In Proceedings of the International Workshop on Fuzzy-Neuro Systems, Leipzig, 1999

Genetic Adaptation to Optimal Membership Functions for Modelling with B-Splines

J. Zhang, I. Renners und A. Knoll
AG Technical Computer Science, Faculty of Technology,
University of Bielefeld, D-33501 Bielefeld, Germany
Tel.: ++49 521 106-2951, Fax: ++49 521 106-6440
E-mail: zhang|irenners|knoll@techfak.uni-bielefeld.de

Abstract

This paper proposes a comprehensive approach on automatic optimisation of B-spline based neuro-fuzzy models. The underlying structure of a system decided by the displacements of the membership functions is an important factor for its generalisation ability and output accuracy. Therefore a genetic algorithm is introduced which is used to find a suitable model for an arbitrary given problem. Approximation of benchmark test functions shows that our approach beats the models using other set functions.

Keywords: Neuro-Fuzzy Modelling, B-Spline, Genetic Algorithm (GA), Generalisation.

1 Introduction

Fuzzy controller is an important model for abstraction complex data thanks to its interpretability and function approximating capability. However, the choice of MF affects how precise the fuzzy controllers are able to approximate functions. In [6] the common MFs like triangles, trapezoids and other set functions were employed and compared. In our previous work [7] we showed the advantages of using B-splines as MF. We compared splines and a fuzzy controller with SISO (*single-input-single-output*) and MISO (*multi-input-single-output*) structures.

1.1 B-Spline Networks

B-Spline Networks (BSNs) make use of piecewise polynomials as membership functions (MFs). Each interval of interest (in example each input dimension) is divided into a number of subintervals, where each subinterval is delimited by breakpoints called *knots* which determine the appearance and position of each B-spline (Fig.1).
Figure 2 illustrates a BSN with 8 MFs on each uniformly subdivided input interval and the activated B-splines (lightly shaded) for a given input. Since learning one new part of the input space affects only a given number of network response values (see darkly shaded area of Fig. 2),

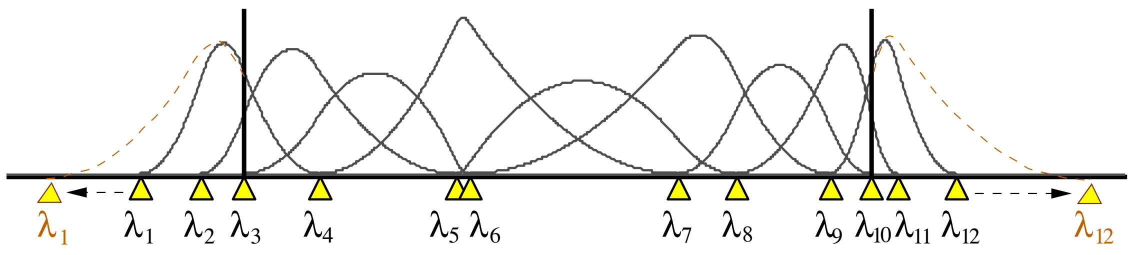


Figure 1: Nine B-splines of order 3 defined over 12 non-uniformly distributed knots $(\lambda_1, \ldots, \lambda_{12})$.

fast on-line learning can be derived. On the basis of these advantages, B-spline networks (BSNs) are often used in control systems and will be denoted as B-Spline Fuzzy Controllers (BSFCs). By using BSNs the approximation ability is only limited by the number of knots distributed on each input interval. Regarding that most observed data are disturbed to a certain degree the problem of overfitting can occur. GA optimised BSNs are a promising approach to find sparse models, which are able to bridge the gap between high bias and high variance of a model.

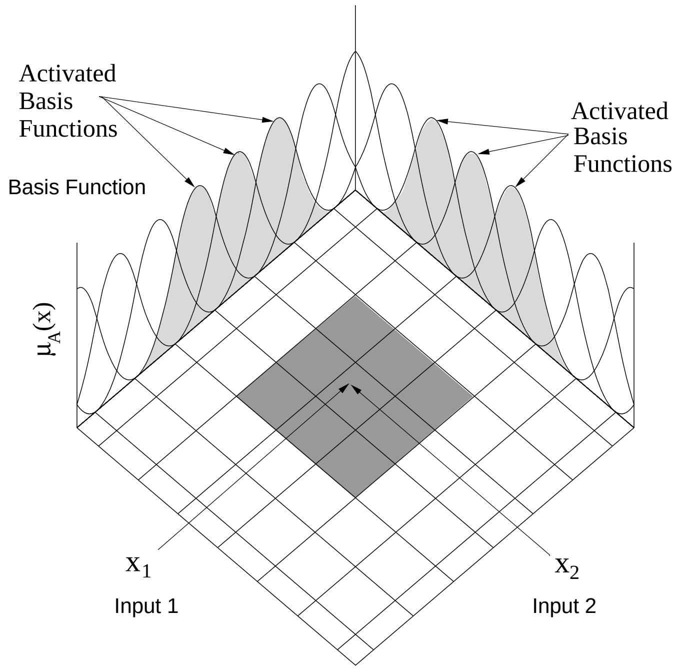


Figure 2: Two-Dimensional B-spline Network with uniform knot distribution.

