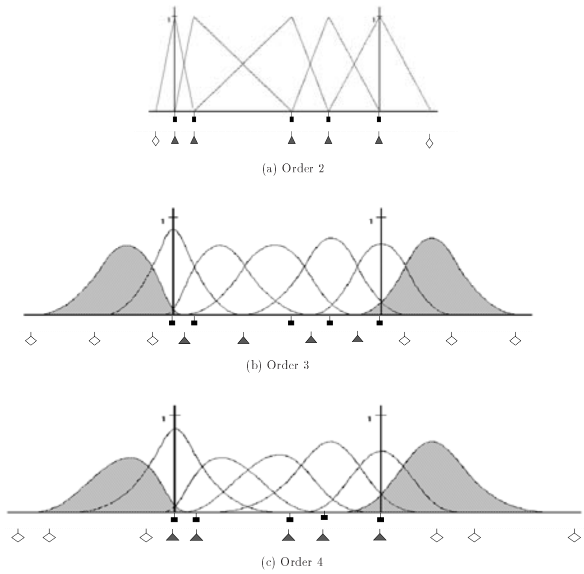


Figure 1: Non-uniform B-functions of different orders defined for *real* and *virtual* linguistic terms by the same knot vector (*Peak support points*: □; *Iknots*: △; *Eknots*: ◇; *virtual linguistic terms*: shaded).

the computation of the output of such a fuzzy controller is equivalent to that of a *general B-spline hypersurface*. Generally, we consider a MISO system with n inputs $x_1, x_2, \ldots, x_n$. A $Rule(i_1, i_2, \ldots, i_n)$ with the n conjunctive terms in the IF-part is given in the following form:

IF (x_1 is $X^1_{i_1,k_1}$) and (x_2 is $X^2_{i_2,k_2}$) and ... and (x_n is $X^n_{i_n,k_n}$) THEN y is $Y_{i_1 i_2 \ldots i_n}$,

where x_j is the j-th input ($j = 1, \ldots, n$), k_j is the order of the B-spline basis functions used for x_j, $X^j_{i_j,k_j}$ is the i-th linguistic term of x_j defined by B-spline basis functions, $i_j = 1, \ldots, m_j$ represents how fine the j-th input is fuzzy partitioned, $Y_{i_1 i_2 \ldots i_n}$ is the control vertex (deBoor points) of $Rule(i_1, i_2, \ldots, i_n)$.

Then, the output y of a MISO fuzzy controller is:

$$y = \frac{\sum_{i_1=1}^{m_1} \cdots \sum_{i_n=1}^{m_n} (Y_{i_1,\ldots,i_n} \prod_{j=1}^{n} X^j_{i_j,k_j}(x_j))}{\sum_{i_1=1}^{m_1} \cdots \sum_{i_n=1}^{m_n} \prod_{j=1}^{n} X^j_{i_j,k_j}(x_j)} = \sum_{i_1=1}^{m_1} \cdots \sum_{i_n=1}^{m_n} (Y_{i_1,\ldots,i_n} \prod_{j=1}^{n} X^j_{i_j,k_j}(x_j)) \quad (1)$$

2.4 Adaptation of Control Vertices and Knots

Assume $\{(\mathbf{X}, y_d)\}$ is a set of training data, the squared error is computed as $E = \frac{1}{2}(y_r - y_d)^2$, Each control vertex $Y_{i_1,\ldots,i_n}$ can be modified by using the gradient descent method:

$$\Delta Y_{i_1,\ldots,i_n} = -\epsilon \frac{\partial E}{\partial Y_{i_1,\ldots,i_n}} = \epsilon (y_r - y_d) \prod_{j=1}^{n} X^j_{i_j,k_j}(x_j) \quad (2)$$

The gradient descent method guarantees that the learning algorithm converges to the global minimum of the error function because the second partial differentiation with respect to $Y_{i_1,i_2,\ldots,i_n}$ is always positive. This means that the error function is convex in the space $Y_{i_1,i_2,\ldots,i_n}$ and therefore possesses only one (global) minimum.