1.2 Definition of B-Splines

The B-spline $N_{i,k}$ of order k with knots $\lambda_1, \ldots, \lambda_{k+1}$ (Fig. 3) is defined as:

$$N_{i,k}(x) = (\lambda_{i+k} - \lambda_i) \sum_{p=1}^{k+1} \frac{(\lambda_{i+p} - x)_+^k}{\prod_{\binom{h=1}{h \neq p}}^{k} (\lambda_{i+p} - \lambda_{i+h})}. \quad (1)$$

Therefore m knots $\lambda_i (i = 1, \ldots, m)$ form $l = m - k$ B-splines (Fig.1). Using this l B-splines as linguistic terms, the minimum input value of a BSN is determined by $a = \lambda_k$ and the maximum input value by $b = \lambda_{m-k+1}$, constituting the valid input interval to $[a, b]$. Thus the required parameters to define the structure of a normal B-spline model are:

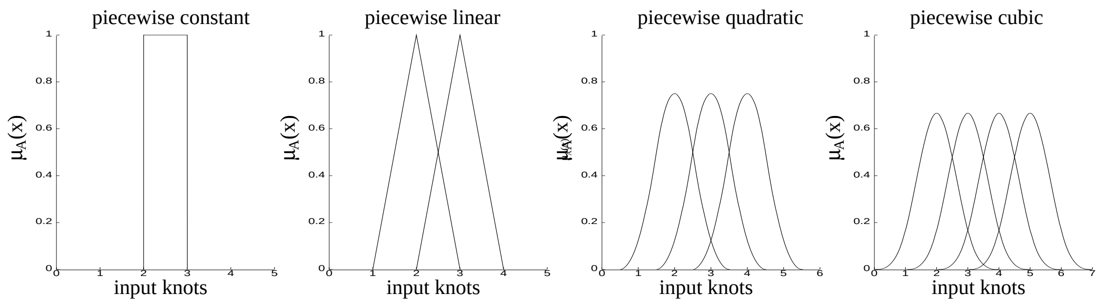
$$Parameter(BSN) = \sum_{j=1}^{n}(m - 2 + 1) = \sum_{j=1}^{n}(m - 1) \tag{2}$$

The Term "-2" is needed since the two outermost knots does not influence the valid input interval $[a_j, b_j]$ and the term "$+1$" is used because this parameter contains k_j.

1.3 Basic Properties of B-Splines

The most important properties of B-splines, with respect to neuro-fuzzy modelling are:

- Recursion: $$N_{i,p+1}(x) = \frac{x-\lambda_i}{\lambda_{i+p-1}-\lambda_i} N_{i,p}(x) + \frac{\lambda_{i+p}-x}{\lambda_{i+p}-\lambda_{i+1}} N_{i+1,p}(x)$$ $$N_{i,1}(x) = \begin{cases} 1, & \text{if } x \in [\lambda_i, \lambda_{i+1}) \\ 0, & \text{otherwise} \end{cases}$$
- Positivity: $N_{i,k} \geq 0$ for all x
- Local support: $N_{i,k} = 0$ if $x \notin [\lambda_i, \lambda_{i+k}]$
- Partition of unity: $\sum_{i=1}^{l} N_{i,k}(x) = 1$



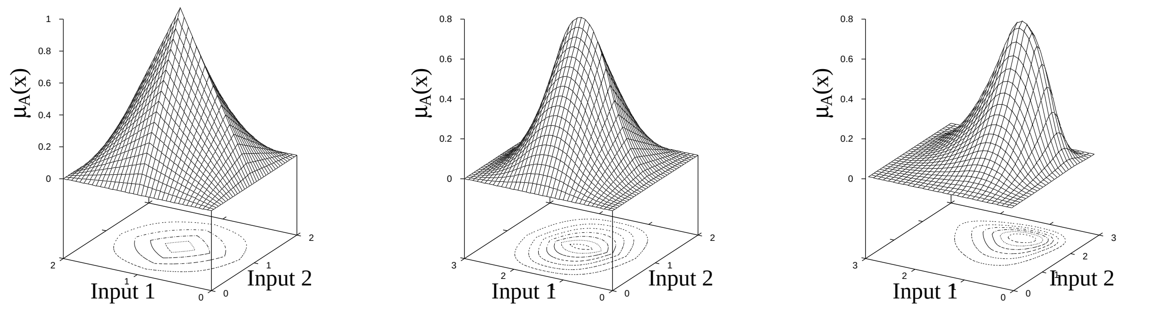
(a) Order 1 (b) Order 2 (c) Order 3 (d) Order 4

Figure 3: Univariate B-splines (UBs) of order 1 – 4.

Each n-dimensional rectangle $(n > 1)$ of the lattice is covered by the j^{th} multivariate B-spline $N_k^j(x)$ which is formed by taking the tensor product of n univariate B-splines:

$$N_k^j(x) = \prod_{j=1}^{n} N_{i_j,k_j}^j(x_j) \tag{3}$$

Therefore the shape of each B-spline, and thus the shape of multivariate ones (Fig. 4), is implicitly set by their order and their given knot distribution on each input interval.



(a) Formed by two, order 2 UBs.
(b) Formed by one order 3 and one order 2 UB.
(c) Formed by two UBs of order 3.

Figure 4: Bivariate B-splines formed by taking the tensor product of two univariate B-splines.

1.4 Network Output

The output of a Single Input Single Output (SISO) B-spline network is simply the unique representation of a B-spline $s(x),\ x \in [a, b]$:

$$y = \sum_{i=1}^{l} c_i N_{i,k}(x) \tag{4}$$

in which c_i are called *B-spline coefficients of* $s(x)$ (also denoted as *weights*, *control points* or *de Boor points*) and l denotes the number of B-splines.
The output of a Multiple Input Single Output (MISO) B-spline network can be computed straightforward by merging (3) and (4):

$$y = \sum_{i_1=1}^{l_1} \cdots \sum_{i_n=1}^{l_n} \left(c_{i_1 i_2 \ldots i_n} \prod_{j=1}^{n} N^j_{i_j,k_j}(x_j) \right) \tag{5}$$

where:

- x_j: the j^{th} input $(j = 1, \ldots, n)$,
- k_j: the order of the B-splines used for x_j,
- $N^j_{i_j,k_j}$: the i^{th} linguistic term of x_j defined by B-splines,
- $i_j = 1, \ldots, l_j$: represents how fine the j^{th} input is partitioned,
- $c_{i_1,i_2,\ldots,i_n}$: the coefficients of Rule $(i_1, i_2, \ldots, i_n)$.

2 Determining Optimal Coefficients

Determining optimal coefficients of a B-spline model with fixed knot vector is generally more simple than finding the optimal knot vectors. Applying methods of B-spline theory in CAGD

(Computer Aided Graphic Design), the coefficients can be estimated by solving an overdetermined linear system (section 2.1). Another method based on gradient descent can also be used (section 2.2). Although an iterative solution, this learning method is conceptually more easily understandable. Based on this learning method, unsupervised learning procedure of B-spline coefficients can also derived.

2.1 Coefficient Estimation by Solving the Overdetermined Linear System

2.1.1 One Dimensional Case

Assume that (x_r, y_r) is a set of equally weighted training data, with $r = 1, \ldots, q$ and $a \leq x_r < x_{r+1} \leq b$. To determine a spline $s(x)$ with coefficients c_i on $[a, b]$, of order k, with given knots $\lambda_i, i = 1, \ldots, m$, we substitute (4) in the well-known least-square criterion:

$$\delta(c) := \sum_{r=1}^{q} (y_r - s(x_r))^2 \tag{6}$$

yields:

$$\delta(c) = \sum_{r=1}^{q} \left(y_r - \sum_{i=1}^{l} c_i N_{i,k}(x_r) \right)^2 \tag{7}$$

The searched B-spline coefficients can then be computed as the least-square solution of an overdetermined linear system, i.e.

$$\sum_{i=1}^{l} c_i N_{i,k}(x_r) = y_r \quad , r = 1, \ldots, q, \tag{8}$$

which can be written in matrix notation as:

$$\delta(c) = (y - Ec)^T (y - Ec) = ||y - Ec||^2, \tag{9}$$

with $|| \cdot ||$ denoting the standard Euclidean vector norm, where

$$E = \begin{bmatrix} N_{1,k}(x_1) & \cdots & N_{l,k}(x_1) \\ \vdots & & \vdots \\ N_{1,k}(x_q) & \cdots & N_{l,k}(x_q) \end{bmatrix}$$

is the observation matrix and

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_q \end{bmatrix}, c = \begin{bmatrix} c_1 \\ \vdots \\ c_l \end{bmatrix}.$$