We also developed an algorithm for adapting the *knots*. This algorithm is a modified algorithm for self-organising neural networks, [ZLK97].

3 Comparative Results for Function Approximation

3.1 Test Functions

We tested our approach with the functions used in [MK96]:

One-dimensional functions:

$$
\begin{aligned}
f_1(x) &= 3x(x-1)(x-1.9)(x-0.7)(x+18), \quad \text{for } -2 \le x \le 2 \\
f_2(x) &= 10\tan^{-1}\left(\frac{(x-0.2)(x-0.7)(x+0.8)}{(x+1.4)}\right), \quad \text{for } -1 \le x \le 1 \\
f_3(x) &= \frac{100(x+0.95)(x+0.6)(x+0.4)(x-0.1)(x-0.4)(x-0.8)(x-0.9)}{(x+1.7)(x-2)^2}, \quad \text{for } -1 \le x \le 1 \\
f_4(x) &= 8\sin(10x^2+5x+1), \quad \text{for } -1 \le x \le 1 \\
f_5(x) &= 10\tan^{-1}\left(\frac{(x-0.2)(x-0.7)(x+0.8)}{(x+1.4)(x-1.1)x+0.7}\right), \quad \text{for } -1 \le x \le 1 \\
f_6(x) &= 10\left(e^{-5|x|}+e^{-3|x-0.8|/10}+e^{-10|x+0.6|}\right), \quad \text{for } -1 \le x \le 1
\end{aligned}
$$

Two-dimensional functions:

$$
\begin{aligned}
g_1(x,y) &= f_2(x) \times 10\left(\sin(4y+0.1)+\sin(11y-0.2)+\sin(14y)+\sin(17y+0.3)\right), \quad \text{for } -1 \le x,y \le 1 \\
g_2(x,y) &= f_4(x) \times 2\left(e^{-\left(\frac{y-0.1}{0.25}\right)^2} - 0.8e^{-\left(\frac{y+0.75}{0.15}\right)^2} - 0.4e^{-\left(\frac{y-0.8}{0.1}\right)^2}\right), \quad \text{for } -1 \le x,y \le 1 \\
g_3(x,y) &= f_1(x) \times \sin(y), \quad \text{for } -2 \le x,y \le 2
\end{aligned}
$$

Three-dimensional functions:

$$
\begin{aligned}
h_1(x,y,z) &= 0.1\left(e^{-\frac{|x|}{0.2}}+e^{-\frac{|x-0.8|}{0.3}}+e^{-\frac{|x+0.6|}{0.1}}\right) \times \left(\tan^3(1.5y)+10\tan^2(y)-20\tan(0.7y)\right) \times \\
&\quad \left(\arccos^3(z)-\arccos^2(-z)-\arccos(-z)\right), \quad \text{for } -1 \le x,y,z \le 1 \\
h_2(x,y,z) &= e^{-(xy-0.7)(xz-0.5)} \times \frac{\sin\left(\frac{125}{(x+1.5)}\right)}{y+1.1} \times \frac{(5xy^3-6z^3)}{(xyz+2)}\tan^{-1}(10xy+z^3), \quad \text{for } -1 \le x,y,z \le 1 \\
h_3(x,y,z) &= 1000(x+0.95)(x+0.6)(x+0.4) \times (x-0.1)(x-0.4)(x-0.8)(x-0.9)(y+0.7) \times \\
&\quad (y-0.35)(y-0.9)(z+0.7)(z+0.2)(z-0.4) \times (yz+0.6)(x+yz), \quad \text{for } -1 \le x,y,z \le 1
\end{aligned}
$$

3.2 Implementation with B-functions

The simulation results achieved by [MK96] are summarised in Table 1. The same test functions were approximated with the fuzzy controller based on the model proposed in section 2. We first used the same number of linguistic terms as in [MK96], then several more. Curves of the Mean-Squared Error are plotted in Fig. 2.