Finding the B-spline coefficients as a solution of the overdetermined linear system

$$Ec = y \tag{10}$$

solves the problem in the least-square sense. The solution of (10) can be found by solving the *normal equations*:

$$Ac = r \tag{11}$$

where:

$$A = E^T E = \begin{bmatrix} < N_1, N_1 > & \cdots & < N_1, N_l > \\ < N_2, N_1 > & \cdots & < N_2, N_l > \\ \vdots & & \vdots \\ < N_l, N_1 > & \cdots & < N_l, N_l > \end{bmatrix}$$

is the autocorrelation matrix and

$$r = E^T y = \begin{bmatrix} < N_{1,y} > \\ < N_{2,y} > \\ \vdots \\ < N_{l,y} > \end{bmatrix}$$

is the cross-correlation matrix, with:

$$\begin{aligned} < N_j, N_i > &= \sum_{r=1}^{q} N_{j,k}(x_r) N_{i,k}, \\ < N_j, y > &= \sum_{r=1}^{q} N_{j,k}(x_r). \end{aligned} \tag{12}$$

The matrix A has two useful properties:

- A is symmetric and positively definite ($x^T A x > 0$ for all $x \neq 0$).
- A is a sparse matrix (many zero elements) with band structure of bandwidth k.

To solve the system efficiently we used the Cholesky method for positive band matrices [1].

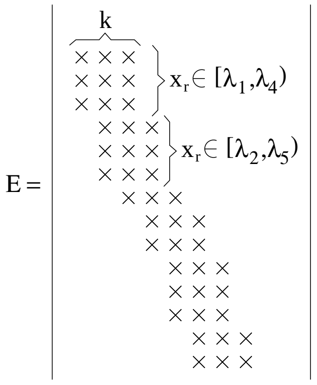
2.1.2 High-Dimensional Case

Assume that $(x_1^r, \ldots, x_n^r, y^r)$ is a set of equally weighted training data, with $r = 1, \ldots, q$, and $(x_1^r, \ldots, x_n^r, y^r) \in \mathbb{R} = [a_1, b_1] \times \cdots \times [a_n, b_n]$. To determine a tensor product spline $s(x_1^r, \ldots, x_n^r)$ with coefficients c_{i_j}, order k_j, and given knots $\lambda_{j_i}, i = 1, \ldots, m_j$ (at each case $j = 1, \ldots, n$) we substitute (5) in (6) yielding:

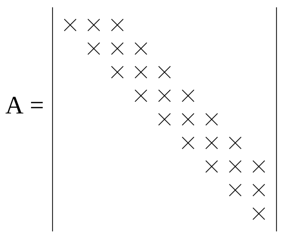
$$\delta(c) = \sum_{r=1}^{q} \left(y^r - \sum_{i_1=1}^{l_1} \cdots \sum_{i_n=1}^{l_n} \left(c_{i_1 i_2 \ldots i_n} \prod_{j=1}^{n} N_{i_j, k_j}^{j}(x_j^r) \right) \right)^2 \tag{13}$$

The searched B-spline coefficients can then again be computed as the least-squares solution of an overdetermined linear system, i.e.

$$\sum_{i_1=1}^{l_1} \cdots \sum_{i_n=1}^{l_n} \left(c_{i_1 i_2 \ldots i_n} \prod_{j=1}^{n} N_{i_j, k_j}^{j}(x_j^r) \right) = y^r \ \ , r = 1, \ldots, q, \tag{14}$$



(a) Observation matrix E.



(b) Autocorrelation matrix $A = E^T E$.

Figure 5: The observation matrix E and the corresponding autocorrelation matrix $A = E^T E$.

which can again be written in matrix notation as:

$$Ec = y, \tag{15}$$

where E is the $q \times (l_1) \cdots (l_j)$ obervation matrix with as elements $\prod_{j=1}^{n} N_{i_j,k_j}^{j}(x_j)$. When arranging the data points according to the n-dimensional rectangle containing them, the matrix E builds up a band structure and thus (11) can be solved similarly by using the Cholesky method.

2.2 Coefficient Estimation by Gradient Descent

Assume that $(x_1^r, \ldots, x_n^r, y_{\text{desired}}^r)$ is a set of training data. The output value computed by a BSN is denoted with $y_{computed}^r$. By defining the Mean-Square Error (MSE) criterion as:

$$E = \frac{1}{2} \cdot (y_{computed} - y_{\text{desired}})^2 \equiv \text{MIN}, \tag{16}$$

the derivation of each coefficient $c_{i_1,\ldots,i_k}$ is:

$$\Delta c_{i_1,\ldots,i_k} = -\epsilon \frac{\partial E}{\partial c_{i_1,\ldots,i_k}} = \epsilon (y_{\text{computed}}^r - y_{\text{desired}}^r) \cdot \prod_{j=1}^{n} N_{i_j,n_j}(x_{i_j}^r), \tag{17}$$

where ϵ denotes the *learning rate*. Since the second partial derivation of $c_1^r, \ldots, c_k^r$ is always positive, the error function (16) is convex in weight space. If the inputs $(x_1^r, \ldots, x_n^r)$ are *linearly independent* there exists, due to the convexity of the MSE performance surface, only one global minimum and no local minima. On the other hand, if the autocorrelation matrix is singular, which occurs when the inputs x_r are *linearly dependent*, there exists an infinite number of global minima (Fig. 6), but still no local minima, in weight space.

3 Finding (Sub)Optimal Knot Vectors

By freeing knot positions, the task of finding coefficients and accurate knot vectors to fit training data becomes a non-linear minimisation problem. To solve this problem we follow a strategy

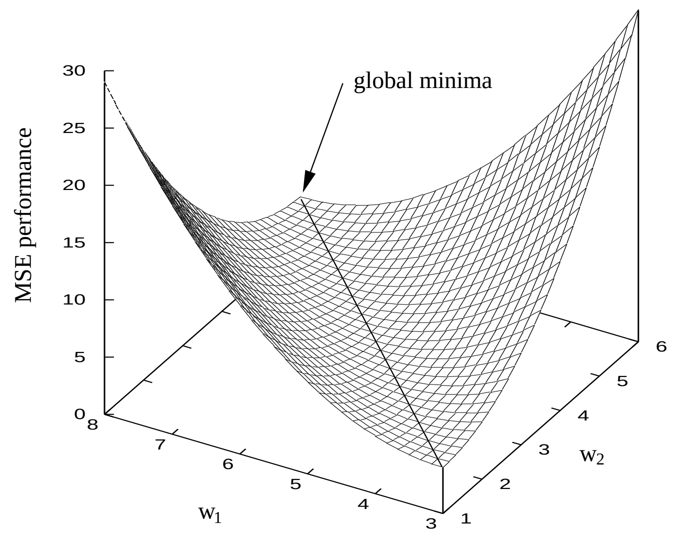


Figure 6: A singular MSE performance surface.

of problem splitting. We first consider the underlying model $\delta(\lambda)$ of the network and then compute the B-spline coefficients to minimise $\delta(c)$. Instead of using constrained least-square methods (constrained because of avoiding to “ride” on the gradient edge of coincident knots [2]), we try to estimate the knot positions by using GAs, because GAs are both, theoretically and empirically proven to provide the means for efficient search, even in complex spaces [3]. Therefore each individual, here a B-spline network with its special knot distribution, represents one point in the search space.

3.1 The Genetic Algorithm

We applied the basic GA introduced by Holland [4] with some modifications as follows (Fig.7):

- We use *gray coding* instead of standard binary code while representing coded chromosomes, a common modification.
- To bypass the undesirable effect of the increasing probability along the descendent chromosome string to receive a changed allele (bit) (thus the conjointly heredity of genes decreases when the distance of their position increases) while using n-point crossover, we used *uniform crossover*. This kind of crossover has no positional and a high distributional bias, so that a high blending rate between participant chromosomes is granted. This leeds to an algorithm permanently producing solutions which explore new locations by bridging even great distances of the search space – a characteristic which is convenient for small populations [5].
- Instead of using fitness-proportional selection it has some advantages to use *tournament selection*. This selection schema draws ξ individuals ($2 \leq \xi \leq \mu$) with a probability $\frac{1}{\mu}$ from the current population and copies the individual with the best fitness into the mating