4 Conclusions

- B-spline basis functions are one of the most suitable parameterised set functions for modelling fuzzy sets of a controller. B-functions are naturally defined fuzzy sets, i.e. no additional parameters are needed

Function	Rules	Membership Functions in the *Standard Additive Model*						
		Triangle	Gaussian	Cauchy	Sinc	Laplace	Hyperbolic Tangent	Logistic
f_1	12	0.4	0.2	0.2	0.18	0.28	0.08	0.7
f_2	12	0.3	0.02	0.05	0.07	0.07	0.09	0.1
f_3	12	0.03	0.008	0.005	0.01	0.002	0.005	0.03
f_4	12	6	6	2	0.1	6	6	10
f_5	12	0.2	0.05	0.9	0.01	0.1	0.02	1
f_6	12	0.1	0.2	0.2	0.1	0.2	0.1	0.4
g_1	8×8	23	12	12	9	18	10	26
g_2	8×8	18	14	14	7	17	12	19
g_3	8×8	3.8	2	3	1.2	3	2	6
h_1	$5 \times 5 \times 5$	9	7	6	4	5	7	8
h_2	$5 \times 5 \times 5$	18	15	15	15	15	16	15
h_3	$5 \times 5 \times 5$	3	1	0.9	1	0.9	1	0.05

Table 1: Summary of the Results of [MK96]. The contents of the table indicate the Mean-Sqaured Erorr after sufficient epochs of learning.

apart from the partition parameters (knots) of an input. This feature enables that a user only needs to concentrate on the initial partition of the input space using *a priori* knowledge if it is available.

- Although it is natural that using more B-functions in the IF-part increases the approximation precision, it is an important property that the price for that is merely more external memories to store the knots and control vertices. The computation cost (k^n instead of m^n) stays the same regardless how many more B-functions are used.

The real meaning of this experiment is not only to test the ability for function approximation, but to test how fast and how flexible a fuzzy controller can be automatically constructed for shaping any control surfaces. Fuzzy controllers based on the B-spline model possess not only the intrinsic advantages of B-splines, like smoothness, intuitive geometric interpretation of parameters and control vertices, but also the following desired properties for adaptive modelling and control:

- By applying the algorithm in [ZLK97], the knots for defining the B-functions can be automatically adjusted, which results in precise positioning and efficient utilisation of linguistic terms.
- Based on the Squared-Error goal function and Gradient Descent method, the learning of control vertices converges very fast (much faster than SAM with any set function used in [MK96]) since the modification of control vertices only influences the local area of control surface.
- The efficient evaluation of rule basis and simple gradient descent computation enable on-line learning. We have applied the approach in on-line learning of robot motions both for mobile robots and manipulators, [ZvCK97].
- By selecting an appropriate cost function, this learning method can be generalised for unsupervised learning, [ZLK].