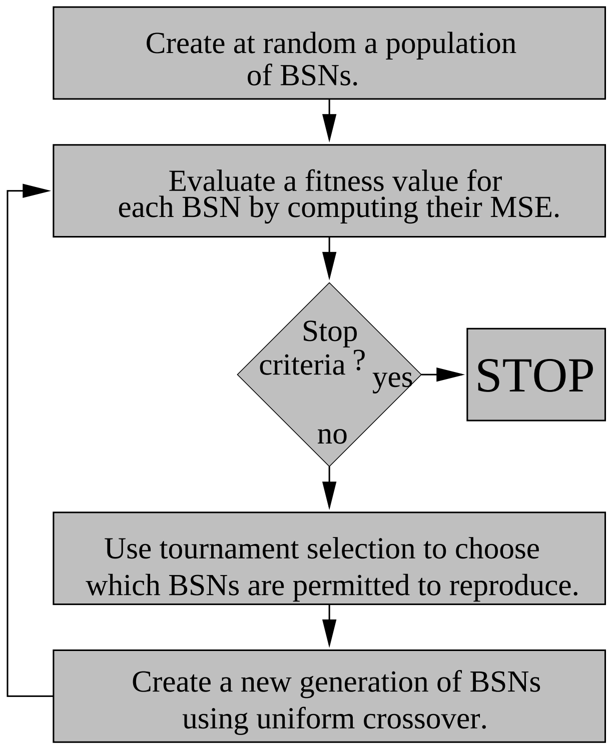

Create at random a population of BSNs.
Evaluate a fitness value for each BSN by computing their MSE.
Stop criteria ?
yes
STOP
no
Use tournament selection to choose which BSNs are permitted to reproduce.
Create a new generation of BSNs using uniform crossover.

Figure 7: Flowchart of the applied genetic algorithm.

pool. Besides saving computational power as a result of no need to sort the population (as in ranking based selection schemes), it is easier to bias the *takeover time*.

3.2 Chromosome Encoding for the Knot Placement Problem

To minimise $\delta(\lambda)$ each individual consists of n knot vectors, where n is the problem dimension. Each encoded knot vector consists of 32 knots and a so-called *activation string* of 32 bit length. Which knots are used to define the current model is encoded through the activation string. Activated knots are represented by 1 and inactivated knots are represented by 0. Every knot position is encoded by 16 bit (Fig. 8) and therefore each knot can be placed on a given interval $[c_j, d_j]$ with an accuracy of $\frac{1}{2^{16}}(c_j - d_j)$. After creating a new individual the knots are arranged in increasing order. To map λ_k with a and λ_{m-k+1} with b the knot values are multiplied by an appropriate factor. The fitness value for each individual is simply computed by determine the coefficients by solving the overdetermined linear system (10). Using this coefficients the mean square error is evaluated and the fitness for one individual is set equal to $\frac{1}{MSE}$.

4 Numerical Results

For comparison the above described GA was applied on one and two-dimensional functions also used in [6] (for function definitions see [6]). These one-dimensional and two-dimensional functions are depicted in Fig. 9 and 10 respectively.
In these examples, parameter settings were chosen as crossover probability $= 0.75$, mutation probability $= 0.0005$, $\mu = 40$ and $\xi = 3$. A maximum generation index of 200

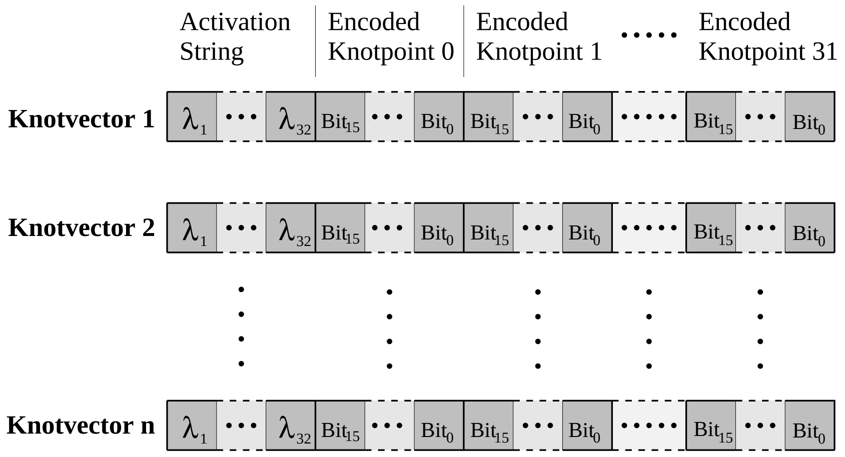


Figure 8: Encoded BSN.

was used as stop criterion. The MSE of each adapted BSN represents the average MSE of 3 runs. The comparative results are summarised in Tab. 1.
By searching for an optimal configuration of B-splines in two or more dimensions, it is important to consider how many B-splines are used for each input. In the examined two-dimensional cases we used the simple constraint of allowing no more then 64 rules. By using more input dimensions, the problem of finding (sub)optimal knot numbers becomes more relevant because of the exponentially increasing combination possibilities and thus an exponentially increasing search space.

Function	Rules	Used Membership Function			
		Equidistant B-splines	Adapted B-splines	Best of [5]	Worst of [5]
f_1	12	0.02	0.007	0.08	0.7
f_2	12	0.0005	0.00003	0.02	0.3
f_3	12	0.0008	0.000048	0.002	0.03
f_4	12	4.9	0.04	0.1	10
f_5	12	0.04	0.0002	0.01	1
f_6	12	0.6	0.012	0.1	0.4
g_1	64	766	26.36 (60 rules)	9	26
g_2	64	10.91	2.8 (60 rules)	7	19
g_3	64	5.78	0.01 (60 rules)	1.2	6

Table 1: MSE results from [6] in comparison to results of an equidistant distributed B-spline network and results of a GA modified B-spline network.

5 Discussion

Our experiment shows that using equidistant distributed B-spline networks can already achieve good results. Function f_1, f_2 and f_3 are approximated better than all approximations in [6] and the other function approximations (except g_1) receive a position in the leading third of all results. By using the above described GA to find a better distribution of MFs along the input interval,

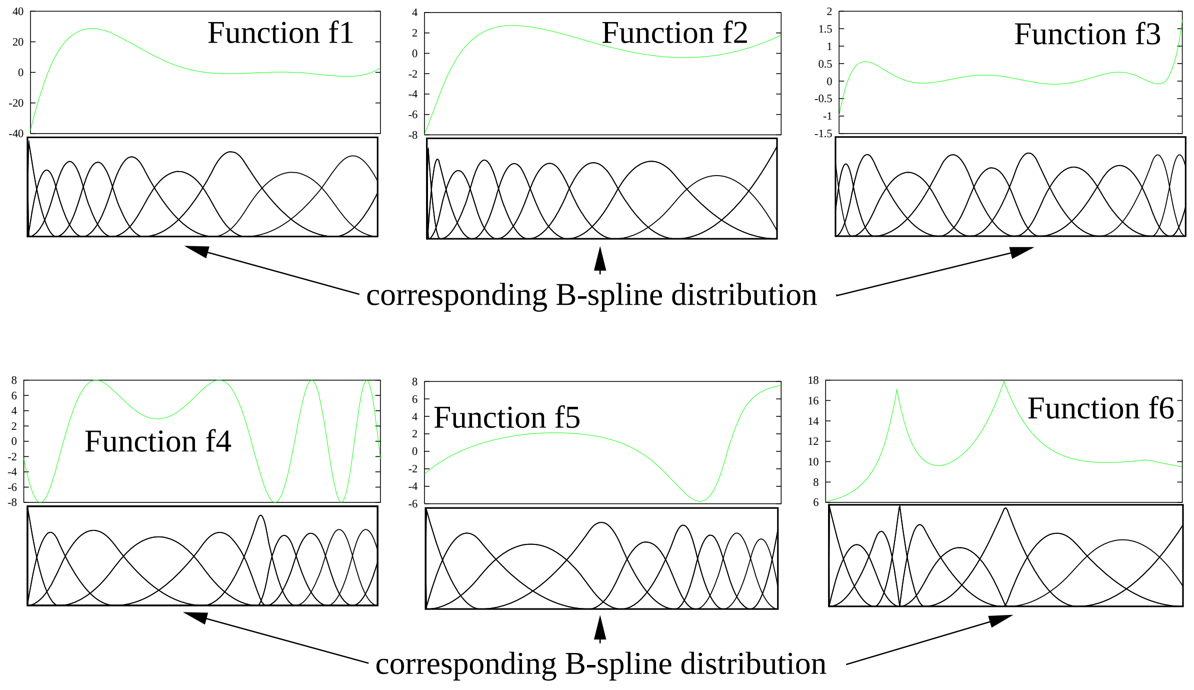


Figure 9: One dimensional functions and their encountered B-spline model.

we partially achieve results beating the best MSE results in [6] by two decimal magnitudes. All but one function (g_1) are approximated more exactly in comparison with the best MFs used in [6]. By viewing the functions and the appropriate encountered B-spline distributions (Fig. 9 and 10), it can be recognised that the GA has arranged the B-splines so that more B-splines lay in input interval spaces with high complexity. Furthermore more knots are arranged on the input dimension with higher complexity (Fig. 10).