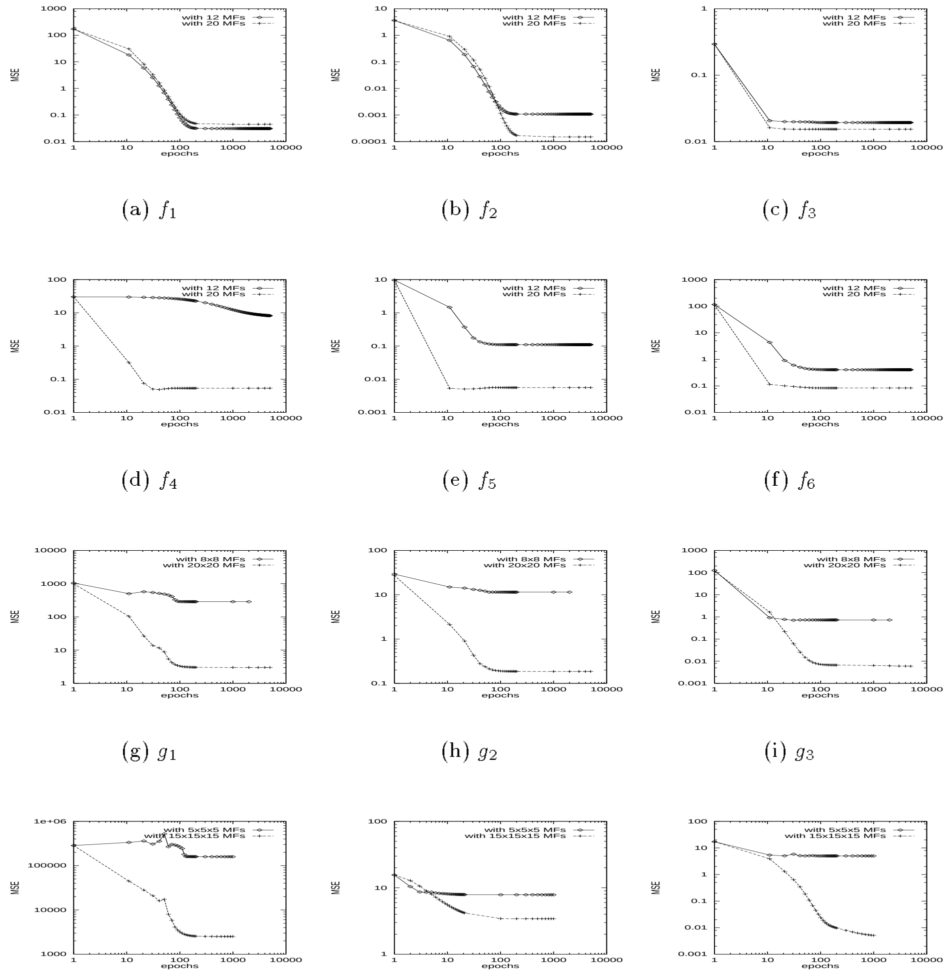
References

[BH94] M. Brown and C. J. Harris. *Neurofuzzy Adaptive Modelling and Control.* Prentice Hall, 1994.

[Jan93] J.-S. R. Jang. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on System, Man and Cybernetics*, 23(3):665–685, 1993.

[MK96] S. Mitaim and B. Kosko. What is the best shape of a fuzzy set in function approximation. In *IEEE International Conference on Fuzzy Systems*, 1996.

[NKK94] D. Nauck, F. Klawonn, and R. Kruse. *Neuronale Netze und Fuzzy-Systeme.* Vieweg, 1994.



(a) f_1 (b) f_2 (c) f_3

(d) f_4 (e) f_5 (f) f_6

(g) g_1 (h) g_2 (i) g_3

(j) h_1 (k) h_2 (l) h_3

Figure 2: Mean-squared error of approximating functions of f_1 to f_6 (one-dimensional), g_1 to g_3 (two-dimensional) and h_1 to h_3 (three-dimensional). Each function was approximated with as many MFs as used in [MK96] and with several more MFs. In both cases, the computation times for evaluation of the rule base are the same.

[ZK96] J. Zhang and A. Knoll. Constructing fuzzy controllers with B-spline models. In *IEEE International Conference on Fuzzy Systems*, 1996.

[ZLK] J. Zhang, K. V. Le, and A. Knoll. Unsupervised learning of control spaces based on B-spline models. To appear in *Proceedings of IEEE International Conference on Fuzzy Systems*, 1997.

[ZLK97] J. Zhang, V. K. Le, and A. Knoll. A rapid learning approach for developing fuzzy controllers. In *4. International Workshop in Fuzzy-Neuro-Systems, Soest*, 1997.

[ZvCK97] J. Zhang, Yorck v. Collani, and A. Knoll. On-line learning of sensor-based control for acquiring assembly skills. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1997.