Literature

[1] Martin, R.S., Wilkenson, J.H. Symetric decomposition of positive definite bandmatrices. In *Numerische Mathematik, 7, p.355-361*, 1965

[2] Jupp, D.L.B. The 'Lethargy' theorem - a property of approximation by γ-polynomials. In *Journal of Approximation Theory, 14, p.204-217, 1975*

[3] Goldberg, D.E. Genetic Algorithms in Search Optimization, and Machine Learning. *Reading, MA: Addison-Wesley*, 1989

[4] Holland, J.H. Adaption in Neural and Artificial Systems. *Ann Arbor, MI: University of Michigan Press*, 1975

[5] Schwefel, H.P. Parallel Problem Solving from Nature. *LNCS 496, p.38-47, Springer Verlag, Berlin*, 1991

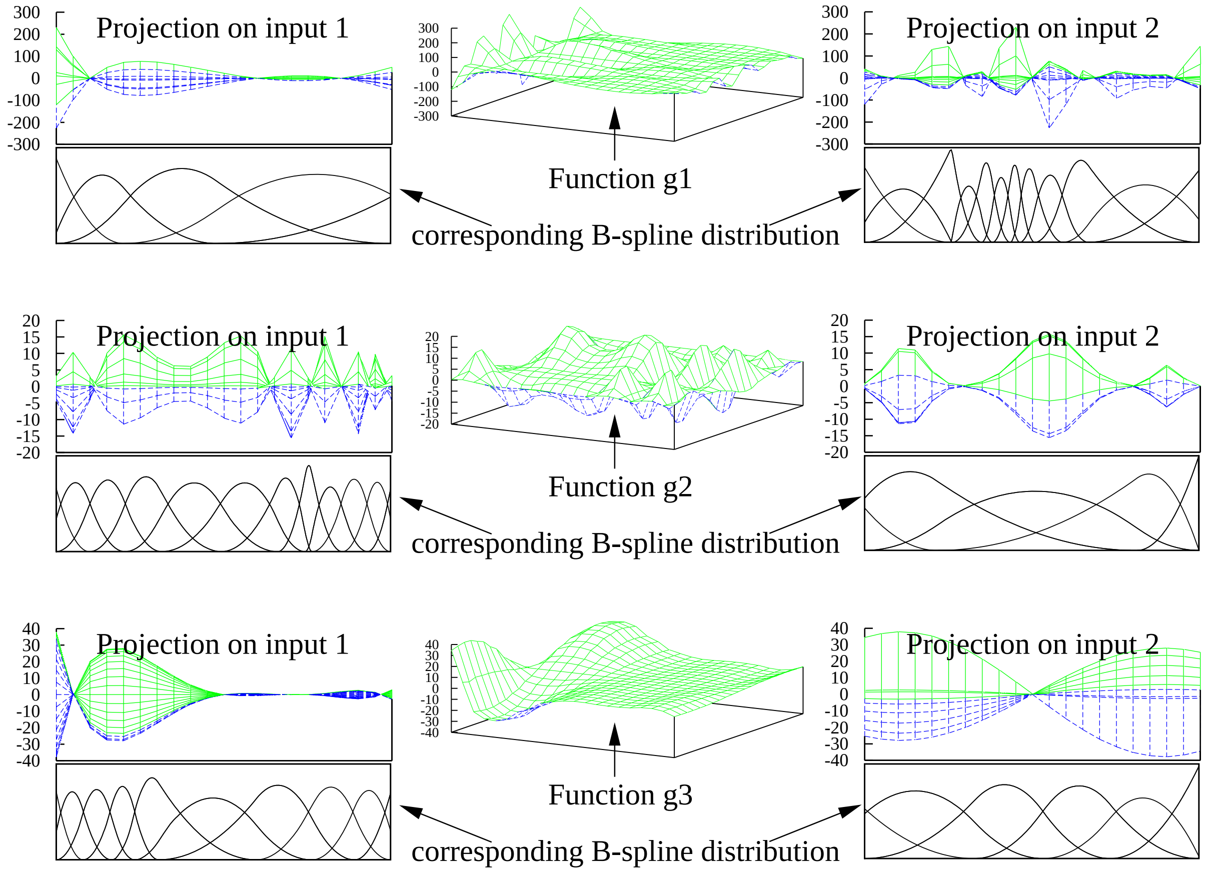


Figure 10: Two dimensional functions and their encountered B-spline model.

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

[7] Zhang, J. and Knoll, A. Constructing fuzzy controllers with B-spline models - principles and applications. *International Journal of Intelligent Systems*, 13(2/3):257–285, Feb./Mar. 1998.